UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MICROSOFT CORPORATION AND MICROSOFT MOBILE INC.,
Petitioner,

v.

KONINKLIJKE PHILIPS N.V.,

Patent Owner.

DECLARATION OF LOREN TERVEEN IN SUPPORT OF PETITIONS FOR *INTER PARTES* REVIEW OF U.S. PATENT NOS. 6,690,387 B2 AND 7,184,064 B2

LIST OF APPENDICES

Appendix	Description
A	Listing of ’387 Patent Claims
B	Listing of ’064 Patent Claims
C	Curriculum Vitae of Loren Terveen, Ph.D.
D	B. Shneiderman, “Direct Manipulation: A Step Beyond Programming Languages,” IEEE, Computer vol.16, iss. 8 at 57-69 (August 1983).
E	E. Hutchins, J. Hollan, and D. Norman, “Direct Manipulation Interfaces,” Human-Computer Interaction vol. 1.4, 311-338 (1985).
F	Excerpt from J. Nielsen, USABILITY ENGINEERING, Academic Press, 1993.
G	J. Foley, V. Wallace, and P. Chan, “The human factors of computer graphics interaction techniques,” *IEEE computer Graphics and Applications* vol.4.11, pp. 13-48 (1984).
H	Excerpt from B. Shneiderman and C. Plaisant, DESIGNING THE USER INTERFACE, Addison Wesley (5th ed. 2010).
I	U.S. Patent No. 6,724,403 (“Santoro”)
J	U.S. Patent No. 5,241,303 (“Register”)
K	Excerpt from The Authoritative Dictionary of IEEE Standards Terms, Seventh Edition, 2000
L	Excerpt from Webster’s New World Computer Dictionary, Ninth Edition, 2001

I. INTRODUCTION

1. I, Loren Terveen, have been retained by Petitioners Microsoft Corporation and Microsoft Mobile Inc. ("Microsoft") to investigate and opine on certain issues relating to United States Patent Nos. 6,690,387 ("the '387 patent") and 7,184,064 ("the '064 patent") (collectively, the "Challenged Patents") in Microsoft's Petitions for *Inter Partes* Review of those patents. Each Petition requests that the Patent Trial and Appeal Board ("PTAB" or "Board") review and cancel all claims of the underlying challenged patent.[1]

2. I am being compensated for my work on this matter by Microsoft for consulting services including time spent testifying at any hearing that may be held. I am also reimbursed for reasonable and customary expenses associated with my work in this case. I receive no other forms of compensation related to this case. My compensation does not depend on the outcome of this *inter partes* review or the co-pending district court litigation, and I have no other financial interest in this *inter partes* review.

[1] Charts listing the claims of the '387 patent and the '064 patent, numbered with the conventions used in this Declaration and the accompanying Petitions, are attached to this Declaration as Appendices A and B, respectively.

3. I understand that the ’387 and ’064 patents have been assigned to Koninklijke Philips Electronics N.V.

4. This declaration is based on the information currently available to me. To the extent that additional information becomes available, I reserve the right to continue my investigation and study, which may include a review of documents and information that may be produced, as well as testimony from depositions that have not yet been taken.

II. QUALIFICATIONS AND EXPERIENCE

5. I have summarized in this section my educational background, career history, publications, and other relevant qualifications. My full curriculum vitae is attached as Appendix C to this report.

A. Education and Experience

6. I received a B.A. in Computer Science, Mathematics, and History from the University of South Dakota in 1984, a M.S. in Computer Science from the University of Texas in 1988, and a Ph.D. from the University of Texas in Computer Science in 1991.

7. I am a member of the Association for Computing Machinery (ACM), the oldest, largest, and most prestigious computing society in the world. I am the President of ACM’s Special Interest Group on Computer-Human Interaction, one of its largest and most active special interest groups. I also am a member of the

ACM Council, the highest governing body of the ACM. I received the ACM Distinguished Scientist Award in 2009.

8. My research and teaching focus on human-computer interaction, user interface design, and social computing. I have several decades of experience in these specialties of computer science in both industry and academia. I worked for AT&T Laboratories from 1991 through 2002, during which time I conducted research and developed systems that solved problems in software engineering, web information seeking and organization, and recommender systems. In all my research, I designed, implemented, and tested graphical user interfaces, including on mobile devices.

9. I have been employed full-time as a professor at the University of Minnesota since 2002; my current title is Distinguished McKnight University Professor. I teach classes in computer science, human-computer interaction and social computing, and have conducted, supervised, and published research in the field. My research has been published in numerous journal and conference papers, as well as in a book I co-authored entitled "Foundational Issues in Artificial Intelligence and Cognitive Science: Impasse and Solution." I have served on the editorial board of ACM's Transactions on Human-Computer Interaction and the Communications of the ACM, and have served as a reviewer for numerous journals, including ACM Computing Surveys, IEEE Transactions on Data and Knowledge Engi-

neering, the International Journal of Human-Computer Studies, and the Journal of Computer-Supported Cooperative Work.

10. I am a listed inventor on nine patents, including those related to the computer graphical environment (e.g., U.S. Patent Numbers 5,680,530, 5,806,060, 5,809,492, and 6,256,648).

11. I have consulted on over 10 intellectual property cases that have dealt with topics in user interface design, recommender systems, web information systems, and set top boxes. During these cases, I have testified before judges, been deposed, and written multiple expert reports.

III. INFORMATION CONSIDERED

12. In formulating my opinions in this matter, I have reviewed the following materials:

Exhibit	Description
1001	U.S. Patent No. 6,690,387 ("the '387 patent")
1002	U.S. Patent No. 7,184,064 ("the '064 patent")
1003	Claims of '387 and '064 Patents
1005	U.S. Patent No. 7,450,114 ("Anwar")
1006	Japanese Patent Application Publication No. H06-309138 ("Narutaka")
1007	Japanese Patent Application Publication No. S63-206827 ("Westerman")
1008	U.S. Patent No. 6,943,778 ("Astala")
1009	European Patent Application No. 880,091A2 ("Korhonen")

Exhibit	Description
1010	Ivan E. Sutherland, "Sketchpad, a Man–Machine Graphical Communication System," Ph.D. Thesis, Massachusetts Institute of Technology, January 1963, reprinted in Technical Report No. 574 (Univ. of Cambridge Computer Laboratory 2003)
1011	E.A. Johnson, "Touch Displays: A programmed Man-Machine Interface, Ergonomics," 10:2 pp.271-77 (1967)
1012	Andrew Sears and Ben Schneiderman, "High Precision Touchscreens: Design Strategies and Comparisons with a Mouse," Int. J. Man-Machine Studies vol. 34, pp.593-613 (Academic Press Ltd. 1991)
1013	Margaret Minsky, "Manipulating Simulated Objects with Real-world Gestures using a Force and Position Sensitive Screen," Computer Graphics Vol. 18, No. 3, July 1984 ("Minsky")
1014	Japanese Patent Application Publication No. S63-206827 ("Asami")
1015	U.S. Patent No. 6,459,424 ("Resman")
1016	U.S. Patent No. 6,278,443 ("Amro")
1017	U.S. Patent No. 6,707,449 ("Hinckley")
1018	U.S. Patent No. 5,880,411 ("Gillespie")
1019	U.S. Patent No. 6,587,093 ("Shaw")
1020	Excerpts from File History of '387 Patent
1021	Excerpts from File History of '064 Patent
1022	Excerpts from Patent Owner's Opening Claim Construction Brief, *Koninklijke Philips N.V. v. Acer Inc.*, No. 1:15-cv-1170-GMS (D. Del. Mar. 3, 2017), Dkt. 140
1023	Excerpts from Claim Construction Order, *Koninklijke Philips N.V. v. Acer Inc.*, No. 1:15-cv-1170-GMS (D. Del. July 11, 2017), Dkt. 241
1024	U.S. Pat. No. 6,310,610 ("Beaton")
1025	U.S. Pat. No. 6,073,036 ("Heikkenen")

Exhibit	Description
1026	Excerpts from Infringement Contentions, *Koninklijke Philips N.V. v. Acer Inc.*, No. 1:15-cv-1170-GMS (D.Del. Dec. 9, 2016)

13. I have reviewed the Petitions and the prior art, documents, patents, and publications listed in the Appendices and the body of this Declaration. I have also reviewed the Petitions and expert declaration filed in Cases IPR2017-00409 and IPR2017-00408.

14. I also refer to my own expertise and experience as reflected in my curriculum vitae, which is attached as Appendix C to this Declaration.

15. In connection with live testimony in this proceeding, should I be asked to provide it, I may use as exhibits various documents that refer to or relate to the matters contained within this Declaration, or which are derived from the results and analyses discussed in this Declaration. Additionally, I may create or supervise the creation of certain demonstrative exhibits to assist me in testifying.

16. I am prepared to use any or all of the above-referenced documents, and supplemental charts, models, and other representations based on those documents, to support my live testimony in this proceeding regarding my opinions covering the Challenged Patents. If called upon to do so, I will offer live testimony regarding the opinions in this Declaration.

IV. STATEMENT OF LEGAL PRINCIPLES

17. I am not an attorney and offer no legal opinions. For purposes of this report, I have been informed about certain aspects of the law for my analyses and opinions. I set forth my understanding of the law regarding patent invalidity below.

A. Claim Construction

18. Microsoft's counsel has advised me that, when construing claim terms of an unexpired patent, a claim subject to *inter partes* review receives the "broadest reasonable interpretation (BRI) in light of the specification of the patent in which it appears."

B. Priority Date

19. I have been informed and understand that a U.S. patent application may claim the benefit of the filing date of an earlier foreign patent application only if two requirements are met. First, the U.S. patent application must have been filed within one year of the foreign parent application to which it claims priority. Second, the foreign patent application must disclose each limitation of the claimed invention of the U.S. patent application. If one or both requirements are not met, the U.S. patent application may not properly claim priority back to the filing date of the earlier filed foreign patent application.

20. I have been informed and understand that priority is determined on a claim-by-claim basis. In other words, certain claims of a U.S. patent may be enti-

tled to claim priority to a foreign patent application, while other claims of the same U.S. patent may not be entitled to claim priority to the foreign patent application.

21. I have been further informed that (for pre-AIA patent applications filed prior to March 16, 2013) a patent is invalid if it is described in a printed publication in any country more than one year before the date of the actual filing of the application in the United States, regardless of the date the applicant conceived of the claimed invention. I also understand that the patent laws were recently amended by the America Invents Act (AIA), but that earlier statutory requirements still apply for "pre-AIA" patents. I am informed that the patents asserted in this case are "pre-AIA," and that the "pre-AIA" requirements are controlling. Therefore, unless otherwise stated, my understanding of the law regarding patent invalidity as set forth in this Declaration relates to the pre-AIA requirements.

C. Invalidity Based on Prior Art

1. Anticipation

22. It is my understanding that a claim directed to subject matter that is not new or novel based on a single prior art reference is said to be "anticipated by the prior art" under 35 U.S.C. § 102.

23. It is my understanding that in order for a claim to be invalid as anticipated by the prior art, every element of that claim must be found in a single prior art reference or system arranged as in the claim. It is my further understanding that

for anticipation, each element of a claim must be found explicitly or inherently in that single prior art reference. In other words, it is my understanding that in determining whether a single prior art reference anticipates a patent claim, one should take into consideration not only what is expressly disclosed in that item, but also what inherently occurred as a natural result of the practice of the system or method disclosed in that item.

24. It is my understanding that to establish such "inherency," the evidence must make clear that the missing descriptive matter is necessarily present in the prior art and that it would be so recognized by persons of ordinary skill in the art.

25. I have been informed and understand that there are several ways that a patent claim may be anticipated. The ones that are relevant to this declaration are summarized below.

- One way to anticipate a patent claim requires a prior art reference that was either known or used by someone other than the patent applicant in the U.S., or patented or described in a printed publication in the U.S. or a foreign country, before the alleged invention by the patent applicant of the subject matter recited in the claim.
- Another way to anticipate a patent claim requires a prior art reference that was either patented or described in a printed publication in the U.S. or a foreign

country, or was in public use or on sale in the U.S., more than one year prior to the U.S. patent application filing date.

- Another way to anticipate a patent claim requires a prior art reference that was application for a patent in the U.S., which has since been published, where the application was filed before the priority date of the relevant patent claim; or a prior art reference that is a patent granted on an application which was filed in the U.S. before the priority date of the relevant patent claim.

2. Obviousness

26. I am informed and understand that a patent cannot be properly granted for subject matter that would have been obvious to a person of ordinary skill in the art at the time of the alleged invention, and that a patent claim directed to such obvious subject matter is invalid under 35 U.S.C. § 103. It is also my understanding that in assessing the obviousness of claimed subject matter, one should evaluate obviousness in light of the prior art from the perspective of a person having ordinary skill in the art at the time the alleged invention was made (and not from the perspective of either a layman or a genius in that art). It is my further understanding that the question of obviousness is to be determined based on:

- The scope and content of the prior art;
- The difference or differences between the subject matter of the claim and the prior art (whereby in assessing the possibility of obviousness one should con-

sider the manner in which a patentee and/or a Court has construed the scope of a claim);

- The level of ordinary skill in the art at the time of the alleged invention of the subject matter of the claim; and
- Any relevant objective factors (the "secondary indicia")[2] indicating nonobviousness, including evidence of any of the following: commercial success of the products or methods covered by the patent claims; a long-felt need for the alleged invention; failed attempts by others to make the alleged invention; copying of the alleged invention by others in the field; unexpected results achieved by the alleged invention; praise of the alleged invention by the alleged infringer or others in the field; the taking of licenses under the patent by others and the nature of those licenses; expressions of surprise by experts and those skilled in the art at the subject matter of the claim; and whether the patentee proceeded contrary to accepted wisdom of the prior art.

[2] I am not currently aware of any evidence regarding secondary indicia of non-obviousness related to the claims of the Challenged Patents. Should Patent Owner attempt to present any alleged evidence of secondary indicia of nonobviousness, however, I reserve the right to opine on any such alleged evidence.

- Any relevant objective factors (the "secondary indicia") indicating obviousness: independent invention of the claimed invention by others before or at about the same time as the named inventor thought of it; and other evidence tending to show obviousness.
- I understand that for objective evidence of secondary indicia to be accorded substantial weight, its proponent must establish a nexus between the evidence and the merits of the claimed invention. I also understand that, where the offered secondary indicia actually results from something other than what is both claimed and novel in the patent claim, there is no nexus to the merits of the claimed invention.

27. It is my understanding that to determine whether it would have been obvious to combine known elements in a manner claimed in a patent, one may consider such things as the interrelated teachings of multiple patents, the effects of demands known to the design community or present in the marketplace, and the background knowledge of one of ordinary skill in the art.

28. It is further my understanding that determining obviousness is expansive and flexible. That is, granting patent prosecution to advances that would occur in the ordinary course without real innovation impedes progress and may, in the case of patents combining previously known elements, deprive prior inventions of their value or utility. I also understand that there is no requirement for an obvi-

ousness analysis to find precise teachings that are directed to specific subject matter of a claim; rather, the common sense, inferences, and creative steps that a person of ordinary skill in the art would employ should be taken into account. An obviousness analysis therefore does not require rigid rules that ignore common sense.

29. It is my understanding that a need or problem known in the field at the time of an alleged invention can provide an obvious reason to combine elements in the manner claimed. A patent's subject matter would have been obvious if at the time of the invention, there was a known problem for which the patent claims encompassed an obvious solution. Further, if a patent claims a structure known in the prior art that only substitutes one element for another that is known in the field, I understand that the combination would have been obvious unless the result is unexpected and fruitful. Therefore, a predictable variation of prior art is obvious.

30. It is also my understanding that if a technique was used to improve a device or method, and if a person of ordinary skill in the art would recognize that the technique would improve similar devices or methods in the same way, using the technique is obvious unless applying it is beyond the person's skill.

31. It is my further understanding that an alleged improvement claimed in a particular patent must do more than use prior art elements according to their established functions. That is, when a patent simply arranges old elements with each

performing the same function it had been known to perform and yields no more than one would expect from such an arrangement, the combination is obvious.

32. It is my additional understanding that items may have obvious uses beyond their primary purposes. Common sense teaches that familiar items may have obvious uses beyond their primary purposes, and in many cases a person of ordinary skill will be able to fit the teachings of multiple pieces of prior art together like pieces of a puzzle. I understand that neither a particular motivation nor an avowed purpose of a patentee controls how a piece of prior art may be used.

33. I also understand that if a combination was obvious to try, it may be obvious. I understand that obviousness is therefore not confined to a formalistic conception of teaching, suggestion, and motivation or by overemphasizing published publications and the explicit content of issued patents.

34. It is also my understanding that in developing opinions as to whether or not certain claimed subject matter would have been obvious, each claim of a given patent should be considered in its entirety and separately from any other claims. In so doing, it is my further understanding that while I should consider any differences between the claimed invention and the prior art, I should also assess the obviousness or non-obviousness of the entirety of a claim covering an alleged invention, not merely some portion of it.

35. Also, when there is a design need or market pressure to solve a problem and there are a finite number of identified, predictable solutions, a person of ordinary skill in the art would have good reason to pursue the known options within his or her technical grasp. If that pursuit likely leads to the anticipated success, it is likely the alleged invention is a product not of innovation, but of ordinary skill and common sense. In that instance, the fact that a combination was obvious to try might show that it was obvious.

36. I have also been informed that, when a work is available in one field of endeavor, design incentives and other market forces can prompt variations of it, either in the same field or a different one. If a person of ordinary skill in the art can implement a predictable variation matching a patent's claim, it is likely that the claim is invalid for being obvious.

37. It is my further understanding that multiple prior art references can be combined to show that a claim is obvious. Any need or problem known in the field and addressed by a patent claim can provide a reason for combining multiple references in the manner claimed. To determine whether there was an apparent reason to combine those references in the way a patent claims, it is my understanding that I can look to interrelated teachings of multiple pieces of prior art, to the effects of demands known to the design community or present in the marketplace, and/or to the background knowledge possessed by a person of ordinary skill in the art.

38. It is my further understanding that a single reference can alone render a patent claim obvious if any differences between the reference and the claim would have been obvious to a person of ordinary skill in the art at the time of the alleged invention (that is, if the person of ordinary skill in the art could adapt the reference to meet the claims of the patent by applying known concepts to achieve expected results).

39. It is my further understanding that one of ordinary skill in the art is not confined to prior art that attempts to solve the same problem as the patent claim since common sense teaches that familiar items may have obvious uses beyond their primary purposes.

40. I am also aware that another way to decide whether one of ordinary skill in the art would combine what is described in various items of prior art is whether there is some teaching, suggestion, or motivation in the prior art for a skilled person to make the combination covered by the patent claims. Motivation can be implicit. In other words, such motivation need not be explicitly stated.

41. I have been informed and understand that hindsight reasoning is not an appropriate basis for combining references to form an obviousness combination.

V. LEVEL OF ORDINARY SKILL IN THE ART

42. I understand that in analyzing questions of invalidity and infringement, the perspective of a person having ordinary skill in the art ("POSITA") is often implicated, and the Court may need assistance in determining that level of skill.

43. I have been informed and understand that factors that may be considered in determining the level of ordinary skill in the art include: (1) the educational level of the inventor; (2) the type of problems encountered in the art; (3) prior art solutions to these problems; (4) the rapidity with which innovations are made; (5) the sophistication of the technology; and (6) educational level of active workers in the field. A person of ordinary skill in the art is also a person of ordinary creativity.

44. Based upon my experience and training in the field of user interface design, as well as my reading of the Challenged Patents, it is my opinion that a person of ordinary skill with regard to the subject matter of the Challenged Patents, at the time of the alleged invention, would have had: (a) a master's degree in electrical engineering, computer engineering, computer science or the equivalent, with coursework covering programming user interfaces including interfaces on touch-sensitive devices, or (b) a bachelor's degree in one of those fields and at least two years of industry experience working with programming user interfaces, including for touch-sensitive devices; or (c) four years of industry experience programming user interfaces, including for touch-sensitive devices.

45. I am a person of at least ordinary skill in the art and was so at the date for which the Challenged Patents claim priority (December 28, 2001). As demonstrated by my qualifications (discussed above) and my curriculum vitae attached hereto as Appendix A, I am aware of the knowledge and skill possessed by such a person of ordinary skill in the pertinent art at the time of the purported inventions claimed by the Challenged Patents. In performing my analysis, I have applied the standard set forth above.

VI. GENERAL BACKGROUND OF THE RELEVANT TECHNOLOGY

A. The Challenged Patents' Priority Date

46. The '387 patent was filed on December 28, 2001, and issued on February 10, 2004. The '064 patent was filed on December 16, 2003, and issued on February 27, 2007. The '064 patent purports to be a continuation of the '387.

47. Although I express no opinion on the correctness of this alleged priority date with respect to particular claims, I have used December 28, 2001, as the priority date for the purpose of this declaration and for the overview of the related technology below.

48. The Challenged Patents relate generally to systems and methods for scrolling displayed information in response to a user touch. Below, I briefly describe the state of the art in 2001 as it related to networking and the remote management of electronic devices via a server.

B. State of the Art from the Perspective of Ordinary Skill

49. The field of human-computer interaction was significantly well-developed by December 2001. Since the early 1980s, the focus of research and development had been on creating interaction techniques and implementing systems that made it easy and natural for people to enter, access, and navigate information on computers. "Direct manipulation" was the unifying conceptual framework driving the design and implementation of these systems. Its hallmarks are that the objects of interest in an application—for example, documents and text in a word processing system, photographs in a photo-editing application, etc.—are visibly represented on a display, and users operate on these objects—for example, to edit text, delete a document, adjust a photo's contrast—by "pointing" at them. *See generally* Apps. D, E, F.

50. It was well understood there were several fundamental interaction tasks that **any** pointing device had to support (*see generally* Apps. G, H), including:

- *Selecting* objects to operate on;
- *Postioning* – indicating a point to, for example, place a new object or window.
- *Gesturing* – for example, "swiping" to indicate turning a page in a multi-page document or scrolling through a document too large to fit in the available display space.

This is a crucial observation because it meant that interaction techniques developed using one technology or device (for example, “clicking” and “dragging” with a mouse) found direct analogues on other technologies and devices (for example, “tapping” and “dragging” with one’s finger).

51. Therefore, by December of 2001, companies developing touch input systems had a well-developed set of concepts (such as “direct manipulation”, “pointing”, and “gesturing”), interaction techniques (such as “clicking” and “dragging”), and implemented examples to draw on in creating natural user interfaces that interpreted user intent and mimicked authentic gestures as closely as possible.

52. The pre-history of these techniques began in the early 1960s; the “Sketchpad” drafting tool developed at MIT allowed a user to draw shapes directly on a computer screen using a “light pen.” Ex. 1010 at 9. Just a few years later, researchers at the Royal Radar Establishment in the United Kingdom developed a capacitive touch screen, called the “Touch Display.” Ex. 1011. The Touch Display reported a “faster and more accurate means of communicating” between operator and system, and predicted use in air traffic control and “wider application in other systems.” *Id*. at 8.

53. Initially, limitations of the touchscreen hardware—such as a lack of resolution and inaccurate pixel reporting—resulted in high input error rates. E.g., Ex. 1012 at 3-4. But as hardware capabilities improved, touchscreen devices be-

came more accurate. In the 1980s, an MIT researcher used a touch-sensitive screen to interpret multiple finger-touch gestures: a "move" along a particular line; a "tap" that indicated selection; a long press that indicated "a desire to move the selected button on the screen"; and a "flick" that "sen[t] an object to another part of the screen." Ex. 1013 ("Minsky") at 2, 4, 9. The force of the user's flick determined the object's initial velocity, and the object then "slow[ed] down by 'friction.'" *Id*. at 9. A 1988 Japanese patent application filed by Fujitsu ("Asami," Ex. 1014) disclosed a touch panel "display scrolling system" where the speed of the displayed image was "approximated to a movement speed of a pointing device such as a finger" and "gradually falls over time." *Id*. at 1. Asami recognized that it was already well-known to scroll an image at the speed of a touch. *E.g.*, *id*. at 2 (in a "conventional display scrolling system … [t]he movement speed of the finger therefore corresponded one-to-one with the movement speed of the display image being scrolled"). Asami aimed to improve this standard by providing a "more natural movement" of the image, where "the display image moves as though with an appropriate sensation of weight." *Id.*

54. In the 1990s, companies used gestures as well as touch duration to provide even more intuitive touch interfaces. One of Hewlett-Packard's patents, for example, disclosed a touch screen panel with touch-controlled scroll bar regions. Ex. 1015 ("Resman"), 6:45-51, Fig. 2. Resman recognized that "[t]ouch-

pad display screens ha[d] found wide application in [PDAs]" and predicted that this same technology "can be readily incorporated" in any computer LCD display. *Id*. at 1:13-15, 1:46-55. IBM patented a touchscreen device that allowed "the whole screen of data [to] be moved" in a touch-indicated direction "when scrolling." Ex. 1016, Abstract. Microsoft Corporation patented a touchscreen device for a computer that used differences in touch duration to invoke various operations such as sliding from the active region to cause absolute scrolling and finger taps to cause page up or page down movement. Ex. 1017 at 16:12-46.

55. Similarly, touch-sensitive systems patented by Synaptics disclosed sensing various touch gestures and associating them with specific operations. E.g., Ex. 1018 at 35:23-63 (sensing "tap" and "drag" gestures); Ex. 1019 at 12:27-13:66 (associating different touch gestures with basic scroll, "coasting" scroll, and zoom operations). The Westerman application (Ex. 1007) also integrated the ability to perform panning, zooming, scrolling, and a variety of other operations based on different touch gestures into a touchscreen system. Westerman at 1, claims 14-15. And a 1994 Toshiba Japanese application ("Narutaka," Ex. 1006) disclosed a "simpler, faster, and more intuitive" touch screen system that allowed a user to scroll an image "in accordance with the direction and amount of movement of the finger." *Id*., Abstract, [0009]. Like others had done, Narutaka interpreted short

touches as an input signal, and touches lasting "a predetermined fixed amount of time" as scroll signals. *Id*., [0019], [0020].

56. The combination of smaller size and ever-increasing functionality in PDAs and other handheld touchscreen devices meant that it was "increasingly important for a user to be able to enter commands and information … in an efficient manner. Ex. 1008 ("Astala") at 1:29-33. Thus, gesture sensing became even more of a focus in connection with handheld touchscreen devices of the late 1990s. The Anwar patent (Ex. 1005) disclosed a handheld computing device that displayed documents and graphical tools, such as magnifiers, to a user. Anwar, Abstract. Anwar's device distinguished between various touch gestures to allow a user to click on, drag, and scroll documents and other displayed items. *E.g*., *id*. at 14:3-32. Nokia's Korhonen application disclosed a mobile touchscreen device, such as a cellular telephone, that scrolled lists in the direction of a user touch. Korhonen, Abstract, Figure 2. And the Astala patent, also to Nokia, used the location and duration of a touch input to differentiate between select, drag, activate, and other operations. Astala, Abstract, 9:3-18.

VII. THE CHALLENGED PATENTS

A. Background and General Description of the Challenged Patents

57. Below, I provide a general background discussion of the Challenged Patents. Each patent has the same specification. The claims of the '387 and '064

patents are virtually identical to each other, as indicated by Exhibit 1003, which compares the two sets of claims side by side. The primary difference is that the claims of the ’064 patent omit the “keyboard” limitations from the corresponding claims of the ’387 patent. All limitations of the ’064 patent claims are found in the claims of the ’387 patent, in nearly verbatim language. Thus, except where specifically noted in this Declaration, a POSITA would have understood the respective claims of the ’387 and ’064 patents to recite effectively identical subject matter. For simplicity, the citations in the following discussion are to the ’387 patent.

58. The ’387 patent describes a touchscreen system for scrolling an image displayed on a screen. ’387 patent, 1:8-12, 1:53-57, 1:65-2:1. The patent’s preferred embodiment is a list. *Id*. at Abstract. When a user sweeps his finger across the screen, the system senses the user touch and scrolls the image in response to the touch. *Id*. at 1:53-57.

The ’387 patent discloses that its basic scrolling functionality was already well-known. *Id*. at 1:18-22 (“it has further become well-known to cause the image of the list to ‘scroll’ past the screen … until a desired section of the list, or portion of a line, appears on the screen”). The patent teaches that it was well-known to scroll the screen in the direction and speed of user input, for example in cursor-based systems (*id*. at 1:22-40):

> It is known that the systems and methods currently being used to control the scrolling motion of the screen image are subject to numerous limitations and disadvantages. For example, in one system a cursor may be positioned at one edge of the screen and then moved toward the opposite edge while holding down a selected "mouse" button, thereby engaging and "dragging" the screen image in a desired direction. It is well known that such displacement of the screen image is slow and cumbersome except for relatively slight relative movements. Another system in current use activates an automatic continuous "scrolling" motion of the image when the cursor is positioned on a specific portion of the image, while a selected mouse button is depressed. This requires holding down the selected button until the desired portion of the screen image is displayed. A related system in current use varies the speed of the scrolling motion in accordance with the position of the cursor relative to the edge of the screen.

59. The patent also teaches that it was well-known to drag and move specific objects on a display by sensing a user touch. *Id*. at 3:34-37 ("the 'selected' item on the list will then 'stick to the finger' so that the item can be repositioned on the list by the known process of 'touch-dragging'); *id*. at 2:44-45 ("'Touch marking' is a well-known feature of scrolled display technology at this time").

60. The '387 patent contends it added a "a new … form of scrolling motion control" to these known methods (*id*. at 2:45-47) by using the speed and direction of a finger touch to determine the speed and direction of scrolling. *Id*. at 1:58-60. The patent discloses that after the finger lifts off the screen, scrolling slows down until the system senses one of a number of criteria: a still touch (touching the

screen without moving the finger); the speed decreasing to zero or to a minimum speed; or the end of the list. *Id*. at 1:60-65. The system can interpret the touch as an item selection if it lasts for less than a minimum time period. *Id*. at 2:4-8 ("If a finger is applied to the surface of the screen for a shorter period of time, for example for a period less than a minimum set time, the finger touch can be deemed to be a 'selection' of an item or 'thing' corresponding to the image displayed at the touched location."); *id*. at 3:24-27 ("If no motion occurs and the touch contact continues for less than a predetermined minimum time, the touch is treated in step 100 as a "selection" of the data term touched, and the system continues with 'selection' path 102."). If the touch lasts longer than a period of time, it is moved in line with a "known process" called "touch-dragging. " *Id*. at 3:31-37 ("if the touch contact continues for more than the first predetermined minimum time, and the finger then moves after that time, the process of the invention will proceed to step 103, in which the 'selected' item on the list will then 'stick to the finger' so that the item can be repositioned on the list by the known process of 'touch-dragging')."

61. The patent discloses that the touch can select the entire display with the finger. *Id*. at 3:65-4:3 ("if contact with the screen is not broken, the method of the invention proceeds from step 104 to step 105, wherein the entire display [not just a selected item] in effect 'sticks to the finger' so that the entire display can be moved up or down or back and forth, as the case may be, with the finger."

62. The ’387 patent asserts that it uses a new form of scrolling in combination with these various known techniques. *Id*. at 2:45-47 (“but this invention discloses its use in combination with a new, and heretofore unknown, form of scrolling motion control.”). The ’387 patent discloses sensing the speed and direction of a user touch and moving the image in that same speed and direction until scrolling is stopped by another touch, the speed dropping to zero or a minimum number, or the end of the image (*id*. at 1:53-65):

> The present invention overcomes and avoids the limitations of known control systems for scrolling electronic displays by providing a touch-screen responsive system that imparts a scrolling motion to the displayed image in response to the motion of a finger in contact with the screen. The speed and direction of motion of the finger along the screen determines the initial speed and direction of motion for the image. After the finger separates from the screen, the image continues to move in the same direction at a gradually decreasing speed until motion is stopped manually by touching the screen without movement of the finger, or the speed decreases to zero, or to a predetermined minimum speed, or until the image reaches its “end”.

63. The system can interpret the touch as selecting different operations, including dragging an item or scrolling the display in one-to-one correspondence with the finger touch. *Id*. at 3:24-37, 2:4-8, 3:65-4:3.

B. The Prosecution History

64. Neither of the ’387 or ’064 patents had substantive office action rejections based on prior art.

65. The ’387 patent was filed on December 28, 2001, and issued on February 10, 2004. The examiner allowed all claims in the first office action. He determined that “[t]he prior art of record fails to teach a touch screen for controlling scrolling wherein the initial scrolling direction and speed is based on the traveling direction and speed of the user’s finger on the display surface, and wherein the system further automatically reduce[s] the speed at a given rate as recited in the claims.” Ex. 1020 at 2.

66. The ’064 patent was filed on December 16, 2003, and issued February 27, 2007. The examiner again allowed the claims without any rejections over the prior art, other than a double-patenting rejection over the ’387 patent. Philips overcame that rejection by filing a terminal disclaimer. Ex. 1021 at 6-7.

67. The examiner did not consider any of the references discussed in the petitions during prosecution.

C. Claim Construction

68. In conducting my analysis of the asserted claims of the Challenged Patents, I have applied the legal understandings I set out below regarding claim constructions consistent with the “broadest reasonable interpretation” (BRI) stand-

ard described above, and offer them only for this *inter partes* review. The claim constructions do not necessary reflect the appropriate claim constructions to be used in litigation proceedings, such as litigation in a district court, where a different standard applies.

69. I understand that, under the BRI claim construction, claim terms are given their broadest reasonable construction in light of the specification as it would be interpreted by one of ordinary skill in the art. Claim terms must be given their ordinary and customary meaning as would be understood by one of ordinary skill in the art in the context of the entire disclosure unless such meaning is inconsistent with the specification. A patentee may rebut that presumption by providing a definition of the term in the specification with reasonable clarity, deliberateness, and precision. In the absence of such a definition, limitations are not to be read from the specification into the claims.

70. I am informed that language in a claim's preamble generally provides context for the claim and is not normally a limitation, absent any indication to the contrary in the claim. I understand that a preamble is limiting if (1) when read in the context of the entire claim, the preamble recites limitations of the claim or (2) the claim preamble is necessary to give "life, meaning, and vitality" to the claim.

71. I am informed that the USPTO's rules for *inter partes* review proceedings require the petitioner to identify how challenged claims are to be construed.

72. A summary of my opinions regarding the proposed constructions of certain terms of the challenged claims are as follows:

Term	Construction
"stopping motion program instructions associated with said microprocessor for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising (i) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (ii) an end-of-scroll signal received from said scroll format data source"	requiring sensing for each of the signals in the recited group and terminating scrolling upon whichever signal is sensed first
"timer means … to provide timing capacity for a microprocessor / computer apparatus"	timer that provides timing capacity for a microprocessor

73. A detailed explanation of my opinions regarding the proposed constructions of certain terms of the challenged claims is set forth below.[3]

[3] Again, for simplicity, the citations in the following discussion are to the '007 patent only because each of the Challenged Patents has the same specifi-

1. “Stopping Motion Program Instructions”

74. Independent claims 1, 7, and 8 of each of the Challenged Patents recite “stopping motion program instructions associated with said microprocessor for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising (i) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (ii) an end-of-scroll signal received from said scroll format data source.”

75. I understand that the Board has previously construed this phrase as “requiring sensing for *each of* the signals in the recited group and terminating scrolling upon whichever signal is sensed first.” IPR2017-00409, Paper 10 at 8 (emphasis added); *see also* IPR2017-00410, Paper 10 at 12 (construing phrase similarly as “instructions for sensing for each of the signals in the recited group and terminating scrolling upon whichever signal is received first”). I also understand that Philips proposed this construction. I have therefore used this construction in my opinions.

cation. For the same reason, the specifications of the Challenged Patents are referred to collectively as “the specification.”

2. "Timer Means"

76. Independent claims 1, 7, and 8 recite a "timer means … to provide timing capacity" for a microprocessor (claims 1 and 8) or computer apparatus (claim 7). Claim 5 further recites that the timer means and microprocessor make up a conventional computer processing unit.

77. I understand that this phrase is written in what is called "means-plus-function" format as a "means" for performing a certain function. I also understand, however, that Philips has argued that a POSITA would have understood the claim to recite structure that removes "timer means" from means-plus-function treatment, and that a POSITA would have understood this term to refer simply to "a timer." Ex. 1022 at 13. I further understand that the district court has adopted this position in the ongoing litigations, finding that a POSITA would have interpreted this phrase as referring to "structure for providing timing capacity for a microprocessor." Ex. 1023 at 5, n.9.

78. I agree that a POSITA would have understood the "timer means … to provide timing capacity" for a microprocessor or computer apparatus to be a timer. A clock is required for any microprocessor. Its primary job is to synchronize operations performed by the microprocessor—for example, so an instruction is not processed before all data is in place. Clocks are implemented as oscillating electrical signals that oscillate (i.e., go from "off" (0) to "on" (1)) at a fixed rate; for exam-

ple, a 1 Gigahertz process oscillates one billion times a second. Since every microprocessor has a clock, and the clock has a fixed 'speed' (oscillation rate), every microprocessor inherently implements a timer; for example, a ½ second timer would be implemented on a 1Gigahertz CPU as ½ billion oscillations.

79. The '387 patent itself recognizes that computers "inherent[ly] had an "internal timer facility" as of its filing date. '387 patent at 5:31-35 ("the internal timer facility now inherent in such computer apparatus, in cooperation with the programming of processing unit 12 responds to the start of motion by gradually decreasing the speed of displacement, as explained previously herein."). Figure 3 shows "Timer" 43, associated with "Microprocessor" 42:

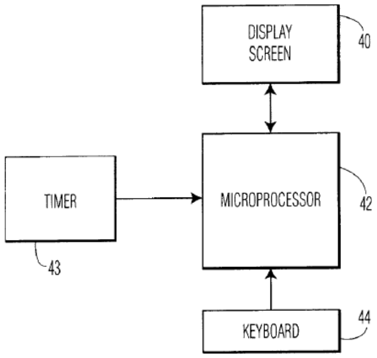


FIG. 3

80. The '387 patent states that the Figure 3 embodiment illustrates the "essential elements of the computer apparatus" and depicts "microprocessor 42 and the associated timer means 43 together, [which serve] the same function as central

processing unit 12 in Figure 2." *Id*. at 5:48-58. This "inherent" timer "cooperat[es] with the programming of processing unit 12" to provide timing capability, such as gradually decreasing the speed of the scrolling motion. *Id*. at 5:29-35.

81. In view of both the specification and my understanding of the art in general, a POSITA would have understood the "timing means" term to refer specifically to a standard timer that is inherent in computer central processing units. A POSITA would have understood this term to refer to specific structure. I would understand the broadest reasonable interpretation of the "timing means" to be "timer that provides timing capacity for a microprocessor."

VIII. OVERVIEW OF THE PRIOR ART

A. Identification and Summary of the Prior Art

1. Korhonen

82. I understand that the Korhonen application is included as Exhibit 1009 to the Petitions.

83. Korhonen discloses scrolling a list on a touch sensitive display of a mobile device in the direction of a user touch:

> The object of the invention is a method to scroll data presented on the display of a mobile station. According to the invention the control area of the display is touched with a pointing means, the pointing means is moved in contact with the control area of the display, and the displayed part of the presented information is scrolled in the

> display in the direction of the movement of the pointing means. The control area of the display can be a touch sensitive area arranged on the surface of the display, or a separate surface in another location of the mobile station for controlling the display.

Id. at 1:54-2:6.

84. Korhonen's station continues to scroll the list even after the finger lifts off the screen. *Id.* at 2:7-17. The display scrolls at the speed of the touch right before it lifted off the screen:

> According to a preferred embodiment of the invention said displayed part, which is left scrolling, is arranged to scroll at a rate measured for, or proportional to, the speed of the pointing means at the last moment before it was removed. The speed of the pointing means in the direction of the control area of the display, when the pointing means is removed, determines the initial scrolling speed of the displayed part and the scroll direction.

Id. at 2:18-26.

85. The rate of scrolling is then slowed at an exponential rate. *Id.* at 2:31-34 ("According to a preferred embodiment of the invention said displayed part left scrolling is arranged to scroll and retard by itself. Said retardation is effected by applying a suitable, for instance an exponential formula."). If the scrolling rate drops below a certain speed, scrolling stops. *Id.* at 2:35-39. Scrolling may also be "stopped by touching the display control area with a pointing means." *Id.* at 2:44-46.

86. Figure 1 shows the primary embodiment, where a scrollable list is arranged on "an imaginary cylinder 11" and only some of the list elements displayed on the screen at a given time (*id.* at 4:31-37):

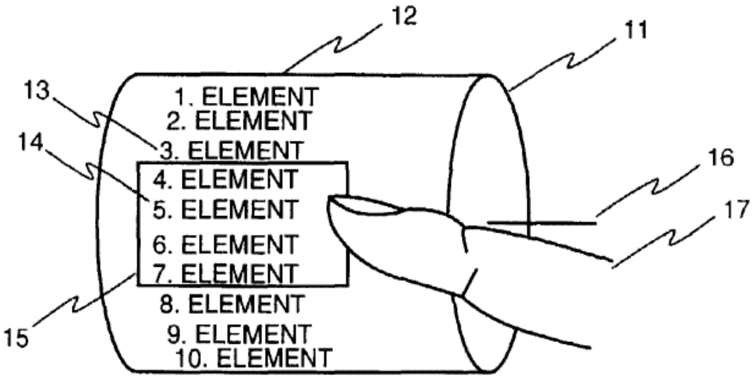


FIGURE 1

87. If the user wants to view a list element that is not visible on the display, the "list 12 and the [imaginary] cylinder are rotated around their axis" by a finger or other pointer. *Id.* at 4:37-40. The end of the list includes "an empty space … so that the repeated beginning of the list is clearly perceived, and so that a short list 12 will not be repeated on the display." *Id.* at 4:43-47.

2. Anwar

88. I understand that the Anwar patent is Exhibit 1005 to the Petitions.

89. Anwar describes a handheld touchscreen device that "exhibits a touch and feel user interface experience" by simulating tactile control over a displayed document. Anwar at 2:6-12. The touchscreen device includes a screen monitor

that detects movement across the touch-sensitive display and an interface process that detects commands to change how the document is displayed. *Id*. at 2:28-33. It also includes a velocity detector that determines the velocity of a touch movement across the touch-sensitive display and applies that velocity to the displayed document. *Id*. at 3:10-14.

90. When a user drags and then releases the document at a certain speed, the velocity detector causes the document to continue moving in the established direction. The document keeps scrolling until the user clicks on it:

> To this end, a process may employ the velocity determination to direct the parser/render 18 to redraw the document in a series of pictures that will portray the document as moving across the screen. For example, a user may drag a document at a certain speed and then release the stylus, mouse or other input device from the document. Optionally, upon release the document may stop moving. However, in an alternative practice the page may continue to move in the established direction until the user indicates that the document is to stop moving such as clicking on the document.

Id. at 14:12-22.

91. Alternatively, the scrolling speed can decrease by a constant page inertia until it reaches zero. *Id*. at 14:26-28 ("the velocity may decrease by a constant page inertia until it reaches zero velocity and page scrolling ceases").

3. Narutaka

92. I understand that Narutaka is included as Exhibit 1006 to the Petitions.

93. Narutaka describes a touchscreen device that provides a "simple, faster, and more intuitive" method for scrolling a screen display. Narutaka at [0006]. When the touch panel detects that a user's finger has touched the screen, it checks whether the duration of the touch is longer than "a predetermined fixed amount of time." *Id*. at [0018]. If the touch is shorter than this fixed amount, the system interprets the touch as an input rather than a scroll. *Id*. at [0019]. But if the touch lasts "for the fixed amount of time or longer," the system "determines that the operator has given a scroll instruction." *Id*. at [0020]. In this case, when "the finger [] of the operator is lifted off" the screen, the system calculates a movement on the basis of the finger's starting and ending coordinates (*id*. at [0022]-[0023]) and scrolls the screen to the finger's end coordinates (*id*. at [0024], Fig. 2).

94. If the system determines that a finger is moving but has not yet lifted off the screen, the entire display screen moves with the finger until the finger lifts off. *Id*. at [0032]-[0036].

4. Westerman

95. I understand that Westerman is Exhibit 1007 to the Petitions.

96. Westerman describes techniques for tracking multiple finger contacts across a multi-touch surface, such as a touchscreen display. Westerman, Abstract ("Apparatus and methods are disclosed for simultaneously tracking multiple finger (202-204) and palm (206, 207) contacts as hands approach, touch, and slide across

a proximity-sensing, compliant, and flexible multi-touch surface (2).”); *id.*, claims 14 (“The multi-touch surface apparatus of claim 7 being one of fabricated on or integrated with a display device.”) and 15 (“The multi-touch surface apparatus of claim 14, wherein the display device comprises one of a liquid crystal display (LCD) or a light-emitting polymer display (LPD).”).

97. Westerman recognizes that various methods for manual input of commands were available, such as rollers and trackballs that “excel at panning and scrolling,” but that there was a need to integrate the functionality of all these devices into a single input device. *Id*. at 1. This was because users would not pay for a “multitude of input devices” and would not tolerate switching between peripherals to perform different functions. *Id*. To address these needs, Westerman’s system integrated different types of manual input into a single multi-touch surface. *Id*. at 8. In one application, Westerman’s system allowed a user to scroll a document with a multi-finger slide gesture and stop the scrolling with another touch or when the document hit its end. *Id*. at 72.

5. Astala

98. I understand that Astala is included as Exhibit 1008 to the Petitions.

99. Astala describes a technique for inputting data using “a touch screen of [an] electronic device” that accepts finger inputs. Astala, Abstract. Astala’s device detects the location of the finger touch and the duration between when the fin-

ger first touches the screen and when the finger stops touching the screen. *Id*. The system determines the "particular function of the electronic device" based on the detected touch location and time duration. *Id*. The device interprets a touch input as a "long click" if it lasts longer than a defined time threshold and as a "short click" if it is shorter than this threshold. *Id*. at 8:64-9:8. The system interprets these "long click" and "short click" functions as indicating different instructions from the user. *Id*. at 9:12-20. Astala also discloses that "three or more different time periods of touching may be used to detect different intended input functions." *Id*. at 9:27-29.

IX. UNPATENTABILITY OF THE '387 PATENT CLAIMS

100. Based upon my experience in the fields of user interface design and computer science, my review of the '387 patent and claims, as well as other materials cited herein and attached as exhibits to the Petitions, it is my opinion that challenged claims 1-12 of the '387 patent would have been obvious to a POSITA at the time of the earliest possible priority date of the '387 patent (December 28, 2001) in view the combinations of Anwar, Korhonen, Narutaka, Westerman, and/or Astala discussed below. More specifically, it is my opinion that claims 1-12 of the '387 patent are unpatentable as follows:

Challenged Claims	Unpatentable As Obvious Over
1, 5, 6, 7, 9	Anwar and Narutaka

Challenged Claims	Unpatentable As Obvious Over
1, 5, 6, 7, 9	Anwar, Narutaka, and Westerman
2, 3, 8, 11, 12	Anwar, Narutaka, and Astala
2, 3, 8, 11, 12	Anwar, Narutaka, Westerman, and Astala
4, 10	Anwar, Narutaka, and Korhonen
4, 10	Anwar, Narutaka, Westerman, and Korhonen
1, 4, 5-7, 9, 10	Korhonen and Narutaka
1, 4, 5-7, 9, 10	Korhonen, Narutaka, and Westerman
2, 3, 8, 11, 12	Korhonen, Narutaka, and Astala
2, 3, 8, 11, 12	Korhonen, Narutaka, Westerman, and Astala

101. A detailed discussion of my opinions regarding the unpatentability of claims 1-12 follows. While I discuss specific portions of the prior art references and ’387 patent in this Declaration to exemplify my analysis, I am prepared to use any or all of these references and the ’387 patent to support my opinions.

A. Claims 1, 5, 6, 7, and 9 are Obvious Over the Combinations of Anwar and Narutaka and Anwar, Narutaka, and Westerman.

102. It is my opinion that claims 1, 5, 6, 7, and 9 of the ’387 patent would have been obvious to a POSITA in view of the combination of Anwar and Narutaka and Anwar, Narutaka, and Westerman. The following is a summary of the reasons why one designing a system or implementing methods according to Anwar would have been motivated to incorporate the teachings of Narutaka and Westerman.

103. Each of Anwar, Narutaka, and Westerman share the same fundamental goal: providing natural and intuitive user interfaces for a touchscreen system. Anwar states that "the systems and methods described herein provide a graphical user interface that exhibits a touch and feel user interface experience." Anwar at 2:6-8; *see also id*. at 14:11-12 ("the user interface … present[s] a more natural way of moving documents through a viewing space"). Narutaka discloses "screen scrolling using a screen display that is simpler, faster and more intuitive." Narutaka, [0006]. Westerman states that "[i]t is another object of the present invention to provide an improved method for invoking cursor motion continuation only when the user wants it." Westerman at 9. I therefore understand these references to be in the same field of endeavor. As I explained in paragraphs 49-56, by the time of the '387 and '064 patents, there was a well-understood body of concepts, techniques, and previous implementations focusing on natural and easy to use interfaces. This included identification of common interaction tasks—such as selecting items and navigating through information too large to fit in available display space—and common interaction techniques—such as "clicking", "tapping", "scrolling", "zooming", etc. It also was common to combine interaction techniques and even interaction devices in any given implementation, such as a mouse as well as a built-in trackpad or trackball. *E.g.*, App. I at 7:11-13 ("the input device 106 may be a keyboard, mouse, trackpad, trackball, or any combination there-

of."); App. J at 5:14-15 (disclosing a compact computer with "capability for both keyboard input and stylus input"). Thus, a POSITA faced with a problem relating to improving user interfaces for touchscreen systems would have been motivated to look to combine relevant teachings to create natural and easy to use ways to select objects and navigate through information, and teachings of Anwar, Narutaka, and Westerman all offer useful and closely related such teachings.

104. Anwar, Narutaka, and Westerman also address their goal of improving touchscreen user interfaces in the same way: by using touch gestures to differentiate between different intended user instructions. *E.g.*, Anwar at 9:46-49 ("[touch] movements … may be passed to an interface process … to detect a motion representative of a known command"); Narutaka, [0019]-[0020] (detecting a "touch input" operation for a short touch, and a scroll for a long touch); Westermans at 72 (describing simulated "mouse clicks … generated by a tap of a fingertip pair," "double-clicks … by a single tap of three fingertips," and "[w]indow scrolling … allocated to slides of four fingers"). As I have noted, a shared understanding of fundamental interaction tasks meant that two user interfaces that both enabled selecting, pointing, and scrolling would have been seen by a POSITA as close relatives in the family of interactive systems, no matter what particular gestures (or even interface devices) were used to implement the tasks. A "select" is still a "select", whether instantiated as using a mouse to move a cursor then clicking the

mouse, or as a tap by one's finger. Therefore, given the common goals and overlapping interaction tasks of Anwar, Narutaka, and Westerman, a POSITA would have been motivated to combine their teachings in the manner described.

105. Further, Anwar, Narutaka, and Westerman all disclose systems with similar structure and features that would be complementary. Anwar discloses handheld computing devices with a touch sensitive display that a user employs to interact with documents. *See* Anwar, Abstract. Narutaka discloses a touch screen system that allows a user to scroll through displayed data. Narutaka, [0008], [0010]. Westerman discloses a touch screen or other touch-sensitive surface that senses and classifies gestures by touches from multiple fingers. Westerman at Abstract, claims 14-15. Because each of these inventions share common goals and overlap in the interactions they support, I further believe that where they are complementary, a POSITA would have been motivated to and would have seen the benefit of incorporating gesture interpretation techniques disclosed in any of these references into any of the others.

106. Additionally, the claims of the '387 patent recite a combination of familiar elements that are taught by Anwar, Narutaka, and Westerman. The claims generally recite sensing the speed, direction, and duration of a finger touch; scrolling or dragging content in response to this sensing; and decaying and stopping scrolling upon a touch or end signal. All of this functionality was well known

years before the ’387 patent was filed. All of this functionality was well known years before the ’387 patent was filed. *See generally* Section VI(B). As discussed in preceding paragraphs, the claims do not present this known functionality in any new or unusual way, nor do they indicate that it achieved surprising results. Instead, each of these references discloses performing scrolling and duration and direction-sensing functionality in a predictable and standard manner.

107. For these reasons and the reasons discussed below, a POSITA would have found it obvious to combine the relevant aspects of Anwar, Narutaka, and Westerman in the manner discussed below.

1. Independent Claim 1

108. Claim 1 of the ’387 patent reads as follows (with labeling added in brackets for ease of discussion):

> 1[pre]. An improved touch-screen image scrolling system, comprising:
>
> 1[a] an electronic image display screen;
>
> 1[b] a microprocessor coupled to said display screen to display information thereon and to receive interactive signals therefrom;
>
> 1[c] timer means associated with said microprocessor to provide timing capacity therefor;
>
> 1[d] a source of scroll format data capable of display on said display screen;
>
> 1[e] a keyboard coupled to said microprocessor to provide input control signals thereto;

> 1[f] finger touch program instructions associated with said microprocessor for sensing the speed, direction and time duration of a finger touch contact with said display screen;
>
> 1[g] scrolling motion program instructions associated with said microprocessor responsive to said duration of said finger touch contact such that, when said duration exceeds a first given preset minimum time and is accompanied by motion along the surface of said screen followed by separation of said finger touch from said screen, a scroll format display on said screen is caused to begin to scroll in said sensed direction and at said sensed initial speed;
>
> 1[h] time decay program instructions associated with said microprocessor for reducing the rate of scrolling displacement on said display screen at a given rate until motion is terminated;
>
> 1[i] stopping motion program instructions associated with said microprocessor for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (b) an end-of-scroll signal received from said scroll format data source.

Below, I explain the bases for my opinion that claim 1 of the ’387 patent would have been obvious in view of Anwar and Narutaka, and Anwar, Narutaka, and Westerman.

109. The preamble of claim 1 reads: “An improved touch-screen image scrolling system, comprising: ….” I am informed that the preamble of a claim may not be limiting. Even so, I will show that the preamble of claim 1 of the ’387 patent is disclosed by Anwar.

110. Anwar discloses systems for displaying and manipulating documents. Anwar at 5:48-50 ("The systems and methods described herein include systems and methods for manipulating and viewing documents displayed on a viewing surface"). These systems and methods can be implemented on handheld touchscreen devices. *Id*. at 5:53-56 ("the systems and methods will be described with reference to certain exemplary embodiments, including hand held computer systems that include touch screen displays"). The touchscreen systems may allow users to scroll displayed documents. "For example, the system 10 may be required to generate a zoomed view of part of a document, and then to pan or scroll the zoomed view to display adjacent portions of the document." *Id*. at 7:39-42. Therefore, I understand Anwar to disclose "an improved touch-screen image scrolling system."

111. Element 1[a] recites "<u>an electronic image display screen.</u>" Anwar discloses that the computer system includes a screen that displays an electronic image. Anwar at 5:54-56 (describing "exemplary embodiments, including hand held computer systems that include touch screen displays"). Anwar's Figure 1, shown below, depicts "a video display 26" that "can present the images of a plurality of different documents." *Id*. at 5:63, 6:5-10. Therefore, I understand Anwar to disclose "an electronic image display screen."

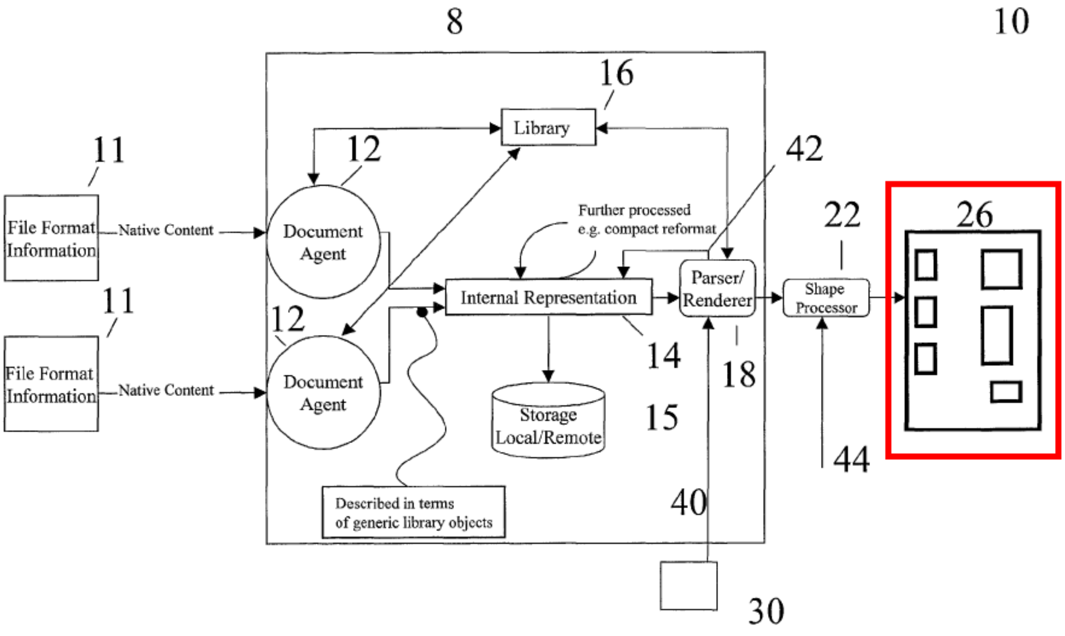


Figure 1

112. Element 1[b] recites "a microprocessor coupled to said display screen to display information thereon and to receive interactive signals therefrom." Anwar's computer system is "a computer device of the type that commonly includes a processor, a memory and a display." Anwar at 5:63-66. Anwar discloses that the disclosed techniques may be implemented in code that is executed on a processor. *Id*. at 5:66-6:2 ("the system 10 may also be realized, in whole or in part as a software system comprising system code capable of executing on a processor to configure the processor as a system according to the invention"). Anwar also discloses that the processor can be a microprocessor. *Id*. at 15:12-17 ("in an embodiment where the platform is primarily a microprocessor, microcontrollers or DSPs, the user interface systems can be realized as a computer program written in

microcode or written in a high level language and compiled down to microcode that can be executed on the platform employed").

113. Anwar describes this processor-executed code as the "computer process 8" in Figure 1. *Id*. at 6:2-3. Figure 1 shows computer process 8 as coupled to the display screen. Anwar also discloses that computer process 8 displays information on the screen. "The computer process 8 generates a single output display that includes … one or more of the documents," which are "displayed within the program window generated by computer program 8." *Id*. at 6:19-24; *id*. at 6:29-31 ("display 26 presents content representative of different data types in a single, integrated display"). The "program window generated by computer program 8" also enables the process to receive interactive signals from the user that are input through the display. *Id*. at 6:24-28 ("The program window for the computer process 8 may also include a set of icons representative of tools provided with the graphical user interface and capable of allowing a user to control the operation, in this case the display, of the documents appearing in the program window."). Therefore, I understand Anwar to disclose "a microprocessor coupled to said display screen to display information thereon and to receive interactive signals therefrom."

114. Element 1[c] recites "timer means associated with said microprocessor to provide timing capacity therefor." As discussed in Section VII(C), I believe

the broadest reasonable interpretation of “timing means” is a timer that provides timing capacity for a microprocessor. Anwar’s computer system uses time measurements to make various calculations. Anwar notes that the system can measure “the amount of time that has passed since [a] document was last viewed” (Anwar at 10:20-21) and incorporate this time difference into visual elements of the display. *Id*. at 10:17-23 (“The measure of distance from the center document may be representative of the distance in pages from the document 44, or may be representative of some other measure of distance or difference, such as the amount of time that has passed since the document was last viewed, the difference in alphabetical order, or some other characteristic.”). Anwar also discloses that the system tracks “the time at which the graphical interface tool acts on the rendered content” to determine “[t]he contextual relationship between the graphical interface tool and the rendered content.” *Id*. at 3:62-66.

115. The fact that Anwar’s system is able to perform time-based operations such as measuring the amount of time that has passed since a particular event (e.g., the time since a document was last viewed) and capturing the actual time of an event (e.g., when the graphical interface tool interacts with content) necessarily indicates that the system includes a timer. A POSITA would have understood that ultimately, all operations—including these time-based operations—in a computer are performed by a timer associated with its processing unit, or microprocessor.

Therefore, a POSITA would have understood Anwar’s disclosures of a microprocessor and time-based calculations to necessarily disclose a timer. The ’387 patent acknowledges this, by disclosing that timers were “inherent” in computers when that patent was filed. ’387 patent at 5:31-35. I have explained this further in paragraph 79.

116. A POSITA would also have considered it obvious to integrate a timer into Anwar’s system to provide the microprocessor with timing capacity, to the extent the system did not already have one. Timers and counters have been an essential component of embedded systems for decades. I understand that both Philips and the district court have recognized as much in this case. Ex. 1022 at 13; Ex. 1023 at 5, n.9. Timers provide a microprocessor with basic functionality such as measuring elapsed time (by counting processor clock ticks or with a real time clock) and timing external events. As of 2001, effectively all microprocessors had a basic on-chip timer.

117. A POSITA would have seen the value in incorporating this type of well-known timer into Anwar’s microprocessor in order to accomplish the time-based measurements disclosed in Anwar. Practically speaking, a timer would have been necessary to perform any of those operations—there would have been no way to perform those functions without a timer. Therefore, I understand Anwar to necessarily disclose “timer means associated with said microprocessor to provide tim-

ing capacity therefore" and I also understand that a POSITA would have found it obvious to incorporate a standard timer into Anwar's touch screen system as of the filing date of the '387 patent, based on the knowledge in the art.

118. Element 1[d] recites "a source of scroll format data capable of display on said display screen." Anwar discloses that the display screen can "present the images of a plurality of different documents." Anwar at 6:9-10. These can include any number of different types of data: documents, streaming media, web pages, anything that can be processed and displayed. *Id*. at 6:12-18 ("each of the depicted documents can be associated with one separate application program, such as Word, Netscape Navigator, Real Player, Adobe, Visio and other types of applications. It will be understood that the term document as used herein will encompass documents, streamed video, web pages, and any other form of data that can be processed and displayed by the computer process 8.").

119. Anwar's system translates source documents into an "internal representation of the document" that includes the content itself and information about the page layout. *Id*. at 7:57-65 ("In one practice the created digital representation includes information that describes the page layout of the document, including information about page size, margins and other page layout information. The digital representation also includes information about the content of the source document, such as the text, figures, and other content information that appears in the docu-

ment.”). This internal representation is stored in memory as a “content document file.” *Id.* at 3:42-45 (“These devices may comprise a processor, memory, and a touch-sensitive display; a content document file stored in the memory and being representative of an internal representation of the content”). Figure 1 of Anwar shows this “Internal Representation” that is stored in “Local/Remote [Storage]”:

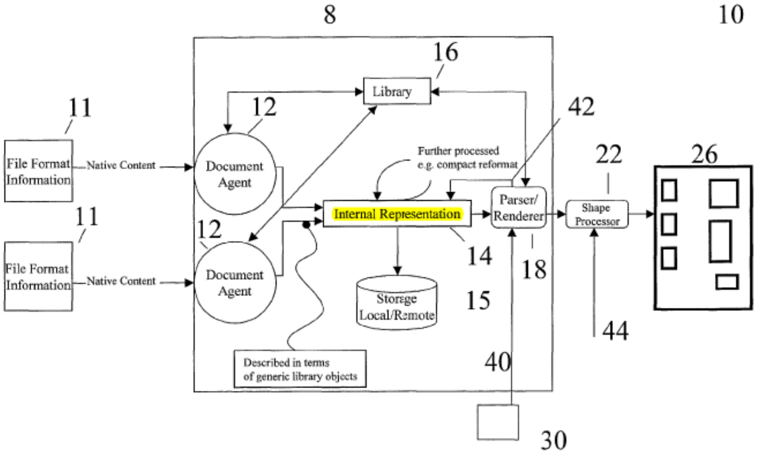


Figure 1

120. Figure 1 shows that there is a feedback path 42 between the parser/renderer 18 component of the computer process 8. The stored internal representation 14 “trigger[s] an update of the content of the internal representation 14.” *Id.* at 7:48-51. The stored internal representation 14 is passed to the parser/renderer 18 element, which generates “a ‘view’ of the documents represented by the internal representation 14.” *Id.* at 7:32-34. The parser/renderer receives inputs that define context and time-based parameters for this view, including what parts of the content are to be displayed and when/for how long. “which parts of the

internal representation are required for a particular view and how, when and for how long the view is to be displayed." *Id*. at 7:36-39 ("The parser/renderer 18 receives view control inputs which define the viewing context and any related temporal parameters of the specific document view which is to be generated."); id. at 7:42-46 ("The view control inputs are interpreted by the parser/renderer 18 to determine which parts of the internal representation are required for a particular view and how, when and for how long the view is to be displayed."). This stored data can be displayed and then scrolled. "For example, the system 10 may be required to generate a zoomed view of part of a document, and then to pan or scroll the zoomed view to display adjacent portions of the document." *Id*. at 7:39-42. Therefore, I understand Anwar to disclose "a source of scroll format data capable of display on said display screen."

121. Element 1[e] recites "<u>a keyboard coupled to said microprocessor to provide input control signals thereto</u>." Anwar states that its techniques "may be employed in other applications including applications wherein content is displayed on a conventional computer workstation that includes typical input tools such as a standard keyboard and a mouse." Anwar at 4:10-13. "In those applications where the user is provided a keyboard … the user may employ that particular input device for selecting which of the documents within the array of documents that the user would like to appear within the viewing area." *Id*. at 10:45-49; *id*. at 15:53-56

("the systems described herein may be practiced with any suitable interface devices, including … keyboards"). Therefore, I understand Anwar to disclose "a keyboard coupled to said microprocessor to provide input control signals thereto."

122. Element 1[f] recites "finger touch program instructions associated with said microprocessor for sensing the speed, direction and time duration of a finger touch contact with said display screen." Anwar's discloses "a command stroke that may be employed by a user … to cause page movement of [a] document within the viewing area" in connection with Figures 12A-13B. Anwar at 14:3-6. A "velocity detector process takes position readings periodically," and "[f]rom these position readings a page velocity determination may be made." *Id.* at 14:7-9. The parser/renderer component of the computer program "employ[s] the velocity determination … to redraw the document in a series of pictures that will portray the document as moving across the screen," thus "present[ing] a more natural way of moving documents through a viewing space." *Id.* at 14:10-15; *id.* at claim 1 ("the engine pans the displayed document … at a rate based on the determined velocity vector"). I understand Anwar's system to include touch program instructions for "sensing the speed" of a "touch contact with said display screen."

123. Anwar discloses that the "user may drag a document at a certain speed and then release the stylus, mouse, or other input device from the document" and that the page "may continue to move *in the established direction* until the user in-

dicates that the document is to stop moving[.]" *Id*. at 14:16-19 (emphasis added). By stating that the document continues to scroll in "the *established* direction," I understand Anwar to be talking about the direction in which the user is dragging the document at the certain speed referenced immediately above. Thus, I understand Anwar's system to necessarily include touch program instructions for "sensing the … direction" of a touch contact with the screen. Therefore, I understand Anwar to disclose "program instructions associated with said microprocessor for sensing the speed [and] direction" of a "touch contact with said display screen."

124. A POSITA would have found it obvious to use a finger touch with Anwar's system. Anwar already discloses that the computer system may be used with a broad range of input devices. *Id*. at 9:39-42 ("display 26 may include a screen monitoring process for monitoring the screen of the display 26 to detect movement of a cursor, stylus or some other pointer across the images of the documents presented on the screen"); *id*. at 15:52-59 ("systems described … may be practiced with any suitable interface devices, including touch-sensitive screens and pads, … and any other suitable devices"). Touch screen systems that accepted finger touch inputs were common and well-known as of the priority date, and accepting finger touch input would have allowed Anwar's mobile station to provide the "touch and feel user interface experience" that it expressly aimed to provide. Anwar at 2:6-8, Abstract.

125. Even further, Anwar provides user functionality that is effectively identical to a finger touch, through simulating "tactile" control and detecting users' "brushing" motions. *Id*. at 2:9-11 ("the systems and methods described herein include hand-held or mobile computer devices having a system for simulating tactile control over a document"); *id*. at 9:61-64 ("novel graphical user interface tools that allow a user to manipulate and view a document on a display and to simulate tactile control over the depicted documents"); *id*. at 14:38-43 (if "the screen monitor, detect[s] a brushing motion across the surface of the document 44," it directs the parser/renderer to "flip" a page); *id*., claims 1, 3 ("[a] computer device having a system for simulating tactile control over a document"). A POSITA would therefore have considered it obvious to use a finger touch input as one of the "other pointers" Anwar refers to.

126. Additionally, as I have explained above, it was well understood that many different input techniques and devices could be used to support the same fundamental interaction tasks. The fundamental tasks included *pointing*, *selecting, dragging*, with a mouse, stylus, touchpad, trackball, and touch all well-known interface technologies to support these tasks A number of earlier patents and articles discuss touchscreen systems that accepted finger touch inputs. *See*, *e.g*., Resman at 1:13-15 (noting that "[t]ouch-pad display screens have found wide application in Personal Digital Assistants" and describing finger touch); Minsky at 196 (system

"tracks the motion of a finger … on the screen"); Amro, Abstract (system with "a user interactive touch screen" controlled by a "touch finger"); Asami at 1 (touch panel display system moved image at speed of a finger). In fact, one of the references cited by the Patent Office during prosecution of Anwar, U.S. Patent No. 6,310,610 to Beaton, discloses finger-touch and stylus capabilities that the Patent Office determined met Anwar's "pointer" limitations. *See* Ex. 1024 at 6:40-44 ("use of a finger may, for example, invoke tools or dialogues that are finger-touchable and large whereas the use of a sharp stylus may invoke a modified GUI with smaller touch targets."). A POSITA would have been very familiar with finger touch functionality as of the priority date.

127. A POSITA would also have found it trivial to incorporate finger touch functionality into Anwar's touchscreen: the technology for sensing a finger touch was only minimally different from that for sensing a stylus. Anwar's touchscreen would have been either a resistive or a capacitive touchscreen, and both types would have been readily able to sense input from a finger or a stylus alike. Resistive touchscreens were made up of sheets of resistive material separated by an air gap or insulating layer. When the sheets were pushed together at some location in response to a touch, connectors on the sheets detected that touch location. Because a resistive touchscreens detected only the location at which the sheets were pressed together, nearly any type of object—specifically including a finger or a stylus—

could serve as the touch input device. Capacitive touchscreens used an insulator panel (such as glass) with a conductive coating for the screen, and simply sensed the location of any distortion in the screen's electrostatic field. Both fingers and capacitive styli work in the same way with respect to a capacitive touchscreen: both distort the screen's electrostatic field at the location of the touch, and both are therefore sensed in the same way. For applications such as Anwar that did not require precision or minute movements, finger and stylus input would have been entirely interchangeable for either resistive or capacitive touchscreens. Therefore, since (1) both finger and stylus input enable the same interaction tasks--including selecting and scrolling, (2) both finger and stylus input may be enabled by the same underlying technologies, and (3) many systems have supported both, a POSITA naturally would look to combine techniques and teachings developed in the context of one of these modalities to the other. This would have been well within the skill of a POSITA, and it would have amounted to nothing more than using well-known elements according to their known functions, with unsurprising results.

128. Further, a POSITA would have seen the value in allowing a user to operate Anwar's computer system with a finger, rather than forcing the user to keep track of a separate stylus or other input device. Finger touch has always been the most natural input mechanism for a touchscreen. It also provides increased

simplicity and flexibility for users, who no longer need to worry about keeping track of a stylus or other peripheral device. It was widely recognized that using a stylus was inconvenient as of the priority date. For example, U.S. Patent No. 6,073,036 to Heikkenen (Ex. 1025) states that stylus operation may "challenge the coordination skills of the user, especially … where the motion of [a] vehicle makes it difficult to control the relative position of the stylus with respect to the touch screen" and that "use of a stylus may be objectionable to some users, and in some situations may present a safety issue. *Id*. at 1:62-2:2. Most often, stylus usage was reserved for applications that required the high degree of precision that only a stylus could provide. *Id.* at 1:41-54 ("the small amount of display area allocated to each symbol, legend, and/or related function on a touch sensitive screen … has forced the use of a stylus, instead of the more natural finger"). A POSITA would have understood that Anwar's application, which displayed a document, did not need this high degree of precision, and therefore would have understood finger touch functionality to be the more natural and convenient option for the user.

129. Additionally, Narutaka discloses accepting "finger touch" inputs. Narutaka discloses techniques for controlling a touchscreen device "whereby screen scrolling using a screen display that is simpler, faster, and more intuitive is realized." Narutaka, [0006]. The device control system "is operated by an operator touching a screen 8 on the CRT 1 *using a finger* 7 (see FIG. 3) and the screen 8

displayed on the CRT 1 is scrolled by the CPU 5 in accordance with the direction and amount of movement of the finger 7." *Id*., [0010].

130. A POSITA would have seen the value in incorporating Narutaka's disclosure of using a finger touch as input. Again, offering a user the option of using his finger would have increased simplicity and flexibility, allowing the user to choose any input method that was most convenient. Using a finger touch would have furthered Anwar's goal of "allow[ing] a user to more easily manipulate and view content" on a handheld device via a "graphical user interface that exhibits a touch and feel user interface experience." Anwar at 2:4-8. There is no closer approximation to a "touch and feel" experience than manipulating content directly, using a finger. *E.g*., Ex. 1012 at 2 ("Pointing at an item or touching it, is one of the most natural ways to select it. Touchscreens allow the software designer to take advantage of this convenient selection method by having the users simply touch the item they are interested in."). Further, a POSITA would have found it trivial to incorporate finger-touch inputs into Anwar's system for the reasons listed above, in connection with Anwar's own disclosure. Indeed, Narutaka even recognized that it was "common" for touchscreen systems to accept input from a finger *and* a pen/stylus. Narutaka, [0002]). Myriad other references also make this observation. *E.g*., Resman at 1:13-15; Minsky at 196; Amro, Abstract; Asami at 1. Therefore, a POSITA would have found it obvious to incorporate finger touch into Anwar's

touch screen system as of the filing date of the ’387 patent, based on the knowledge in the art as well as Narutaka.

131. Element 1[f] also recites “program instructions” for sensing the “time duration of a finger touch contact with said display screen.” Anwar’s system includes a “velocity detector process [that] takes [touch] position readings periodically, such as every centi-second.” Anwar at 14:7-8. “From these position readings a page velocity determination may be made.” *Id*. at 14:8-9. This velocity detector then allows the system to “portray the document as moving across the screen” when a user “drag[s] a document at a certain speed.” *Id*. at 14:12-17. Velocity, or speed, is defined as the distance per unit time, and is typically measured in units of distance per second (*e.g*., m/s or ft/s). Because Anwar’s system can calculate the velocity of a touch input, Anwar’s system necessarily includes program instructions for “sensing the … time duration” of a touch contact with the display screen—this ability is necessary to performing the velocity calculation.

132. Additionally, Narutaka discloses instructions for sensing the time duration of a finger touch contact. Narutaka discloses that “[w]hen the touch panel 2 detects that the screen … has been touched by the finger 7 of the operator … the CPU 5 imports the touch location data and checks whether or not the touch location data is continuously output by the interface circuit 3 for a predetermined fixed amount of time.” Narutaka, [0018]. If the finger touch lasts less than this fixed

amount of time, Narutaka's CPU interprets the instruction as an input. *Id.*, [0019] ("If the touch location data is not output for the fixed amount of time by the interface circuit 3, the CPU 5 determines that the operator has input an instruction other than scroll input, and performs a touch input process in accordance with an icon or the like at the coordinates indicated by that touch location data."). If the finger touch lasts "for the fixed amount of time or longer," however, the CPU interprets the input as a scroll instruction. *Id.*, [0020].

133. A POSITA would have found it obvious to incorporate Narutaka's teaching of sensing the duration of a user's touch into Anwar's system. Anwar discloses "a more natural way of moving documents through a viewing space." Anwar at 14:11-12. A POSITA would have understood there to be a benefit in adding the ability to sense the duration of a user's touch, in that it furthers Anwar's stated goal by allowing the system to discern the user's intended operation. For example, this would allow the system to distinguish between touches intended to be input and those intended to be a scroll command. Narutaka, [0019]-[0020]. Further, Narutaka's system allows scrolling to be "simpler, faster, and more intuitive" (*id.*, [0006]), which supports Anwar's goal of a displaying scrolled or panned document in a "more natural" manner.

134. A POSITA would also have found it straightforward to incorporate the ability to sense duration of a touch into Anwar's system. In fact, as discussed

above, a POSITA would have understood Anwar's system to *already* sense duration of a touch given that Anwar's system includes a "velocity detector." This duration-sensing ability would have been provided by standard timer and counter functionality contained in Anwar's microprocessor. Adding the functionality of sensing touch duration to Anwar's system would have been nothing more than the use of a familiar feature according to its expected function, and it would have yielded entirely predictable results: the timer, operating in its standard fashion, would have measured the length of a finger touch. Therefore, I understand Anwar to necessarily disclose "program instructions" for sensing the "time duration of a finger touch contact with said display screen" and I also understand that a POSITA would have found it obvious to combine Anwar with Narutaka for this limitation as of the filing date of the '387 patent by incorporating Narutaka's disclosure of such instructions into Anwar's touch screen system.

135. Element 1[g] recites "scrolling motion program instructions associated with said microprocessor responsive to said duration of said finger touch contact such that, when said duration exceeds a first given preset minimum time and is accompanied by motion along the surface of said screen followed by separation of said finger touch from said screen, a scroll format display on said screen is caused to begin to scroll in said sensed direction and at said sensed initial speed." Anwar discloses that when the user's touch moves across the surface of the screen and

then lifts off, the displayed document scrolls in the direction and at the speed of the touch. "A process may employ the velocity determination to direct the parser/render 18 to redraw the [displayed] document in a series of pictures that will portray the document as moving across the screen. For example, a user may drag a document at a certain speed and then release the stylus, mouse or other input device from the document. … [T]he page may continue to move in the established direction until the user indicates that the document is to stop moving such as clicking on the document." Anwar at 14:12-22.

136. For documents with multiple pages, the system scrolls through the different pages at a rate determined by the touch speed. *Id*. at 14:22-26 ("For multi page documents the velocity measure may be used for panning different pages of the document across the screen at a rate determined by the page velocity set when the user drags one page of the document across the screen."); *id*. at claim 1 ("in response to the command detected … being the pan command the engine pans the displayed document on the display at the rate based on the determined velocity vector"). Anwar discloses that its system moves the document "more naturally" by mimicking the user's speed. *Id*. at 14:10-12 ("The page velocity determination may be employed for allowing the user interface to present a more natural way of moving documents through a viewing space."). Therefore, I understand Anwar to disclose "scrolling motion program instructions associated with said microproces-

sor … such that, when … [there is] motion along the surface of said screen followed by separation of said finger touch from said screen, a scroll format display on said screen is caused to begin to scroll in said sensed direction and at said sensed initial speed."

137. Element 1[g] also recites "program instructions … responsive to said duration of said finger touch contact such that, when said duration exceeds a first given preset minimum time … a scroll format display on said screen is caused to begin to scroll." Narutaka discloses that when the touch screen detects a finger touch, "the CPU 5 … checks whether or not the touch location data is continuously output by the interface circuit 3 for a predetermined fixed amount of time." Narutaka, [0018]. If the finger touch lasts "for the fixed amount of time or longer," the CPU interprets the input as a scroll instruction. *Id.*, [0020]. The CPU then scrolls the displayed image to the end location at which the touch leaves the screen:

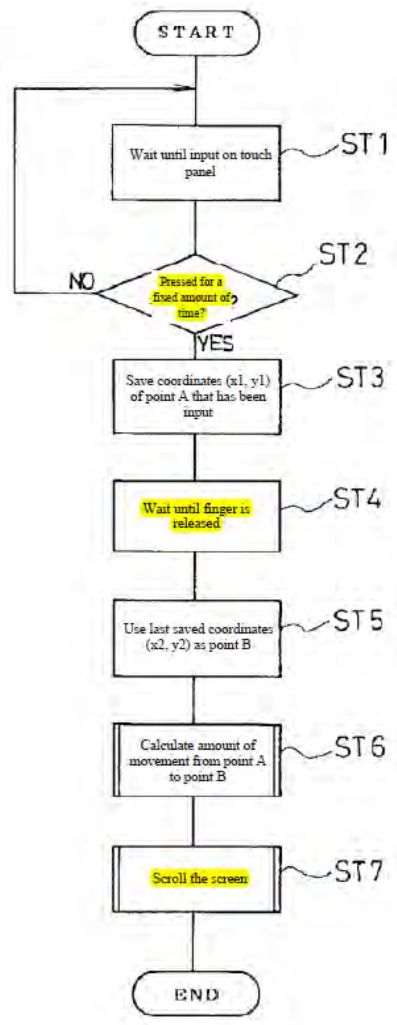
> If the finger 7 of the operator that is touching the CRT 1 is moved without moving off of the screen 8, the CPU 5 reads and stores every change in the values of the touch location data output by the touch panel 2 as candidate ending coordinates (X3, Y3).
>
> The touch panel 2 then detects when the finger 7 of the operator is lifted off of the CRT 1, after which no touch location data is output by the interface circuit 3. The CPU 5 then stores the last candidate ending coordinates (X3, Y3) as official ending coordinates (X2, Y2) (step ST5).

> The CPU 5 then calculates a movement amount (ΔX, ΔY) by making the computation indicated by the following equations on the basis of the starting coordinates (X1, Y1) and the ending coordinates (X2, Y2) as shown in FIG. 4 (step ST6).

Id. at [0021]-[0023].

138. Figure 2 illustrates this process (shown with highlights added):

FIG. 2


START
Wait until input on touch panel
ST1
NO
Pressed for a fixed amount of time?
ST2
YES
Save coordinates (x1, y1) of point A that has been input
ST3
Wait until finger is released
ST4
Use last saved coordinates (x2, y2) as point B
ST5
Calculate amount of movement from point A to point B
ST6
Scroll the screen
ST7
END

139. A POSITA would have seen the value of incorporating Narutaka's teaching of scrolling only upon a touch contact lasting a minimum amount of time into Anwar's system. This would have allowed Anwar's system to more accurately determine user intent. As of the priority date, a short touch (sometimes called a "click") was commonly used to indicate *selection* or input. *Scrolling* is a distinct interaction task, so a POSITA would have considered it obvious to wait for some small period of time after detecting a touch before starting to scroll, in order to confirm that the user actually intended to scroll and not merely to select an item (or perform some other operation). A POSITA also would have recognized that Anwar's goal of using "touch movements" to identify and differentiate between "known command[s]" (Anwar at 9:46-49) would be furthered by making sure that user intent was correctly identified.

140. Further, requiring a time delay before starting to scroll (as taught by Narutaka) would have been a routine design choice and one of limited number of known solutions for distinguishing between different types of user touch gestures in a touch screen system. A user interface designer had two main touch-related variables to work with in crafting touch-sensitive system: the nature (*e.g.*, cadence and shape) of a touch and the duration of the touch. A POSITA would have been able to fit Narutaka's time delay functionality naturally into Anwar's system with minimal effort, simply by encoding if statements (or similar checks) verifying

touch duration before selecting an appropriate operation. Therefore, I understand that a POSITA would have found it obvious to combine Anwar with Narutaka for this limitation as of the filing date of the ’387 patent by incorporating Narutaka’s disclosure of beginning scrolling when the touch duration exceeds a minimum time into Anwar’s touch screen system.

141. Element 1[h] discloses “time decay program instructions associated with said microprocessor for reducing the rate of scrolling displacement on said display screen at a given rate until motion is terminated.” Anwar discloses that after the user stops touching the screen, the document’s scroll velocity decreases at “a constant page inertia”—which I understand to mean at a particular rate—until the document stops moving. Anwar at 14:26-28 (“the velocity may decrease by a constant page inertia until it reaches zero velocity and page scrolling ceases”); *id.* at claim 4 (“A computing device … wherein the rate at which the engine renders the series of pages of the document decreases over time based on the page inertia.”). Therefore, I understand Anwar to disclose “time decay program instructions associated with said microprocessor for reducing the rate of scrolling displacement on said display screen at a given rate until motion is terminated.”

142. Element 1[i] recites “stopping motion program instructions associated with said microprocessor for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising:

(a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (b) an end-of-scroll signal received from said scroll format data source." Anwar discloses that the system stops scrolling the document when it detects that the user has clicked on the screen. "[T]he page may continue to move in the established direction until the user indicates that the document is to stop moving such as clicking on the document." Anwar at 14:18-22. A POSITA would have understood Anwar's disclosure of a "click" to indicate a "substantially stationary" touch. As of 2001, "click" was generally used to indicate a quick press-and-release input at a fixed location, without moving the input device. *E.g.*, App. K ("click … the act of pressing and releasing a mouse button without moving the mouse pointer"); App. L ("To press and quickly release a mouse button."); Westerman at 3 ("Touch screens … often distinguish pointing motions from emulated button clicks … by assuming very little lateral fingertip motion will occur during taps on the touch surface which are intended as clicks.").

143. Additionally, the fact that Anwar's system can detect that the user has clicked on a document at all necessarily indicates that the touch "endure[d] for a period longer than a preset minimum time." A touch must last longer than the rate at which the system scans the touch screen for inputs if it is sensed by the system. For example, a common scan rate for smart phone touchscreens as of 2001 (as well as today) is 60Hz, or 60 scans per second. This means a scan occurs every 16.67

milliseconds, so if a touch were to take place after one scan and end before the next scan, it would not be detected. I understand that Philips has asserted, consistently, that the scan rate of a touch screen satisfies the recited "preset minimum time." *See* Ex. 1026 at 7 ("a first given preset minimum time (e.g., a scan rate of the touch screen)"), 9 ("when a finger touches the screen for more than a first given preset minimum (e.g., a scan rate of the touch screen)"). Therefore, I understand Anwar to disclose "stopping motion program instructions associated with said microprocessor for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time."

144. Even further, a POSITA would have found it obvious to wait for a stationary touch lasting longer than a preset minimum time before terminating the scroll. Requiring a stationary touch of some time duration would have allowed Anwar's system to distinguish between touches intended to stop the scrolling and a mistaken touch or a gesture that indicated a different operation, such as a faster scroll. A POSITA would have seen the value in requiring this stationary touch to ensure that the system identified the user's desired action correctly before acting on it. A POSITA would also have found it straightforward to add instructions to the system code for sensing whether the user's touch lasted longer than a specified

time period and whether the location of the touch varied during that time period. These could have been implemented as simple if statements (or analogous conditional statements) checking for a certain time duration before performing a stop-scroll or other operation.

145. Element 1[i] also recites instructions for terminating scrolling displacement of the image upon “an end-of-scroll signal received from said scroll format data source.” The ’387 patent states that this “end of scroll signal” can be an indication that the end of the document has been reached. ’387 patent at 1:60-65 (“After the finger separates from the screen, the image continues to move in the same direction at a gradually decreasing speed until motion is stopped manually by touching the screen without movement of the finger, … or until the image reaches its ‘end’.”). A POSITA would have found it obvious to stop the scrolling in Anwar’s system when the end of the document is reached if that happened before a finger touch. This would have been the expected, and certainly most natural, operation, given Anwar’s context: it scrolls pages of a document initially at the speed of a user’s touch, and the pages then slow down at an inertial rate. Anwar at 14:12-18, 14:26-28. Anwar’s purpose behind this inertial scrolling is to “present a more natural way of moving documents through a viewing space.” *Id*. at 14:10-12. It would have been natural, and thus supportive of Anwar’s goal, for the document to stop scrolling when it hit the end. It would also have been trivial to ensure that

Anwar did this. Anwar's system already allows the document to simply stop moving upon release (Anwar at 14:18-19); a similar flag could be set to detect when the parser/renderer hits the end of file signal in the internal representation file. A user would have found it strange if Anwar's system caused the document to loop back to the beginning or scroll through a series of blank pages, and this would have undermined Anwar's goal of natural movement. Therefore, a POSITA would have found it obvious to incorporate stopping at an end of scroll signal into Anwar's touch screen system as of the filing date of the '387 patent, based on the knowledge in the art, such that the system stopped scrolling upon either the end of scroll signal or a stationary finger touch of the appropriate time.

146. Additionally, Westerman discloses this limitation. In Westerman, the display enters a scrolling "slide mode" when "significant motion is detected" in multiple fingers touching a touch surface. Westerman at 68-69. When the system detects liftoff by all the moving fingers, it enters "motion continuation mode" in which the "scrolling velocity" continues at the pre-liftoff average of the fingers' velocity. *Id*. at 72. "Motion continuation mode does not stop until any of the remaining fingers not in the synchronized subset are lifted or more fingers touch down." *Id*. Additionally, "the host computer can sent a signal instructing motion continuation mode to be canceled if the cursor reaches the edge of the screen or end of a document." *Id*. Thus, Westerman discloses program instructions for ter-

minating scrolling upon the first occurrence of either (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time (when "fingers touch down" for longer than the touch screen's scan rate) and (b) an end-of-scroll signal received from said scroll format data source (the signal from the host computer indicating the end of the screen or document).

147. A POSITA would have found it obvious to incorporate Westerman's teaching of terminating scrolling upon either an end of scroll signal *or* a stationary finger touch, for similar reasons to those discussed above: stopping at the end of the document would have been most natural to a user, and would have furthered Anwar's operational and design goals. A POSITA would also have found it straightforward to include this functionality into Anwar. Therefore, a POSITA would also have found it obvious to incorporate stopping at an end of scroll signal into Anwar's touch screen system as of the filing date of the '387 patent, based on Westerman.

148. For these reasons, it is my opinion that a POSITA would have found claim 1 obvious over the combination of Anwar with Narutaka, and separately over the combination of Anwar with Narutaka and Westerman.

2. Dependent Claim 5

149. Claim 5 recites "[t]he improved touch-screen image scrolling system of claim 1, wherein said microprocessor, and said timer means together comprise a

processing unit of a conventional computer." Anwar's computer system includes a standard processor that executes the code stored in the system's memory. Anwar at 5:63-66 ("The system 10 is shown as a functional block diagram of a computer device of the type that commonly includes a processor, a memory and a display."); *id*. at 2:13-17 ("These systems may include a housing which supports a processor, memory, and a touch-sensitive display … , [and] system code stored within the memory and adapted to be executed by the processor."); *id*., claim 1 ("computer device having a system for simulating tactile control over a document, comprising a processor, memory, and a touch-sensitive display"). A POSITA would have understood this disclosure of a conventional processor typically included in a computer device processor to include the microprocessor and timer components discussed in connection with element 1[c]. Therefore, I understand Anwar to disclose the system of claim 1, "wherein said microprocessor, and said timer means together comprise a processing unit of a conventional computer."

3. Dependent Claim 6

150. Claim 6 recites "[t]he improved touch-screen image scrolling system of claim 5, wherein said source of scroll format data capable of display on said display screen comprises part of the memory of said conventional computer." Anwar discloses that the internal representation of the document is "stored in the memory" of the computing device. Anwar at 3:43-45 ("a content document file

stored in the memory and being representative of an internal representation of the content"). Figure 1 (below) depicts the internal representation as being stored in "Storage Local/Remote." Therefore, I understand Anwar to disclose the system of claim 1, "wherein said source of scroll format data capable of display on said display screen comprises part of the memory of said conventional computer."

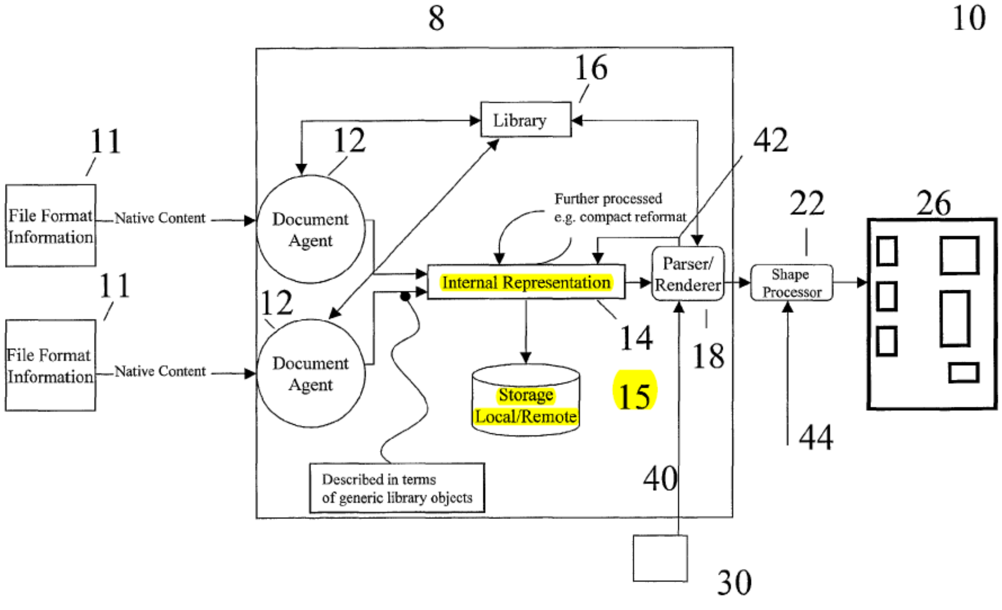


Figure 1

4. Independent Claim 7

151. The limitations of claim 7 are nearly identical to those of claim 1. The only substantive change is that claim 7 recites a "computer apparatus" instead of a "microprocessor." A POSITA would have considered a "computer apparatus" that could perform all of the recited limitations to include a microprocessor. At a

minimum, a POSITA would have understood a "computer apparatus" to be broader than a "microprocessor."

152. Claim 7 of the '387 patent reads as follows (with labeling added in brackets for ease of discussion) (*see also* App. A):

> 7[pre]. An improved touch-screen image scrolling system, comprising:
>
> 7[a] an electronic image display screen;
>
> 7[b] a computer apparatus coupled to said display screen to display information thereon and to receive interactive signals therefrom;
>
> 7[c] timer means within said computer apparatus to provide timing capacity therefor;
>
> 7[d] said computer apparatus having capacity to store scroll format data capable of display on said display screen;
>
> 7[e] a keyboard coupled to said computer apparatus to provide input control signals thereto;
>
> 7[f] finger touch program instructions associated with said computer apparatus for sensing the speed, direction and time duration of a finger touch contact with said display screen;
>
> 7[g] scrolling motion program instructions associated with said computer apparatus responsive to said duration of said finger touch contact such that, when said duration exceeds a preset minimum time and is accompanied by motion along the surface of said screen, a scroll format display on said screen is caused to begin to scroll in the sensed direction and at the sensed initial speed;

> 7[h] time decay program instructions associated with said computer apparatus for reducing the rate of scrolling displacement on said display screen at a given rate until motion is terminated;
>
> 7[i] stopping motion program instructions associated with said computer apparatus for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (b) an end-of-scroll signal received from said scroll format data source.

153. The preamble of claim 7 is identical to the preamble of claim 1. I am informed that the preamble of a claim may not be limiting. Even so, for the reasons I have discussed in connection with the preamble of claim 1, the preamble of claim 7 is disclosed by Anwar for the same reasons discussed in connection with element 1[pre].

154. Element 7[a] is identical to element 1[a]. Anwar discloses this limitation for the same reasons discussed in connection with element 1[a].

155. Element 7[b] is similar to element 1[b], with the caveat that 7[b] recites a "computer apparatus coupled to said display screen" while 1[a] recites a "microprocessor." A POSITA would have understood Anwar to disclose this limitation for the same reasons discussed in connection with element 1[b].

156. Element 7[c] is similar to element 1[c], except that 7[c] also recites a "computer apparatus" while 1[a] recites a "microprocessor." A POSITA would

have understood Anwar to disclose this limitation and have rendered it obvious for the same reasons discussed in connection with element 1[c].

157. Element 7[d] is similar to element 1[d]. A POSITA would have understood Anwar to disclose this limitation for the same reasons discussed in connection with element 1[d].

158. Element 7[e] is similar to element 1[e], except that 7[e] recites a "computer apparatus" while 1[e] recites a "microprocessor." A POSITA would have understood Anwar to disclose this limitation for the same reasons discussed in connection with element 1[e].

159. Element 7[f] is similar to element 1[f], except that 7[f] recites a "computer apparatus" while 1[f] recites a "microprocessor." A POSITA would have understood Anwar to have rendered this limitation obvious, alone or in combination with Narutaka, for the same reasons discussed in connection with element 1[f].

160. Element 7[g] is similar to element 1[g], except that 7[g] recites a "computer apparatus" while 1[g] recites a "microprocessor" and that element 7[g] is broader in that it does not require a touch contact to be "followed by separation of said finger touch from said screen" (as element 1[g] requires). A POSITA would have understood Anwar to have rendered this limitation obvious in combi-

nation with Narutaka for the same reasons discussed in connection with element 1[g].

161. Element 7[h] is similar to element 1[h], except that 7[h] recites a "computer apparatus" while 1[h] recites a "microprocessor." A POSITA would have understood Anwar to have rendered this limitation obvious, alone or in combination with Narutaka, for the same reasons discussed in connection with element 1[h].

162. Element 7[i] is similar to element 1[i], except that 7[i] recites a "computer apparatus" while 1[i] recites a "microprocessor." A POSITA would have understood Anwar to have rendered this limitation obvious, alone or in combination with Westerman, for the same reasons discussed in connection with element 1[i].

163. In sum, Anwar in combination with Narutaka and Anwar in combination with Narutaka and Westerman renders claim 7 obvious.

5. Independent Claim 9

164. The limitations of claim 9 are also nearly identical to those of claim 1. Claim 9 recites a "method of controlling the scroll-like display of data on an electronic display screen" instead of a "touch-screen image scrolling system," and is generally the method counterpart to claim 1.

165. Claim 9 of the '387 patent reads as follows (with labeling added in brackets for ease of discussion) (*see also* App. A):

> 9[pre]. An improved method of controlling the scroll-like display of data on an electronic display screen, said method comprising the steps of:
>
> 9[a] sensing the duration of finger touch contact time with an electronic display screen having scrollable data displayed thereon;
>
> 9[b] sensing the speed and direction of motion of said finger touch contact with said display screen;
>
> 9[c] initiating scrolling motion of said scrollable data on said display screen in said sensed direction and at said sensed speed;
>
> 9[d] slowing the speed of said scrolling motion from the initiated speed thereof, at a predetermined rate;
>
> 9[e] and terminating said scrolling motion when one of the conditions comprising the following group of conditions is sensed: (a) a substantially stationary finger touch having a finite duration is sensed; (b) an end-of-scroll signal is sensed.

166. The preamble of claim 9 ("[a]n improved method of controlling the scroll-like display of data on an electronic display screen") is similar to the preamble of claim 1 and element 1[a] ("[a]n improved touch-screen image scrolling system, comprising an electronic image display screen"). I am informed that the preamble of a claim may not be limiting. Even so, for the reasons I have discussed in

connection with the preamble of claim 1 and element 1[a], the preamble of claim 7 is disclosed by Anwar.

167. Element 9[a] ("sensing the duration of finger touch contact time with an electronic display screen having scrollable data displayed thereon") is similar to element 1[f] ("finger touch program instructions … for sensing the … time duration of a finger touch contact with said display screen"). A POSITA would have understood Anwar to have rendered this limitation obvious, alone or in combination with Narutaka, for the same reasons discussed in connection with element 1[f].

168. Element 9[b] ("sensing the speed and direction of motion of said finger touch contact with said display screen") is similar to element 1[f] ("finger touch program instructions … for sensing the speed [and] direction … of a finger touch contact with said display screen"). A POSITA would have understood Anwar to have rendered this limitation obvious, alone or in combination with Narutaka, for the same reasons discussed in connection with element 1[f].

169. Element 9[c] ("initiating scrolling motion of said scrollable data on said display screen in said sensed direction and at said sensed speed") is similar to element 1[g] ("scrolling motion program instructions … such that … a scroll format display on said screen is caused to begin to scroll in said sensed direction and at said sensed initial speed"). A POSITA would have understood Anwar to have

rendered this limitation obvious in combination with Narutaka for the same reasons discussed in connection with element 1[g].

170. Element 9[d] ("slowing the speed of said scrolling motion from the initiated speed thereof, at a predetermined rate") is similar to element 1[h] ("time decay program instructions … for reducing the rate of scrolling displacement on said display screen at a given rate"). A POSITA would have understood Anwar to have rendered this limitation obvious, alone or in combination with Narutaka, for the same reasons discussed in connection with element 1[h].

171. Element 9[e] ("terminating said scrolling motion when one of the conditions comprising the following group of conditions is sensed: (a) a substantially stationary finger touch having a finite duration is sensed; (b) an end-of-scroll signal is sensed") is similar to element 1[i] ("stopping motion program instructions … for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (b) an end-of-scroll signal received from said scroll format data source"). A POSITA would have understood Anwar to have rendered this limitation obvious, alone or in combination with Westerman, for the same reasons discussed in connection with element 1[i].

172. In sum, Anwar in combination with Narutaka and Anwar in combination with Narutaka and Westerman renders claim 9 obvious.

B. Claims 2, 3, 8, 11, and 12 are Obvious Over the Combinations of Anwar, Narutaka, and Astala and Anwar, Narutaka, Westerman, and Astala.

173. It is my opinion that claims 2, 3, 8, 11, and 12 of the ’387 patent would have been obvious to a POSITA in view of the combination of Anwar, Narutaka, and Astala, and Anwar, Narutaka, Westerman, and Astala.

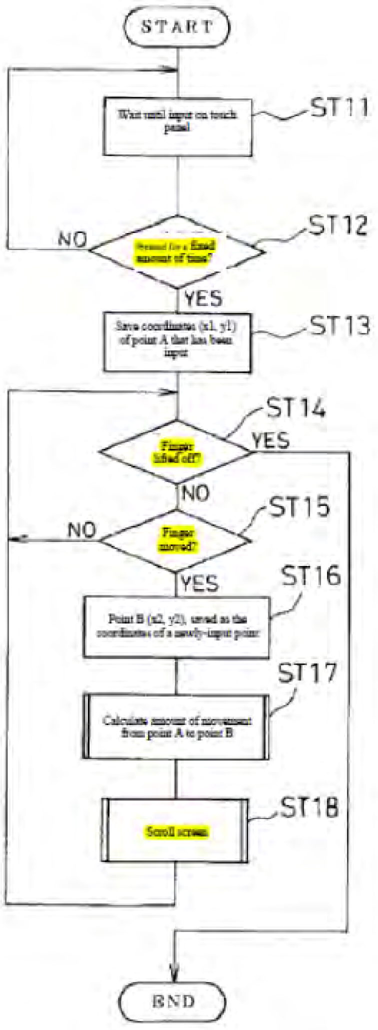
1. Dependent Claim 2

174. Claim 2 recites “[t]he improved touch-screen image scrolling system of claim 1, wherein said scrolling motion program instructions further comprise instructions to move said display in correspondence with movement of the finger touch, in response to movement following a touch having a stationary duration greater than said first preset given minimum time and less than a second given preset minimum time.” In simpler terms, claim 2 recites instructions to move the display in correspondence with a touch that lasts longer than a first time threshold (the “first given preset minimum time” from claim 1[g] that triggers scrolling), but is shorter than a second time threshold (“a second given preset minimum time”).

175. Narutaka discloses instructions for moving the display in correspondence with movement of a finger touch, in response to movement following a touch having a stationary duration greater than the first preset given minimum time.

Narutaka discloses a screen that is scrolled in correspondence with a finger touch. Narutaka, [0010] ("screen 8 … is scrolled by the CPU 5 in accordance with the direction and amount of movement of the finger"). Narutaka discloses one embodiment where the display is moved in correspondence with the finger touch. "[I]t is also possible to continuously scroll the screen 8 … when the finger 7 of the operator touches the screen 8 … and is moved over the screen 8 without being lifted off the screen 8." *Id.*, [0026]. If a finger touch lasts "for a predetermined fixed amount of time" (*id.*, [0029]), the system "determines that the operator has given a scroll instruction" (*id.*, [0031]) and the screen is moved in line with the finger movement (*id.*, [0032]-[0035]). Figure 5 illustrates this process of scrolling along with the finger if the finger is first pressed for a fixed amount of time:

FIG. 1


START
Wait until input on touch panel
ST11
NO
Pressed for a fixed amount of time?
ST12
YES
Save coordinates (x1, y1) of point A that has been input
ST13
ST14
YES
Finger lifted off?
NO
ST15
NO
Finger moved?
YES
ST16
Point B (x2, y2), saved as the coordinates of a newly-input point
ST17
Calculate amount of movement from point A to point B
ST18
Scroll screen
END

176. A POSITA would have understood Narutaka's disclosure of a finger being "pressed for a fixed amount of time" to refer to a touch having a "stationary duration" greater than that fixed amount of time. Narutaka states that there are singular coordinates for the "press" that lasts for the fixed amount of time, indicating that it is a stationary touch. "If the touch location data is output for the fixed

amount of time or longer by the interface circuit 3, however, the CPU 5 determines that the operator has given a scroll instruction and stores the touch location data as starting coordinates (X1, Y1)." Narutaka, [0031]. Also, the term "press" is typically used to refer to a motionless touch in user interface applications. In contrast, words like "drag," "fling," or "flick" are generally used to refer to moving touches.

177. Additionall, a POSITA would have found it obvious to modify Narutaka by determining whether the finger touch had been stationary for a "fixed amount of time" before scrolling the image. The POSITA would have understood that requiring some period of fixed position would reduce the risk of the system guessing incorrectly that the user wanted to perform a one-to-one scroll, when in fact the user wanted to perform some other function (e.g., a fling scroll or cursor move). Adding this requirement would again have been trivial and well within the POSITA's skill set, requiring nothing more than a conditional statement that performed scrolling only if the touch satisfied a durational requirement.

178. A POSITA would have found it obvious to incorporate the functionality of this embodiment of Narutaka—that of moving the screen while the finger maintains contact with the screen—into the combined Anwar/Narutaka and Anwar/Narutaka/Westerman systems. A POSITA would have understood that the two embodiments of Narutaka provided a user with complementary and non-redundant features. The Figure 2 operation (*see* element 1[g] discussion above)

would have allowed a user to scroll repeatedly with each finger liftoff, which could be useful to quickly scroll or to scroll through substantial content. Figure 5's operation would have allowed the display to lock to the user's finger movement, which could be useful for detailed or fine-tuned review. Given that Anwar is focused on a system that allows a user to interface with documents, a POSITA would have seen the benefit to providing a user with as many document review tools as possible, and would have understood combining the Figure 2 and Figure 5 embodiments as a step towards that goal. Many users would even have expected a document interface system to include both scroll functionalities, since both were well-known on desktop computer systems in existence as of 2001.

179. A POSITA would also have considered this a trivial design choice that could be easily implemented using, for example, a designated touch duration or sequence of clicks to differentiate between intended operations. The large overlap between Figures 2 and 5—which have a series of identical steps, ST1-ST3 in Figure 2 and ST11-ST13 in Figure 5—further indicates that combining this functionality would have been as straightforward, requiring nothing more than changing Figure 5 to make the "Yes" decision branch at ST14 connect to step ST5 of Figure 2. Therefore, I understand that a POSITA would have found it obvious to combine the embodiments of Narutaka's Figures 2 and 5 such that the display moved in correspondence with movement of the finger touch, in response to

movement following a touch having a stationary duration greater than a first preset given minimum time.

180. Claim 2 further recites that the display is moved with the finger touch if the touch lasts "less than a second given preset minimum time." Astala discloses "detect[ing] different intended input functions" depending on "different time periods of touching." Astala at 9:27-29. For example, if "the total time of the touch input, Ttouch, was greater than a threshold value, LC, for a long click," the system detects that "a long click has been input." *Id*. at 8:65-9:2. "If Ttouch was less than or equal to LC, however, a short click … has been input." *Id*. at 9:5-8. The system may interpret clicks of different durations to signify different intended functions. *Id*. at 9:15-18 ("a short click selecting an object for a drag or move function, while a long click selects the object for opening or activation"). For example, "three or more different time periods of touching may be used to detect different intended input functions." *Id*. at 9:27-29. Astala discloses that its system can perform different actions by using different predefined time durations to detect different intended user inputs. "[A] first [decision] path [is followed] … where the detected time period is less than a first predetermined value, [and] a second path [is followed] … where the detected time period is equal to or greater than the first predetermined value and less than or equal to a second predetermined value." *Id*. at

9:27-36. Thus, Astala discloses a “a second given preset minimum time” that is used to indicate a particular function.

181. A POSITA would have found it obvious to incorporate Astala’s teaching of using “a second given preset minimum time” to distinguish between intended user inputs in the Anwar/Narutaka system, such that the system moves the screen with the finger only if the initial stationary touch is between the first and second time durations. As of 2001, it was common to use the duration of a touch to select one of multiple user operations. Both Narutaka and Astala recognize this. *E.g.*, Narutaka, [0019]-[0020] (interpreting “an instruction … [for] touch input” if the touch is shorter than some duration, and “a scroll instruction” for longer touches); Astala at 9:27-29 (“different time periods of touching may be used to detect different intended input functions”). A POSITA would have understood this well-known technique of recognizing different durations of touch input to carry the benefit of allowing the system to differentiate between multiple user-desired operations. Due to the small size of the display screen on handheld devices such as those disclosed by Anwar, a POSITA would have been motivated to move as much as possible of the functionality previously provided by larger peripherals—the keyboard and mouse, for example—into the touchscreen, in the form of recognized touch gestures. Both Anwar and Astala recognize this motivation to make it easier to manipulate content on a small touchscreen device, which lacks space for keys

and other input controls outside of the screen. *E.g.*, Anwar at 1:61-64 ("[T]here is a need … [to] provide improved user interface tools that make it more facile to manipulate and view content presented by a handheld or portable device."); Astala at 2:14-18 ("The keypad of the device … [is] deleted and their functions implemented by the touch screen display screen, thereby allowing more space to be utilized for the display screen."). In fact, given that Narutaka already uses finger liftoff to distinguish between operations (for example, scrolling upon finger liftoff, while moving the screen with the finger if liftoff has not occurred), a POSITA would been led to use a *different* criterion to further differentiate between operations. The familiar criterion of touch duration would have provided a ready option.

182. Additionally, a POSITA would have understood Astala to describe a system with similar structure and operation as those of Anwar, Narutaka, and Westerman. Like those systems, Astala relates to "electronic devices and more particularly to a touch screen input technique for allowing a user input to an electronic device having a touch screen." Astala at 1:19-22. Also similarly, Astala provides users with the ability to more conveniently view and manipulate documents. *Id.* at 8:46-9:11, Figs. 6a-d. Even further, Astala's goal is consistent with Anwar, Narutaka, and Westerman, in that it aims to allow a user "to enter commands and information into the communications device in an efficient manner" and uses duration and location of touch inputs to interpret user inputs as a means to

"allow[] more space to be utilized for the display screen." *Id.* at 1:30-33, 2:16-29. Astala is therefore similar in structure, operation, and overall field of endeavor to Anwar, Narutaka, and Westerman, and these overlapping features would have further led a POSITA to consider it obvious to incorporate Astala's teaching of using different touch time durations to indicate different intended inputs into the systems discussed for claim 1.

183. For these reasons, it is my opinion that a POSITA would have found claim 2 obvious over the combination of Astala with Anwar and Narutaka, and separately over the combination of Astala with Anwar, Narutaka, and Westerman.

2. Dependent Claim 3

184. Claim 3 recites "[t]he improved touch-screen image scrolling system of claim 1, wherein said scrolling motion program instructions further comprise instructions to move a touch-selected item relative to the stationary display in correspondence with movement of said finger touch, in response to motion following a touch having a stationary duration greater than said second given preset minimum time." Anwar discloses that items can be "dragged" across the screen in correspondence with a touch. One example Anwar describes is displayed graphical tools, such as a magnifier or query tool, that are dragged onto and across the screen. Anwar at 9:35-38 ("the graphical tool 50 may be moved over the screen… dragging a stylus or some other pointer across the screen of the display"); *id.*, Figure

7A ("Click and drag from magnifier tool button, crates and drags magnifier onto/across screen"); *id*. at 11:54-56 ("The query tool S4 when activated, either by dragging an image of the query tool 55 onto the document"). Anwar does not disclose that moving touch selected items is done "in response to motion following a touch having a stationary duration greater than said second given preset minimum time."

185. As discussed for claim 2, however, Astala discloses that specific touch time durations can be used to identify different intended user inputs. Astala at 9:27-29 ("three or more different time periods of touching may be used to detect different intended input functions"). Astala also discloses that different touch time durations can be used to distinguish drag operations like those disclosed by Anwar from other operations. "For example, the short click-long click function could be used on objects to signify selection of the object according to other functions. One example of this would be a short click selecting an object for a drag or move function, while a long click selects the object for opening or activation." *Id*. at 9:13-18.

186. A POSITA would have found it obvious to use duration, as taught by Astala, to indicate that a particular user touch was meant to trigger a "drag" operation in Anwar's system (movement of a touch-selected item in correspondence with movement of the finger touch). As discussed for claim 2, it was well-known as of 2001 to use the duration of a touch as a differentiating factor to identify

which operation a user meant to perform. A POSITA would have understood that by requiring an extended touch before initiating a drag, the system could have avoided "guessing" the user's intent incorrectly. For example, the system could avoid initiating a drag of an object when the user intended to move the entire display with the finger, if the system required the user's touch to last longer than some duration (selected to be long enough that it could be assumed that the user meant to indicate the object drag). A POSITA would have recognized that incorporating this technique for intuiting user intent would have provided the benefit of offering a user "improved user interface tools that make it more facile to manipulate and view content," which is what Anwar tried to do. Anwar at 1:61-64.

3. Independent Claim 8

187. Independent claim 8 recites limitations that are similar to those of independent claims 1, 7, and 9. The majority of its language is a nearly verbatim replica of claim 1. *See* Appendix A. In addition to those limitations also recited by claim 1, claim 8 recites limitations similar to those of claim 2: instructions for "mov[ing] said display in correspondence with movement of the finger touch, in response to motion following a touch having a stationary duration greater than said first given preset minimum time and less than [a] second given preset minimum time." *Id.* The limitations of claim 8 are therefore obvious for the reasons discussed for the corresponding limitations of claims 1 and 2.

188. Claim 8 of the ’387 patent reads as follows (with labeling added in brackets for ease of discussion) (*see also* App. A):

> 8[pre]. An improved touch-screen image scrolling system, comprising:
>
> 8[a] an electronic image display screen;
>
> 8[b] a microprocessor coupled to said display screen to display information thereon and to receive interactive signals therefrom;
>
> 8[c] timer means associated with said microprocessor to provide timing capacity therefor;
>
> 8[d] a source of scroll format data capable of display on said display screen;
>
> 8[e] a keyboard coupled to said microprocessor to provide input control signals thereto;
>
> 8[f] finger touch program instructions associated with said microprocessor for sensing the speed, direction and time duration of a finger touch contact with said display screen;
>
> 8[g] scrolling motion program instructions associated with said microprocessor responsive to said duration of said finger touch contact such that, when said duration exceeds a first given preset minimum time, and is less than a second given preset minimum that is greater than said first minimum, and is accompanied by motion along the surface of said screen, a scroll format display on said screen is caused to begin to scroll in the sensed direction and at the sensed initial speed;
>
> 8[h] said scrolling motion program instructions still further comprising instructions to move said display in correspondence with movement of the finger touch, in re-

> sponse to motion following a touch having a stationary duration greater than said first given preset minimum time and less than said second given preset minimum time;
>
> 8[i] time decay program instructions associated with said microprocessor for reducing the rate of scrolling displacement on said display screen at a given rate until motion is terminated;
>
> 8[j] stopping motion program instructions associated with said microprocessor for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (b) an end-of-scroll signal received from said scroll format data source.

189. The preamble of claim 8 is identical to the preamble of claim 1. I am informed that the preamble of a claim may not be limiting. Even so, for the reasons I have discussed in connection with the preamble of claim 1, the preamble of claim 8 is disclosed by Anwar for the same reasons discussed in connection with element 1[pre].

190. Element 8[a] is identical to element 1[a]. Anwar discloses this limitation for the same reasons discussed in connection with element 1[a].

191. Element 8[b] is identical to element 1[b]. A POSITA would have understood Anwar to disclose this limitation for the same reasons discussed in connection with element 1[b].

192. Element 8[c] is identical to element 1[c]. A POSITA would have understood Anwar to disclose this limitation and have rendered it obvious for the same reasons discussed in connection with element 1[c].

193. Element 8[d] is identical to element 1[d]. A POSITA would have understood Anwar to disclose this limitation for the same reasons discussed in connection with element 1[d].

194. Element 8[e] is identical to element 1[e]. A POSITA would have understood Anwar to disclose this limitation for the same reasons discussed in connection with element 1[e].

195. Element 8[f] is identical to element 1[f]. A POSITA would have understood Anwar to have rendered this limitation obvious, alone or in combination with Narutaka, for the same reasons discussed in connection with element 1[f].

196. Element 8[g] is similar to element 1[g], except that 8[g] recites the additional element of a duration of a touch that causes scrolling to start "is less than a second given preset minimum that is greater than said first minimum." This is substantively identical to a limitation of claim 2 (which provides that this first touch duration is "greater than said first preset given minimum time and less than a second given preset minimum time"). A POSITA would have understood Anwar to have rendered element 8[g] obvious in combination with Narutaka and Astala for the same reasons discussed in connection with element 1[g] and claim 2.

197. Element 8[h] ("said scrolling motion program instructions still further comprising instructions to move said display in correspondence with movement of the finger touch, in response to motion following a touch having a stationary duration greater than said first given preset minimum time and less than said second given preset minimum time") is identical to claim 2 ("wherein said scrolling motion program instructions further comprise instructions to move said display in correspondence with movement of the finger touch, in response to movement following a touch having a stationary duration greater than said first preset given minimum time and less than a second given preset minimum time"). A POSITA would have understood Anwar to have rendered element 8[h] obvious in combination with Narutaka and Astala for the same reasons discussed in connection with claim 2.

198. Element 8[i] is identical to element 1[h]. A POSITA would have understood Anwar to have rendered this limitation obvious, alone or in combination with Narutaka, for the same reasons discussed in connection with element 1[h].

199. Element 8[j] is similar to element 1[i]. A POSITA would have understood Anwar to have rendered this limitation obvious, alone or in combination with Westerman, for the same reasons discussed in connection with element 1[i].

200. In sum, Anwar in combination with Narutaka and Astala and Anwar in combination with Narutaka, Westerman, and Astala renders claim 8 obvious.

4. Dependent Claim 11

201. Claim 11 is a method claim that depends from claim 9. Claim 11 recites limitations that are generally analogous to those of claim 2: claim 11 recites the method of claim 9 that "comprises the further step of sensing a finger touch on said screen having a duration greater than said first given preset minimum time and less than a second given preset minimum time which is greater than said first given time and then moving said display in correspondence with movement of the finger touch," which is substantively similar to claim 2 (the system of claim 1 "further compris[ing] instructions to move said display in correspondence with movement of the finger touch, in response to movement following a touch having a stationary duration greater than said first preset given minimum time and less than a second given preset minimum time"). Claim 11 is obvious over the Anwar/Narutaka/Astala and Anwar/Narutaka/Wester-man/Astala combinations for the reasons discussed for claims 9 and 2.

5. Dependent Claim 12

202. Claim 12 is a method claim that depends from claim 9. Claim 12 recites the method of claim 9 that "comprises the further step of sensing a stationary finger touch on said screen having a duration greater than a second preset given minimum time which is greater than said first given preset time and then moving a touch-selected item relative to the stationary display in correspondence with

movement of the finger touch," which is substantively similar to claim 3 (the system of claim 1 "further compris[ing] instructions to move a touch-selected item relative to the stationary display in correspondence with movement of said finger touch, in response to motion following a touch having a stationary duration greater than said second given preset minimum time"). Claim 12 is obvious over the Anwar/Narutaka/Astala and Anwar/Narutaka/Westerman/Astala combinations for the reasons discussed for claims 9 and 3. *Id*.

C. Claims 4 and 10 are Obvious Over Anwar and Narutaka and Anwar, Narutaka, and Westerman in Further Combination with Korhonen.

203. It is my opinion that claims 4 and 10 of the '387 patent would have been obvious to a POSITA in view of the combination of Anwar, Narutaka, and Korhonen, and Anwar, Narutaka, Westerman, and Korhonen.

1. Dependent Claim 4

204. Claim 4 recites "[t]he improved touch-screen image scrolling system of claim 1, wherein said group of signals for terminating scrolling, displacement of the image on said display screen further comprises (a) a signal indicating that the rate of scrolling displacement on said screen has decayed to a value below a predetermined given value." Anwar discloses that "the velocity [of the scrolling document] may decrease by a constant page inertia until it reaches zero velocity and page scrolling ceases." Anwar at 14:26-28. Korhonen discloses that "scrolling is

stopped if the scrolling speed falls below a predetermined limit." Korhonen 2:35-37; *id*., claim 6 (same).

205. A POSITA would have found it obvious to incorporate this teaching from Korhonen, of stopping scrolling when the already-slowing rate has decreased below a predetermined value, into the combined Anwar/Narutaka and Anwar/Narutaka/Westerman systems discussed for claim 1. There would have been an apparent value in incorporating this teaching. If Anwar's "constant page inertia" was implemented as an exponential decay function—which is calculated as a constant rate over a period of time—the velocity of the scroll would never exactly reach zero. However, in mathematical terms, it would approach arbitrarily ("asymptotically") close to zero, and in any real computer, due to the limits of computer arithmetic, it eventually would be so close to zero, that the computer would treat it as zero. Setting a sufficiently low velocity at which scrolling stops would have provided a straightforward way to avoid this technicality in a manner that the naked eye could not detect. This would also have created a more natural and intuitive user experience, as Korhonen recognized: the cutoff speed can be set low enough that scrolling "is completely stopped only when in practice it appears to have stopped" to the user. Korhonen at 2:39-41. Setting the cutoff point at a low-value threshold rather than zero would also have been a routine design choice for a

POSITA and trivial to implement. In the code, the only change would be to check for a velocity less than some minimum number, rather than 0.

206. Further, a POSITA would have understood Korhonen's system to be similar in structure and operation to those of Anwar, Narutaka, and Westerman. Like those systems, Korhonen relates to a mobile touchscreen system and aims to "present[] a method [for presenting information on the touchscreen display] which is faster, easier, and clearer." Korhonen at 1:43-46, 1:54-57. Korhonen's goal is also similar those of Anwar, Narutaka, and Westerman: it seeks to improve "the speed and the ease with which [a displayed] list is managed, as well as the clarity of presentation," given the restrictions imposed by a small screen size and fewer input options. *Id*. at 4:1-3, 1:15-35. This similarity in structure, operation, and overall field of endeavor would have further led a POSITA to consider it obvious to incorporate Korhonen's teaching of terminating scrolling if the scrolling speed falls below a predetermined limit.

2. Dependent Claim 10

207. Claim 10 is a method claim that depends from claim 7. Claim 10 recites limitations that are generally analogous to those of claim 4: claim 10 recites the method of claim 7 "wherein said group of conditions to be sensed for terminating said scrolling motion further comprises: the speed of said scrolling motion on said screen slows to a value below a predetermined given value," which is substan-

tively similar to claim 4. Claim 10 is obvious over the Anwar/Narutaka/Korhonen and Anwar/Narutaka/Westerman/Kohonen combinations for the reasons discussed for claim 4.

D. Claims 1, 4, 5, 6, 7, 9, and 10 Are Obvious Over Korhonen, Korhonen and Narutaka, Korhonen and Westerman, and Korhonen, Narutaka, and Westerman.

208. It is my opinion that claims 1, 5, 6, 7, and 9 of the ’387 patent would have been obvious to a POSITA in view of Korhonen and the combinations of Korhonen and Narutaka, Korhonen and Westerman, and Korhonen, Narutaka, and Westerman. The following is a summary of the reasons why one designing a system or implementing methods according to Korhonen would have been motivated to incorporate the teachings of Narutaka and Westerman. As I explained in, *e.g.*, paragraphs 49-56 and 104, by the time of the relevant priority date there was a well-understood body of concepts, techniques, and previous implementations focusing on natural and easy to use interfaces. This included identification of common interaction tasks—such as selecting items and navigating through information too large to fit in available display space—and common interaction techniques—such as “clicking”, “tapping”, “scrolling”, “zooming”, etc. It also was common to combine interaction techniques and even interaction devices in any given implementation. *E.g.*, App. I at 7:11-13 (“the input device 106 may be a keyboard, mouse, trackpad, trackball, or any combination thereof.”); App. J at

5:14-15 (disclosing a compact computer with "capability for both keyboard input and stylus input"). Thus, a POSITA faced with a problem relating to improving user interfaces for touchscreen systems would have been motivated to look to combine relevant teachings to create natural and easy to use ways to select objects and navigate through information, and teachings of Korhonen, Narutaka, and Westerman all offer useful and closely related teachings to this effect.

209. A POSITA would have found it obvious to combine these references in the specific manner described below and would have understood this combination to produce a benefit. Each of Korhonen, Narutaka, and Westerman share the basic goal of providing natural and intuitive user interfaces for a touchscreen system. *See* Korhonen at 1:41-46 ("The object of the invention is to … present[] a method which is faster, easier and clearer than the prior methods [of scrolling]."); Narutaka, [0006] ("screen scrolling using a screen display that is simpler, faster and more intuitive is realized"); Westerman at 9 ("It is another object of the present invention to provide an improved method for invoking cursor motion continuation only when the user wants it"). Thus, the references are in the same field of endeavor. A POSITA faced with a problem relating to improving user interfaces for touchscreen systems would have been led to look to the combined teachings of Korhonen, Narutaka, and Westerman to address that problem.

210. Each of Korhonen, Narutaka, and Westerman also addresses their goal of improving touchscreen user interfaces by using touch gestures to differentiate between different intended user instructions. *E.g.*, Korhonen at 6:50-7:8 (display and touch surface "operate[] alternatively in numerous other ways," including "scrolling one element at a time" by touching a direction arrow, "accelerating scrolling by touching … [for] a longer time," or "touching … continuously in an area"); Narutaka, [0019]-[0020] (detecting a "touch input" operation for a short touch, and a scroll for a long touch); Westerman at 72 (describing simulated "mouse clicks … generated by a tap of a fingertip pair," "double-clicks … by a single tap of three fingertips," and "[w]indow scrolling … allocated to slides of four fingers"). As previously noted, a shared understanding of fundamental interaction tasks meant that two user interfaces that both enabled selecting, pointing, and scrolling would have been seen by a POSITA as close relatives in the family of interactive systems, no matter what particular gestures (or even interface devices) were used to implement the tasks. A "select" is still a "select", whether instantiated as using a mouse to move a cursor then clicking the mouse, or as a tap by one's finger. Given the common goals and overlapping interaction tasks of Korhonen, Narutaka, and Westerman, a POSITA would have been motivated to combine their teachings in the manner described.

211. Each of Korhonen, Narutaka, and Westerman further disclose systems with similar structure and complementary features. Korhonen discloses mobile stations resembling a cellular phone with a touch sensitive display, through which a user may view and manipulate lists of data elements. *See* Korhonen at 2:55-3:4; *id*. at Figures 1, 2. Narutaka discloses a control system with a touch screen that allows a user to scroll through displayed data. *See* Narutaka, [0008], [0010]. Westerman discloses a touch-sensitive surface, such as a touchscreen, that senses and classifies gestures by touches from multiple fingers. *See* Westerman at Abstract, claims 14-15. A POSITA would have understood the touch-sensitive systems disclosed by each of Korhonen, Narutaka, and Westerman to relate to generally compatible technology and gesture-interpretation techniques. Because each of these inventions share common goals and overlap in the interactions they support, I further believe that where they are complementary, a POSITA would have been motivated to and would have seen the benefit of incorporating gesture interpretation techniques disclosed in any of these references into any of the others.

212. Even further, the claims of the ’387 patent recite a combination of familiar elements that are taught by Korhonen, Narutaka, and Westerman. As described in this section, the claims relate generally to sensing the speed, direction, and duration of a finger touch; performing a scroll or drag function in response to this sensing; and, for the scrolling scenario, decaying and stopping upon a touch or

end signal. All of this functionality was well known years before the ’387 patent was filed. *See generally* Section VI(B). As discussed in the preceding paragraphs, the claims do not present this known functionality in any new or unusual way, nor do they indicate that it achieved surprising results.

213. Therefore, it would have been obvious to combine teachings from the references in the manner presented below.

1. Independent Claim 1

214. The preamble of claim 1 reads: “An improved touch-screen image scrolling system, comprising: ….” Although I am told that the preamble may not be a limitation, it is disclosed by Korhonen. Korhonen discloses “a method and an arrangement for scrolling information (12) presented on a display of a mobile station.” Korhonen at Abstract. A finger or other “pointing means” is “moved on the active area of the display (15), and the displayed part of the information (12, 13, 14) … is scrolled in the direction of the movement of the pointing means.” *Id*. The “active area” of the display “is a touch sensitive area in order to control the contents, in this case the list, of the display 22.” *Id*. at 5:5-7. This active area “can be touched with a pointing means, here a finger 29, in order to scroll the list.” *Id*. at 5:9-11.

215. A POSITA would have understood Korhonen’s disclosure as indicating that the “touch sensitive” display is a touchscreen. Figure 2, which depicts a

user finger approaching the touch sensitive display for purposes of scrolling and selecting the list items, further confirms that the touch sensitive display is a touchscreen. Korhonen further teaches that "[t]ouch screens are also already known in personal computers, in personal digital assistants (PDA) and in oscilloscopes" and that "[a] touch screen can be realized with any display unit known per se" (Korhonen at 1:36-38, 3:49-50), which further indicates that Korhonen's touch sensitive display was a touchscreen.

216. Element 1[a] recites "an electronic image display screen." Korhonen's "mobile station 21 comprises i.a. a display 22." Korhonen at 4:55-56; *id*. at 6:5-8 ("In order to scroll the displayed information the arrangement includes … a display 43 for presenting the information"). Figure 2 illustrates this display 22 (*id*. at 4:54-55):

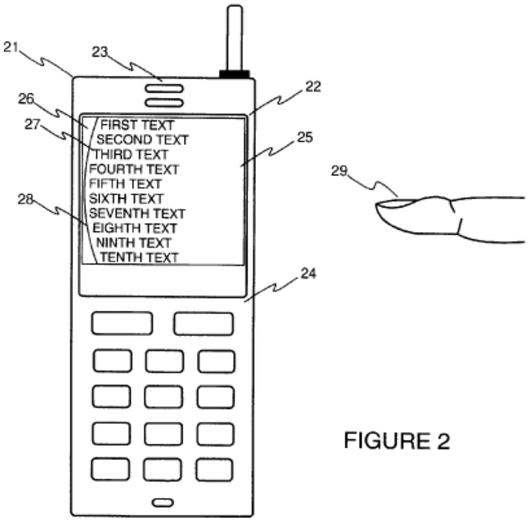


FIGURE 2

217. The display screen displays an electronic image of a list 26. *Id*. at 4:58 ("A list 26 is shown on the display 22."). Korhonen therefore discloses "an electronic image display screen."

218. Element 1[b] recites "a microprocessor coupled to said display screen to display information thereon and to receive interactive signals therefrom." Korhonen's mobile station includes "means for scrolling the [displayed] list 25 [that] comprise a central processing unit 41." Korhonen at 6:12-14. Figure 4 of Korhonen depicts CPU 41 as an "essential" part of the mobile station (*id*. at 6:4-5):

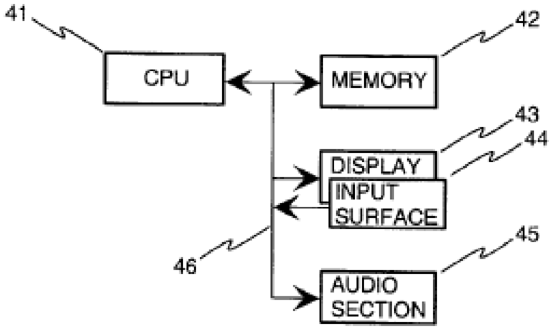


FIGURE 4

219. Figure 4 shows that CPU 41 is coupled to and exchanges signals with the display. Korhonen discloses that "[t]he data communication between the parts 41 – 45 is realized through the microprocessor bus 46." *Id*. at 6:30-32. Korhonen therefore discloses "a microprocessor coupled to said display screen to display information thereon and to receive interactive signals therefrom."

220. Element 1[c] recites "timer means associated with said microprocessor to provide timing capacity therefor." The broadest reasonable interpretation of "timing means" is a timer that provides timing capacity for a microprocessor. Korhonen's mobile station performs various time-based calculations. For example, the system slows scrolling according to the time-based "equation $\nu = \nu_0 - t^2 / K$ where v is the scrolling speed, v_0 is the scrolling speed at the beginning, t is the time, and K is a constant." Korhonen at 7:27-37. The mobile station "count[s] … the time t start[ing] at the moment when the scrolling is left on without user control." *Id*. at 7:35-37.

221. Korhonen's mobile station can also accelerate scrolling by determining how long a touch surface has been touched. *Id*. at 7:1-3, 6-8. This timing functionality necessarily requires a timer that provides "timing capacity" for the microprocessor. A POSITA would have recognized that all timing functionality of a computer is necessarily performed by a timer component that is associated with the computer's processing unit. Thus, a POSITA would have understood Korhonen's disclosures of a microprocessor and time-based calculations to necessarily disclose a timer. The '387 patent itself discloses that timers were "inherent" in computers as of the priority date. '387 patent at 5:31-35.

222. Additionally and alternatively, a POSITA would have found it obvious to incorporate a timer into Korhonen's system to provide the microprocessor

with timing capacity. Timers and counters have long been an essential component of embedded systems, as both Philips and the district court have recognized. Ex. 1022 at 13; Ex. 1023 at 5, n.9. Timers provide a microprocessor with basic functionality such as measuring elapsed time (by counting processor clock ticks or with a real time clock) and timing external events. As of 2001, effectively all microprocessors had a basic on-chip timer. A POSITA would have seen the value in incorporating this type of well-known timer into Korhonen's CPU in order to accomplish the time-based measurements disclosed in Korhonen. Practically speaking, a timer would have been necessary to perform any of those operations—there would have been no way to perform those functions without a timer. Therefore, I understand Korhonen to necessarily disclose "timer means associated with said microprocessor to provide timing capacity therefore" and I also understand that a POSITA would have found it obvious to incorporate a timer that provides timing capacity for a microprocessor into Korhonen's system.

223. Element 1[d] recites "a source of scroll format data capable of display on said display screen." Korhonen discloses that a scrollable list is displayed on the display screen. A "list 12 with its elements 13, 14 is arranged on the outer surface of an imaginary cylinder 11. Only a part of the list 12, or one or more elements 14 35 are visible at a time on the display 15 of the device." Korhonen at

4:32-35. Figure 1 shows this "preferred list form according to the invention" (*id*. at 4:31-32):

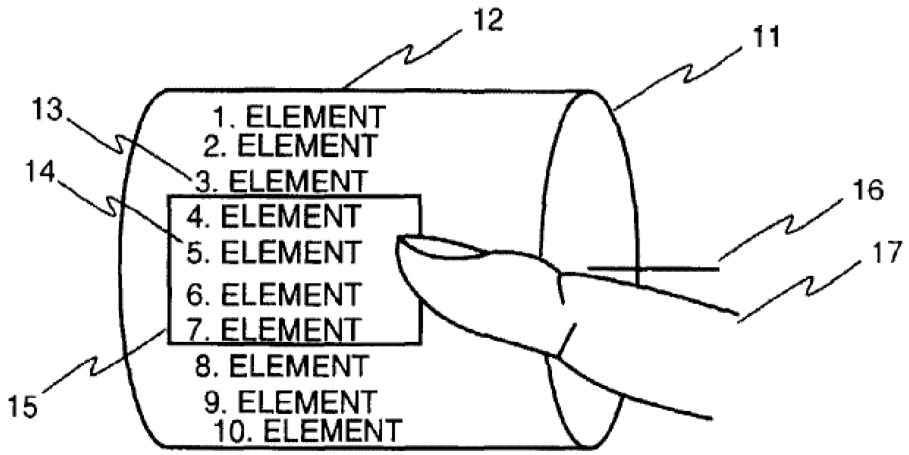


FIGURE 1

224. Korhonen discloses that the scrollable list data is stored in memory ("a source of scroll format data"). "In order to scroll the displayed information the arrangement includes a memory 42 for storing the displayed information." Korhonen at 6:5-7. Figure 4 depicts memory 42:

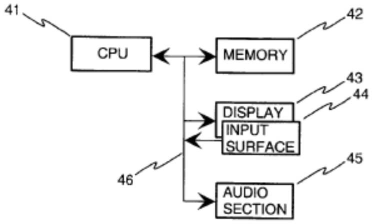


FIGURE 4

Therefore, I understand Korhonen to disclose "a source of scroll format data capable of display on said display screen."

225. Element 1[e]: "a keyboard coupled to said microprocessor to provide input control signals thereto." Korhonen discloses that "[t]he mobile station 12 comprises … a keyboard 24 in order to use the normal functions and to input data." Korhonen at 4:55-58; *see also id*. at Figure 4, element 24. I therefore understand Korhonen to disclose "a keyboard coupled to said microprocessor to provide input control signals thereto."

226. Element 1[f] recites "finger touch program instructions associated with said microprocessor for sensing the speed, direction and time duration of a finger touch contact with said display screen." Korhonen discloses that the "means for scrolling the list" comprise "a program in the memory 42 which reads the display control area 25 in order to detect a touch and the touch point of the pointing means 28" and "scroll[s] the displayed information 26 on the display 22, 20 *so that the displayed information is scrolled in the movement direction of the pointing means 28* controlled by the movement of the pointing means 28." Korhonen at 6:12-22 (emphasis added). The pointing means can be a finger touch. *E.g.*, *id*. at 3:4-6; *id*. at Figures 1, 2.

227. Korhonen also discloses that the displayed information is scrolled at the speed of the touch. "[T]he displayed part … is arranged to scroll at *a rate*

measured for, or proportional to, the speed of the pointing means at the last moment before it was removed." *Id.* at 2:18-22 (emphasis added). "The speed of the pointing means in the direction of the control area of the display, when the pointing means is removed, determines the initial scrolling speed of the displayed part and the scroll direction." *Id.* at 2:22-26; *see also id.* at 8:20-24 ("said displayed part, which is left scrolling, is arranged to scroll at a rate measured for a vertical speed of said pointing means (17) at the last moment before it was removed"). I understand Korhonen to disclose a mobile station that "sens[es] the speed and direction of motion of said finger touch contact with said display screen."

228. Korhonen discloses "a reading surface 44 for detecting a touch of the pointing means 28" and "a program in the memory 42 which reads the display control area 25 in order *to detect a touch and the touch point of the pointing means 28*." *Id*. at 6:9-18 (emphasis added); *id*. at 1:55-2:6. Korhonen teaches a finger as the pointing means. *Id.* 5:25-26. Upon detecting the presence and location of the touch, the program makes the display scroll in the direction of the finger movement. *Id*. at 6:18-22 ("so that the displayed information is scrolled in the movement direction of the pointing means 28 controlled by the movement of the pointing means 28"); *id*. at 1:55-2:6 ("According to the invention the control area of the display is touched with a pointing means, the pointing means is moved in contact with the control area of the display, and the displayed part of the presented infor-

mation is scrolled in the display in the direction of the movement of the pointing means. The control area of the display can be a touch sensitive area arranged on the surface of the display, or a separate surface in another location of the mobile station for controlling the display."). The program then makes the displayed information continue scrolling "at a rate measured for, or proportional to, the speed of the pointing means at the last moment before it was removed." *Id.* 2:18-22. Korhonen's program therefore senses the duration of the finger touch by sensing when it begins and ends.

229. Korhonen's mobile station must also detect multiple touch samples before it can determine the direction of the finger movement. Each of these samples is separated by a defined period of time—as of 2001 a common touch screen scan rate was 60 times per second (60 Hz, or every 16.67 ms)—and the station would have necessarily sensed the duration of the touch based on the number of samples used to determine direction. If, for example, the mobile station had a 60 Hz scan rate and used three touch samples to determine the direction of the finger touch before starting the scroll, the station would necessarily have sensed the duration of the touch that spanned those three samples: 3 samples times the rate of 60 samples per second, which equals .05 seconds. Thus, I understand Korhonen's mobile station to have necessarily "sens[ed] the … time duration" of a touch contact with the display screen.

230. Korhonen's Figure 5 embodiment further discloses sensing the time duration of a finger touch contact:

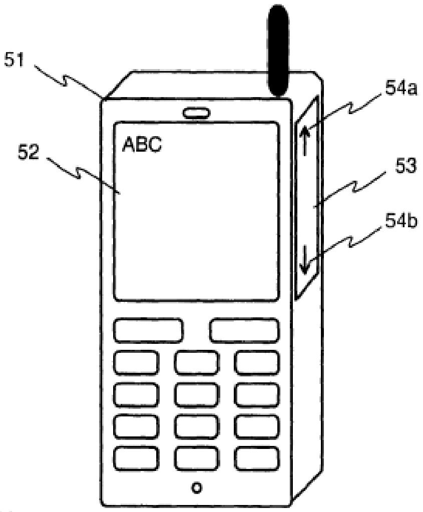


FIGURE 5

231. In this embodiment, a touch-sensitive surface 53 is positioned on the side of the mobile station. Korhonen at 6:46-50. The touch surface operates in the same way as the touch screen. *Id*. at 6:50-52 ("touch surface 53 [preferably] operates in the same way as the active area of the display"). When a user touches the touch surface for a "longer time," the mobile station increases the scroll speed. *Id*. at 7:1-3 (operation of touch surface 53 includes "an accelerating scrolling by touching the touch surface 53 a longer time at a direction arrow 54a, 54b").

232. A POSITA would have found it obvious to incorporate the feature of sensing the duration of a user touch into Korhonen's primary Figure 2 embodiment, to allow the mobile station to accelerate scrolling speed without using a button.

Korhonen seeks to "facilitate the scrolling of a long list" in a manner "faster, easier and clearer than the prior methods." *Id*. at 1:41-46. Korhonen is concerned with providing scrolling features without using components like keys and knobs that occupied space on the device. *E.g.*, *id*. at 1:20-22 ("The scrolling of the lists requires control means … which require space on the device surface."). A POSITA concerned, like Korhonen, with providing list-scrolling features while conserving display space would have seen the benefit of moving the duration-sensing capability into the display itself—rather than the external touch surface that "require[s] space on the device surface"—to minimize the number of external controls on the device. This would have directly advanced Korhonen's objective that "no separate scrolling keys [be] required." Korhonen at 4:7-9. A POSITA would also have found it trivial to incorporate this functionality into the touch sensitive display of Korhonen's Figure 2 embodiment—indeed, Korhonen describes the touch sensitive display and touch surface as operating in the same essential way. *Id*. at 6:50-52; *see also id*. at 5:12-15 ("In figure 2 the display control area is the above mentioned active area 25 of the display, but alternatively it is a separate touch surface outside the display for controlling the display, as shown in figure 5.").

233. Additionally, Narutaka discloses this limitation. Narutaka discloses that "[w]hen the touch panel 2 detects that the screen … has been touched by the finger 7 of the operator … the CPU 5 imports the touch location data and checks

whether or not the touch location data is continuously output by the interface circuit 3 for a predetermined fixed amount of time." Narutaka, [0018]. If the finger touch is shorter than this fixed amount of time, Narutaka's CPU interprets the instruction as an input. *Id*., [0019]. If the finger touch lasts "for the fixed amount of time or longer," however, the CPU interprets the input as a scroll instruction. *Id*., [0020].

234. A POSITA would have found it obvious to incorporate Narutaka's teaching of sensing the time duration of a user's touch into Korhonen's system and would have seen the benefit in doing so. Korhonen aims to present "a faster, easier, and clearer" method of "scrolling of a long list." Korhonen at 1:41-46. The ability to sense the duration of a user's touch directly furthers Korhonen's stated goal, because it allows the system to discern the user's intended operation: the system can distinguish between touches intended to be input and those intended to be a scroll command. Narutaka, [0019]-[0020]. Further, Narutaka's system allows "scrolling … [to be] simpler, faster, and more intuitive" (*id*., [0006]), which furthers Korhonen's goal of facilitating a "faster, easier and clearer" method of scrolling a list. A POSITA would have found it straightforward to incorporate the ability to sense duration of a touch into Korhonen's system. Given that Korhonen's system already detects the direction of the finger movement, a POSITA would have understood the system to already include the ability to sense duration of a touch. The

standard timer in Korhonen's central processing unit would have provided this ability. And incorporating the express step of sensing touch duration in this familiar way would have resulted in predictable operation, with no surprising results. Therefore, I understand Korhonen to both disclose "finger touch program instructions associated with said microprocessor for sensing the speed, direction and time duration of a finger touch contact with said display screen" and to render this obvious in combination with Narutaka.

235. Element 1[g] recites "scrolling motion program instructions associated with said microprocessor responsive to said duration of said finger touch contact such that, when said duration exceeds a first given preset minimum time and is accompanied by motion along the surface of said screen followed by separation of said finger touch from said screen, a scroll format display on said screen is caused to begin to scroll in said sensed direction and at said sensed initial speed." As discussed for Element 1[f], Korhonen discloses scrolling the displayed data in the sensed direction of the finger movement and at the sensed speed of the finger movement when the finger is separated from the screen. Korhonen at 2:18-26 ("According to a preferred embodiment of the invention said displayed part, which is left scrolling, *is arranged to scroll at a rate measured for, or proportional to, the speed of the pointing means* at the last moment before it was removed. The speed of the pointing means in the direction of the control area of the display, *when the*

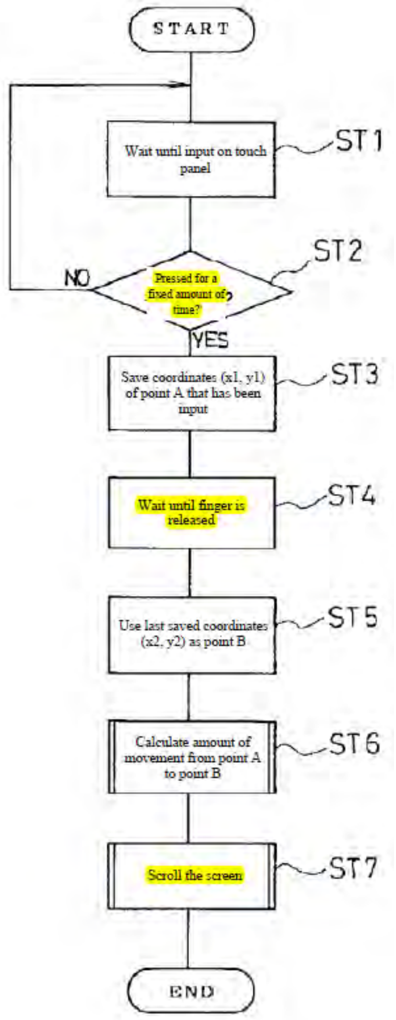
pointing means is removed, *determines the initial scrolling speed of the displayed part and the scroll direction*.") (emphasis added). This "means for scrolling the list" comprises "a program in the memory 42" which "scroll[s] the displayed information 26 on the display 22, 20 so that the displayed information is scrolled in the movement direction of the pointing means 28 controlled by the movement of the pointing means 28." *Id*. at 6:12-22 (emphasis added).

236. As discussed for element 1[f], a POSITA would have understood Korhonen's disclosure that the mobile station calculates the direction of a touch as indicating that the mobile station senses a duration of the touch contact with the display screen: if the station detects a touch, this necessarily means that the touch lasted longer than the time separation between scans of the touch screen for input, conducted at the system's scan rate. Necessarily, this time duration must exceed a "first given preset minimum time": the period between consecutive scans of the touch screen.

237. Additionally, Narutaka discloses this limitation. Narutaka discloses that when the touch screen has detected a finger touch, "the CPU 5 … checks whether or not the touch location data is continuously output by the interface circuit 3 for a predetermined fixed amount of time." Narutaka, [0018]. If the duration of the finger touch lasts "for the fixed amount of time or longer," the CPU interprets the input as a scroll instruction (*id*., [0020]) and causes the screen to scroll

to the end location at which the touch leaves the screen (*id.*, [0021]-[0023]). Figure 2 illustrates this process (shown with highlights added):

FIG. 2


START
Wait until input on touch panel
ST1
NO
Pressed for a fixed amount of time?
ST2
YES
Save coordinates (x1, y1) of point A that has been input
ST3
Wait until finger is released
ST4
Use last saved coordinates (x2, y2) as point B
ST5
Calculate amount of movement from point A to point B
ST6
Scroll the screen
ST7
END

238. A POSITA would have seen the value of incorporating this teaching of Narutaka into Korhonen because this would have allowed the system to determine user intent more accurately. It would also have increased the number of display operations capable of being indicated through various touch gestures. As of

the priority date, a short touch or "click" was commonly used to indicate *selection* or input. *Scrolling* is a distinct interaction task, so a POSITA would have considered it obvious to wait for some small period of time after detecting a touch before starting to scroll, in order to confirm that the user actually intended to scroll and not merely to select an item (or perform some other operation). This would have worked toward Korhonen's goal, which was to present a user with a "faster, easier and clearer" scrolling method. Korhonen at 1:41-46

239. Further, using a time delay before scrolling as taught by Narutaka would have been a routine design choice and one of a limited number of known solutions for differentiating between user touch gestures in a touch screen system. A user interface designer had only two primary touch-related variables to work with in crafting a touch-sensitive system: the nature (*e.g.*, cadence, shape, pressure (for resistive touchscreens)) of a touch, and its duration. A POSITA would have been able to fit Narutaka's time delay naturally into Korhonen's scrolling system. Therefore, I understand that a POSITA would have also found it obvious to combine Korhonen with Narutaka for this limitation as of the filing date of the '387 patent by incorporating Narutaka's disclosure of beginning scrolling when the touch duration exceeds a minimum time into Korhonen's mobile station.

240. Element 1[h] recites "time decay program instructions associated with said microprocessor for reducing the rate of scrolling displacement on said display

screen at a given rate until motion is terminated." Korhonen discloses that the display continues to scroll "even if the pointing means (17) is lifted off the surface of the active area of the display." Korhonen at Abstract. "The displayed part left scrolling will scroll at a rate retarding by itself." *Id.* "Said retardation is effected by applying a suitable, for instance an exponential formula." *Id.* at 2:31-34. "The list scrolling in the display is retarded e.g. according to the equation $\nu = \nu_0 - t^2 / K$ where v is the scrolling speed, v_0 is the scrolling speed at the beginning, t is the time, and K is a constant." *Id.* at 7:27-37. The scrolling is "stopped if the scrolling speed falls below a predetermined limit." *Id.* at 2:35-37. Therefore, Korhonen discloses "time decay program instructions associated with said microprocessor for reducing the rate of scrolling displacement on said display screen at a given rate until motion is terminated."

241. Element 1[i] recites "stopping motion program instructions associated with said microprocessor for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (b) an end-of-scroll signal received from said scroll format data source." Korhonen discloses that scrolling "is stopped by touching the active area of the display (15) with the pointing means (17)." Korhonen at Abstract; *id.* at 5:37-38 ("When required, the list 12 is stopped (step 35) by touch-

ing an element 13 or 14 of the list 12."). Korhonen discloses that the user can instead "[repeat] the scrolling action … until the desired element 13 … is visible on the display 15." *Id*. at 5:33-37. Korhonen's mobile station must therefore be able to distinguish between a user touch intended to *stop* scrolling, and one that is intended to *continue* scrolling. A POSITA would have considered it obvious to require the stop-scrolling touch to be a stationary touch lasting longer than a "preset minimum time" in order to differentiate between touches meant to continue scrolling and touches meant to stop scrolling. It would have been trivial step to add a check for this type of required time duration to the code. This could have been implemented using an if statement, or other conditional statement that stopped the scroll only if the touch duration exceeded the preset time.

242. The '387 patent states that the recited "end of scroll signal" can be an indication that the end of the document has been reached. '387 patent at 1:60-65 ("After the finger separates from the screen, the image continues to move in the same direction at a gradually decreasing speed until motion is stopped manually by touching the screen without movement of the finger, … or until the image reaches its 'end'."). A POSITA would have found it obvious to stop scrolling in Korhonen's system when the end of the displayed list is reached if that happens before a finger touch, especially in the case of short lists. Korhonen directs that an empty space should preferably be inserted at the end of the list. Korhonen at 4:43-45.

Korhonen further states that this empty space is useful in short lists, because it can ensure that the list "will not be repeated on the display." *Id.* at 4:43-50 ("At the end of a list 12 and the next beginning there is preferably arranged an empty space, *particularly when the list 12 is short*, so that the repeated beginning of the list is clearly perceived, and *so that a short list 12 will not be repeated on the display*.") (emphasis added). A POSITA would have taken Korhonen's instruction that short lists should not be repeated on the display, as well as its general goal of presenting a "faster, easier and clearer" manner of scrolling a list (*id*. at 1:41-45), as a reason to stop scrolling the list when the end of the list was reached. To operate in a different fashion—by looping a shorter list, for example—would have been quite *un*natural for a user and would have been counter to Korhonen's goals.

243. It would also have been trivial for a POSITA to implement this feature. For example, a flag could be set to detect when the empty space on the list was reached. In contrast, looping a shorter list would have been very strange and inconvenient for a user and would have opposed Korhonen's goals. Therefore, I understand that it would have been obvious to a POSITA to modify Korhonen's mobile station such that it both stopped upon a touch and at the end of the scrollable list, satisfying the limitation of "stopping motion program instructions associated with said microprocessor for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising:

(a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (b) an end-of-scroll signal received from said scroll format data source."

244. Westerman also discloses this limitation. Westerman discloses that the device display enters a scrolling "slide mode" when "significant motion is detected" in multiple fingers touching a touch surface. Westerman at 68-69. When the system detects liftoff by all the moving fingers, the system enters "motion continuation mode" in which the "scrolling velocity" continues at the pre-liftoff average of the fingers' velocity. *Id*. at 72. "Motion continuation mode does not stop until any of the remaining fingers not in the synchronized subset are lifted or more fingers touch down." *Id*. Additionally, "the host computer can send a signal instructing motion continuation mode to be canceled if the cursor reaches the edge of the screen or end of a document." *Id*. Thus, I understand Westerman to disclose program instructions for terminating scrolling upon the *first occurrence* of either (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time (when "fingers touch down" for longer than the touch screen's scan rate) and (b) an end-of-scroll signal received from said scroll format data source (the signal from the host computer indicating the end of the screen or document).

245. A POSITA would have found it obvious to incorporate Westerman's teaching of terminating scrolling upon either an end of scroll signal *or* a stationary finger touch, whichever is first, for similar reasons to those discussed above in paragraphs 238-239. Terminating scrolling of a document when it reached its end would have been trivial for a POSITA to implement, and it would have furthered Korhonen's goal of faster, easier, and clearer scrolling operation that avoided the awkward and unnatural looping of shorter lists.

2. Dependent Claim 4

246. Claim 4 recites "The improved touch-screen image scrolling system of claim 1, wherein said group of signals for terminating scrolling, displacement of the image on said display screen further comprises (a) a signal indicating that the rate of scrolling displacement on said screen has decayed to a value below a predetermined given value." Korhonen discloses that "scrolling is stopped if the scrolling speed falls below a predetermined limit." Korhonen at 2:35-37; *id*. at 8:35-36 ("said scrolling is stopped if said scrolling speed falls below a predetermined limit"). This limit is "selected [to be] substantially lower than the limit for leaving the display part to scroll. Thus the scrolling is … completely stopped only when in practice it appears to have stopped." *Id*. at 2:37-41. Therefore, Korhonen discloses that the "group of signals for terminating scrolling, displacement of the image on said display screen further comprises (a) a signal indicating that the rate of

scrolling displacement on said screen has decayed to a value below a predetermined given value."

3. Dependent Claim 5

247. Claim 5 recites "[t]he improved touch-screen image scrolling system of claim 1, wherein said microprocessor, and said timer means together comprise a processing unit of a conventional computer." Korhonen discloses a central processing unit that executes the scrolling program code stored in memory. "These means for presenting the list 26 on the display 22 include a central processing unit 41 and a program in the memory 42 which creates a picture of a cylindrical list 26 on 40 the display 43." Korhonen at 6:36-40; *id*. at 6:12-19 ("means for scrolling the list 26 comprise a central processing unit 41 … and a program in the memory 42 which reads the display control area 25 in order to detect a touch and the touch point of the pointing means 28, and in order to scroll the displayed information 26 on the display 22"); *id*. at Figure 4, "CPU" element 41. As I discussed for element 1[c], a POSITA would have understood Korhonen's central processing unit to have included a microprocessor and timer.

4. Dependent Claim 6

248. Claim 6 recites "[t]he improved touch-screen image scrolling system of claim 5, wherein said source of scroll format data capable of display on said display screen comprises part of the memory of said conventional computer."

Korhonen discloses that the scrollable list data is stored in memory within the mobile station. "In order to scroll the displayed information the arrangement includes a *memory 42 for storing the displayed information*." Korhonen at 6:5-7 (emphasis added); *id*. at Figure 4, "memory" element 42. Korhonen recognizes that "[i]nformation such as facts, names and/or numbers concerning persons or things, are often stored in the form of a list in the memory of a mobile station." *Id*. at 1:7-9. Korhonen therefore discloses that the "source of scroll format data capable of display on said display screen comprises part of the memory of said conventional computer."

5. Independent Claim 7

249. The limitations of claim 7 (*see* para. 153) are nearly identical to those of claim 1. The only substantive change is that claim 7 recites a "computer apparatus" instead of a "microprocessor." A POSITA would have considered a "computer apparatus" that could perform all of the recited limitations to include a microprocessor. At a minimum, a POSITA would have understood a "computer apparatus" to be broader than a "microprocessor."

250. The preamble of claim 7 is identical to the preamble of claim 1. I am informed that the preamble of a claim may not be limiting. Even so, for the reasons I have discussed in connection with the preamble of claim 1, the preamble of claim 7 is disclosed by Korhonen.

251. Element 7[a] is identical to element 1[a]. Korhonen discloses this limitation for the same reasons discussed in connection with element 1[a].

252. Element 7[b] is identical to element 1[b], except that 7[b] recites a "computer apparatus coupled to said display screen" while 1[a] recites a "microprocessor." A POSITA would have considered a "computer apparatus" to necessarily include and be broader than a "microprocessor," and would have understood Korhonen to disclose this limitation for the same reasons discussed in connection with element 1[b].

253. Element 7[c] is identical to element 1[c], except that 7[c] also recites a "computer apparatus" while 1[a] recites a "microprocessor." A POSITA would have understood Korhonen to disclose this limitation and have rendered it obvious for the same reasons discussed in connection with element 1[c].

254. Element 7[d] ("said computer apparatus having capacity to store scroll format data capable of display on said display screen") is similar to element 1[d] ("a source of scroll format data capable of display on said display screen"). A POSITA would have understood Korhonen to disclose this limitation for the same reasons discussed in connection with element 1[d].

255. Element 7[e] is identical to element 1[e], except that 7[e] recites a "computer apparatus" while 1[e] recites a "microprocessor." A POSITA would

have understood Korhonen to disclose this limitation for the same reasons discussed in connection with element 1[e].

256. Element 7[f] is identical to element 1[f], except that 7[f] recites a "computer apparatus" while 1[f] recites a "microprocessor." A POSITA would have understood Korhonen to have disclosed this limitation and rendered it obvious, alone or in combination with Narutaka, for the same reasons discussed in connection with element 1[f].

257. Element 7[g] is identical to element 1[g], except that 7[g] recites a "computer apparatus" while 1[g] recites a "microprocessor" and that element 7[g] does not require a touch contact to be "followed by separation of said finger touch from said screen" (as element 1[g] requires). A POSITA would have understood Korhonen to have disclosed this limitation and rendered it obvious in combination with Narutaka for the same reasons discussed in connection with element 1[g].

258. Element 7[h] is identical to element 1[h], except that 7[h] recites a "computer apparatus" while 1[h] recites a "microprocessor." A POSITA would have understood Korhonen to have disclosed this limitation, for the same reasons discussed in connection with element 1[h].

259. Element 7[i] is identical to element 1[i], except that 7[i] recites a "computer apparatus" while 1[i] recites a "microprocessor." A POSITA would have understood Korhonen to have rendered this limitation obvious, alone or in

combination with Westerman, for the same reasons discussed in connection with element 1[i].

260. In sum, Korhonen in combination with Narutaka and Korhonen in combination with Narutaka and Westerman renders claim 7 obvious for the same reasons discussed for claim 1.

6. Independent Claim 9

261. The limitations of claim 9 (*see* para. 166) are also highly similar to those of claim 1. Claim 9 recites a "method of controlling the scroll-like display of data on an electronic display screen" instead of a "touch-screen image scrolling system," and is generally the method counterpart to claim 1.

262. The preamble of claim 9 ("[a]n improved method of controlling the scroll-like display of data on an electronic display screen") is similar to the preamble of claim 1 and element 1[a] ("[a]n improved touch-screen image scrolling system, comprising an electronic image display screen"). I am informed that the preamble may not be limiting. Even so, for the reasons I have discussed in connection with the preamble of claim 1 and element 1[a], the preamble of claim 7 is disclosed by Korhonen.

263. Element 9[a] ("sensing the duration of finger touch contact time with an electronic display screen having scrollable data displayed thereon") is similar to element 1[f] ("finger touch program instructions … for sensing the … time dura-

tion of a finger touch contact with said display screen"). A POSITA would have understood Korhonen to have disclosed this limitation and rendered it obvious, alone or in combination with Narutaka, for the same reasons discussed in connection with element 1[f].

264. Element 9[b] ("sensing the speed and direction of motion of said finger touch contact with said display screen") is similar to element 1[f] ("finger touch program instructions … for sensing the speed [and] direction … of a finger touch contact with said display screen"). A POSITA would have understood Korhonen to have disclosed this limitation and rendered it obvious, alone or in combination with Narutaka, for the same reasons discussed in connection with element 1[f].

265. Element 9[c] ("initiating scrolling motion of said scrollable data on said display screen in said sensed direction and at said sensed speed") is similar to element 1[g] ("scrolling motion program instructions … such that … a scroll format display on said screen is caused to begin to scroll in said sensed direction and at said sensed initial speed"). A POSITA would have understood Korhonen to have disclosed this limitation and rendered it obvious in combination with Narutaka for the same reasons discussed in connection with element 1[g].

266. Element 9[d] ("slowing the speed of said scrolling motion from the initiated speed thereof, at a predetermined rate") is similar to element 1[h] ("time de-

cay program instructions … for reducing the rate of scrolling displacement on said display screen at a given rate”). A POSITA would have understood Korhonen to have disclosed this limitation obvious for the same reasons discussed in connection with element 1[h].

267. Element 9[e] (“terminating said scrolling motion when one of the conditions comprising the following group of conditions is sensed: (a) a substantially stationary finger touch having a finite duration is sensed; (b) an end-of-scroll signal is sensed”) is similar to element 1[i] (“stopping motion program instructions … for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (b) an end-of-scroll signal received from said scroll format data source”). A POSITA would have understood Korhonen to have rendered this limitation obvious, alone or in combination with Westerman, for the same reasons discussed in connection with element 1[i].

268. In sum, Korhonen in combination with Narutaka and Korhonen in combination with Narutaka and Westerman renders claim 9 obvious.

7. Dependent Claim 10

269. Claim 10 depends from claim 7. Claim 10 recites limitations that are similar to those of claim 4, and it is obvious over Korhonen and the Korho-

nen/Narutaka, Korhonen/Westerman, and Korhonen/Narutaka/Westerman combinations for the reasons discussed for claims 4 and 7.

E. Claims 2, 3, 8, 11 and 12 are Obvious Over Korhonen, Korhonen and Narutaka, Korhonen and Westerman, and Korhonen, Narutaka, and Westerman in Further Combination with Astala.

270. It is my opinion that claims 2, 3, 8, 11, and 12 of the ’387 patent would have been obvious to a POSITA in view of Korhonen and the combinations of Korhonen and Narutaka, Korhonen and Westerman, and Korhonen, Narutaka, and Westerman, combined with Astala.

1. Dependent Claim 2

271. Claim 2 recites “The improved touch-screen image scrolling system of claim 1, wherein said scrolling motion program instructions further comprise instructions to move said display in correspondence with movement of the finger touch, in response to movement following a touch having a stationary duration greater than said first preset given minimum time and less than a second given preset minimum time.” In simpler terms, claim 2 recites instructions to move the display in correspondence with a touch that lasts longer than a first time threshold (the “first given preset minimum time” from claim 1[g] that triggers scrolling), but is shorter than a second time threshold (“a second given preset minimum time”).

272. Korhonen discloses instructions for moving the display in correspondence with the movement of a finger touch. Korhonen discloses that when a user

touches the display screen, the displayed list is moved with the finger touch. "According to the invention the control area of the display is touched with a pointing means, the pointing means is moved in contact with the control area of the display, and the displayed part of the presented information is scrolled … in the direction of the movement of the pointing means." Korhonen at 1:55-2:3; *id*. at 2:3-5 (control area can be "touch sensitive area … on the surface of the display"); *see also id*. at 2:55-3:12 ("The [displayed] cylinder can be rotated with a pointing means, such as with a finger, by touching the displayed part of the list element and by moving this element with the pointing means … whereby elements which in the motion direction are in front of the element to be moved will be moved outside the display, and new elements will be visible behind this element."). Korhonen further discloses that the list is moved "directly proportionally" to the finger touch. *Id*. at 7:17-19 ("the list elements are moved directly proportionally to the movement of the touch spot on the touch surface 53");. Figure 1 shows how list items are moved on and off display area 15 in correspondence with a finger touch:

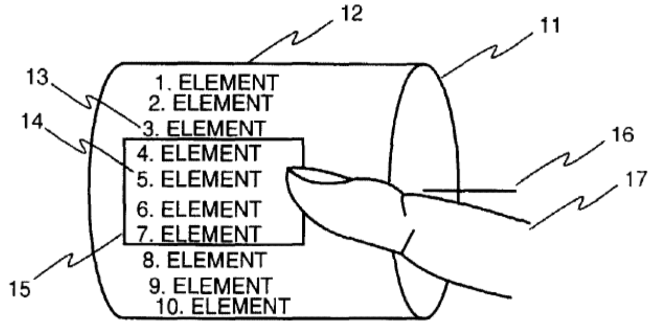


FIGURE 1

273. Korhonen's also discloses that "a program in the memory 42" is responsible for the functionality of "scroll[ing] the displayed information 26 on the display 22, so that the displayed information is scrolled in the movement direction of the pointing means 28." Korhonen at 6:12-22. Korhonen therefore discloses "instructions" for performing the claimed function.

274. Korhonen discloses that the display is moved in correspondence with a finger touch "in response to movement following a touch having a stationary duration." Korhonen consistently talks about a three-step process for moving the list. First, a finger touches the screen. Second, the finger moves while touching the screen. Third, the displayed image moves with the finger. *E.g.*, Korhonen at 1:55-2:3 ("According to the invention the control area of the display is touched with a pointing means, the pointing means is moved in contact with the control area of the display, and the displayed part of the presented information is scrolled … in the direction of the movement of the pointing means."); *id.* at 8:3-22 ("A method for

scrolling information (12, 13, 14) presented on a display of a mobile station, characterized in that a control area of the display is touched with a pointing means (17), said pointing means (17) is moved in contact with said control area of the display, and a displayed part of the presented information (12, 13, 14) is scrolled (34) in the display (15), in a direction of said movement of said pointing means (17)."); *id*. at 5:24-30 ("the list 12 … is touched with a pointing means 17, such as with a finger … and the list 12 is scrolled into the desired direction by moving the pointing means 17 on the surface of the display in this direction (step 34)"); *id*. at 6:14-22 ("a display control area or a reading surface 44 for detecting a touch of the pointing means 28, … and a program in the memory 42 which reads the display control area 25 in order to detect a touch and the touch point of the pointing means 28, and in order to scroll the displayed information 26 on the display 22 …"). A POSITA would have understood Korhonen's differentiation between these steps, where first a finger touches the screen and then separately the finger moves, as indicating that the finger had an initial "stationary" duration of some time duration. Some degree of time separation would have accompanied these steps.

275. It would also have been obvious to a POSITA to check whether the finger touch on the screen of Korhonen's mobile station had been stationary for the preset minimum time before interpreting the touch as an input to move the entire display with the finger. Specifically, a POSITA would have found it obvious to

wait for at least the time period between scans of the touchscreen before deciding to move the display. The POSITA would have seen benefits in requiring the input to be stationary over the fixed preset time duration: this would reduce the possibility of the system "guessing" incorrectly about the user's intent and moving the display when the user wanted to perform some other function. Korhonen discloses a number of other functions the user could perform on the list, such as moving the elements of the list horizontally (*see* Korhonen at 7:38-41), or selecting one of the list elements (*id*. at 1:12-14), or stopping a scroll (*id*. at 2:44-46). A POSITA would have understood this disclosure of the multiple operations a user could perform to necessarily indicate that users had a way to distinguish between the different operations. Allowing a user to cleanly select between operations would have furthered Korhonen's goal of providing a "faster, easier and clearer" way for a user to interact with a scrollable list on a handheld touchscreen device. Korhonen at 1:41-46. Indeed, Korhonen expressly acknowledges the need to distinguish between operations: it specifies that a "transversal movement" to move the display horizontally is preferably "without any scrolling separately left on." Korhonen at 7:41-43. Adding a requirement that the touch be stationary for a fixed amount of time would have been trivial and well within the POSITA's skill set, requiring only the use of a timer and if statement or other conditional statement in the code.

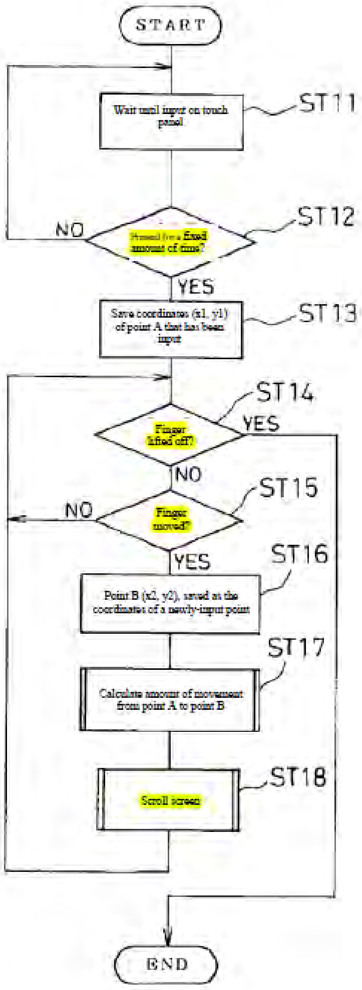
276. Even further, Astala discloses imposing a durational requirement on particular user actions. Astala describes identifying "the particular function … being selected" by a user from "the measured time duration" of a touch. Astala, Abstract. Astala's system "determin[es] whether the [touch] time duration is … greater than the first predetermined value and less than or equal to a second predetermined value" as a way to distinguish between different user-input operations. *Id*. at 10:23-27; *id*. at 9:13-41 ("the short click-long click function could be used on objects to signify selection of the object according to other functions … One skilled in the art will realize that other variations are possible. For example, three or more different time periods of touching may be used to detect different intended input functions. More particularly, the decision box 612 of FIG. 6a would be replaced by a new decision box having three different decision paths, namely, a first path for the case where the detected time period is less than a first predetermined value, a second path for the case where the detected time period is equal to or greater than the first predetermined value and less than or equal to a second predetermined value, and a third path for the case where the detected time period is greater than the second predetermined value."). It would have been obvious to a POSITA to have incorporated Astala's durational requirement into Korhonen, with the result of Korhonen's display being moved in response to a stationary touch with a "duration greater than said first preset given minimum time." Moving the

display in response to a stationary touch would have been a simple design choice for a POSITA. It also would have supplied a straightforward way for Korhonen‘s mobile station to distinguish between the different user operations that could be performed.

277. Additionally, Narutaka discloses this limitation. Narutaka discloses a "screen 8 … [that] is scrolled by the CPU 5 in accordance with the direction and amount of movement of the finger[.]" Naruaka, [0010]. In one of Narutaka's disclosed embodiments, the display is moved with the finger touch. "[I]t is also possible to continuously scroll the screen 8 … when the finger 7 of the operator touches the screen 8 … and is moved over the screen 8 without being lifted off the screen 8." *Id.*, [0026]. If the finger touch is detected "for a predetermined fixed amount of time" (*id.*, [0029]), the system "determines that the operator has given a scroll instruction" (*id.*, [0031]) and the screen is moved in line with the finger movement (*id.*, [0032]-[0035]). Thus, and similarly to Korhonen, Narutaka's disclosed process has three steps: first, detecting the finger touch for the "predetermined fixed amount of time" (*id.*, [0029]); second, identifying the chosen operation (*id.*, [0031]); and third, moving the displayed image along with the finger (*id.*, [0032]-[0035]). Narutaka's Figure 5, shown below with highlighting, illustrates these steps of "scroll[ing the] screen" (step 18) so long as the finger has not "lifted

off" (step 14) and has moved (step 15), all in response to a finger being "pressed for a fixed amount of time" (step 12):

FIG. 5


START
Wait until input on touch panel
ST11
NO
Pressed for a fixed amount of time?
ST12
YES
Save coordinates (x1, y1) of point A that has been input
ST13
ST14
YES
Finger lifted off?
NO
ST15
NO
Finger moved?
YES
ST16
Point B (x2, y2), saved as the coordinates of a newly-input point
ST17
Calculate amount of movement from point A to point B
ST18
Scroll screen
END

278. A POSITA would have understood Narutaka's distinction between steps of the process to indicate that the finger touch had an initial "stationary" duration. In fact, steps 12 and 13 of Figure 5 show that the initial touch location is not even recorded until that location, which Narutaka calls "point A," has been

touched for some amount of time: Figure 5 discloses that the system "Save[s] co-ordinates … of point A that has been input" only if point A has been "Pressed for a fixed amount of time." Further still, Narutaka's "press" terminology is consistent with this disclosure and indicates a still touch. A POSITA would have understood "press" to refer to a motionless touch, unlike other words that indicated movement ("drag," "fling," "flick," etc.).

279. A POSITA would have found it obvious to check whether the finger touch had been stationary for a "fixed amount of time" before interpreting the touch as a input to move the entire display. The POSITA would have understood the benefits of requiring the input to be stationary over the fixed time duration: this would reduce the possibility of the system "guessing" incorrectly about the user's intent and moving the display when the user wanted to perform some other function, such as cycle through elements of the list. Korhonen acknowledges the need to distinguish between scrolling in the direction of the touch and moving the display side-to-side: it specifies that the "transversal movement" to move the display sideways is preferably performed "without any scrolling separately left on." Korhonen at 7:41-43. Adding the requirements for sensing a stationary touch for a fixed amount of time would have been trivial and well within a POSITA's skill set (*see supra* ¶ 276).

280. Astala discloses that the finger touch must not exceed "a second given preset minimum time" in order for the display to move with the finger. Astala discloses "detect[ing] different intended input functions" depending on "different time periods of touching." Astala at 9:27-29. For example, if "the total time of the touch input, Ttouch, was greater than a threshold value, LC, for a long click," the system detects that "a long click has been input." *Id*. at 8:65-9:2. "If Ttouch was less than or equal to LC, however, a short click … has been input." *Id*. at 9:5-8. The system may interpret the short and long clicks to indicate different intended functions: "a short click selecting an object for a drag or move function, while a long click selects the object for opening or activation." *Id*. at 9:15-18.

281. Similarly, "three or more different time periods of touching may be used to detect different intended input functions." *Id*. at 9:27-29. By using different predefined time periods to detect different intended user inputs, Astala's system can take different actions based on different touch durations: "a first [decision] path [is followed] … where the detected time period is less than a first predetermined value, [and] a second path [is followed] … where the detected time period is equal to or greater than the first predetermined value and less than or equal to a second predetermined value." *Id*. at 9:27-36. Thus, Astala discloses a "a second given preset minimum time" that is used to determine an appropriate function.

282. A POSITA would have found it obvious to incorporate Astala's teaching of using "a second given preset minimum time" to distinguish between intended user inputs in the Korhonen/Narutaka system, such that the system moves the screen with the finger only if the initial stationary touch is between the first and second time durations. As of 2001, it was common to use the duration of a touch to select one of multiple user operations. Both Narutaka and Astala recognize this. *E.g.*, Narutaka, [0019]-[0020] (identifying "an instruction … [for] touch input" for touches shorter than some duration, but "a scroll instruction" for longer touches); Astala at 9:27-29 ("different time periods of touching may be used to detect different intended input functions"). A POSITA would have understood this well-known technique of recognizing different durations of touch input to carry the benefit of allowing the system to differentiate between multiple user-desired operations. Korhonen discloses that a user can choose various different operations to perform on the list: a user can move the list with a sustained finger touch (*e.g.*, Korhonen at 1:54-2:3), scroll the list with "flick" or "fling" gesture (*id*. at 2:7-26), move the list horizontally (*id*. at 7:38-43), and select a list element (*id*. at 1:11-14). There would have been various other common operations that a user would have wanted, and have expected to be able, to perform, such as increasing the speed of the scrolling with repeated flicks, editing or deleting a list entry or displaying data underlying a particular list element (*see* Korhonen at 1:9-10 (referencing the

"[d]ata relating to the same fact" that comprises a particular list element. But the small size of the display screen on handheld devices such as Korhonen's would have led a POSITA to move as much as possible of the functionality previously provided by larger peripherals (a keyboard or mouse) into the touchscreen, in the form of recognized touch gestures.

283. Both Korhonen and Astala recognize this motivation to make it easier to manipulate content on a small touchscreen device, which lacks space for keys and other input controls outside of the screen. *E.g.*, Korhonen at 4:7-11 ("The method according to the invention also has the advantage that no separate scrolling keys are required. This is an important advantage, as … there is limited space for keys."); Astala at 2:14-18 ("The keypad of the device … [is] deleted and their functions implemented by the touch screen display screen, thereby allowing more space to be utilized for the display screen."). A POSITA would have understood using touch duration to distinguish between intended operations to advance the goal of reducing keys and external input controls and creating a generally easier and clearer user experience.

284. Additionally, a POSITA would have understood Astala to describe a system with similar structure and operation as those of Korhonen, Narutaka, and Westerman. Like those systems, Astala relates to "electronic devices and more particularly to a touch screen input technique for allowing a user input to an elec-

tronic device having a touch screen." Astala at 1:19-22. Also similarly, Astala provides users with the ability to more conveniently view and manipulate documents. *Id*. at 8:46-9:11, Figs. 6a-d. Even further, Astala's goal is in line with Korhonen, Narutaka, and Westerman: it seeks to allow a user "to enter commands and information into the communications device in an efficient manner" and "allow[] more space to be utilized for the display screen" by using duration and location of touch inputs to interpret user inputs. *Id*. at 1:30-33, 2:16-29. This similarity in structure, operation, and overall field of endeavor would have further led a POSITA to consider it obvious to incorporate Astala's teaching of using different touch time durations to indicate different intended inputs into the systems discussed for claim 9.

2. Dependent Claim 3

285. Claim 3 recites "<u>The improved touch-screen image scrolling system of claim 1, wherein said scrolling motion program instructions further comprise instructions to move a touch-selected item relative to the stationary display in correspondence with movement of said finger touch, in response to motion following a touch having a stationary duration greater than said second given preset minimum time.</u>" Claim 3 recites moving a displayed item with a touch, if the touch lasts longer than a second time threshold (for example, the "second given preset minimum time" that indicates movement of the display along with the finger).

286. Astala discloses a touchscreen device that uses the detected location and "measured time duration" of an object "such as a finger" to determine the operation that the user intends to select. Astala, Abstract. Astala discloses that short-click and long-click inputs can be used to perform a "drag or move" function on a displayed object. "For example, the short click-long click function could be used on objects to signify selection of the object according to other functions. One example of this would be a short click selecting an object for a drag or move function, while a long click selects the object for opening or activation." Astala at 9:13-18. This reference to a "drag" or "move" function would have been understood as indicating a finger touch that "drags" or "moves" a selected object across the display in correspondence with the touch. This type of operation was a common and widespread feature on touchscreen systems as of the priority date. For example, the Minsky article published in the 1980s describes moving and dragging objects around the screen. *E.g.*, Ex. 1013 at 6 ("The user can drag any object (button, logic gate, etc.) through a doorway.").

287. A POSITA would have found it obvious to incorporate Astala's technique of dragging or moving a touch-selected object relative to the display into Korhonen's mobile station. Korhonen discloses displaying a list of elements such as "facts, names and/or numbers concerning persons or things." Korhonen at 1:7-8; *id*. at 1:9-10 ("Data relating to the same fact form an element of the list."). Korho-

nen recognizes that "there are known methods" for "selecting an element in a list." *Id*. at 1:11-12. A POSITA would have understood a drag or move operation as being a commonly-used one of these "known methods" for selecting an item, as Minsky and other early publications demonstrate. Additionally, a POSITA would have seen especial value in providing drag or move operations with respect to a list embodiment. Allowing drag and move operations would have enabled a user to reorder list elements easily. Drag and move operations could also have allowed a user to delete a list element. For example, a user might drag an element off the list and release it, or drag it to a specified area of the display (such as a trash can icon). A POSITA would have recognized that facilitating touch-initiated drag operations for list reordering and deletion would have furthered Korhonen's goals of providing a "faster, easier and clearer" manner of interacting with a displayed list while reducing the need for external "control means" on the device. Korhonen at 1:33-35, 1:41-46. And because the details of drag and move touch operations were well known, incorporating them into Korhonen's mobile station would have produced no surprising results.

288. Astala discloses that specific touch time durations can be used to identify different intended user inputs. Astala at 9:27-29 ("three or more different time periods of touching may be used to detect different intended input functions"). Astala further discloses that different touch time durations can be used to differen-

tiate between drag operations (as disclosed by Korhonen) and other operations. "For example, the short click-long click function could be used on objects to signify selection of the object according to other functions. One example of this would be a short click selecting an object for a drag or move function, while a long click selects the object for opening or activation." *Id*. at 9:13-18. Using touch duration, Astala's system can specifically distinguish between a "time duration [that] is … greater than the first predetermined value and less than or equal to a second predetermined value or *greater than the second predetermined value*." *Id*. at 10:23-28 (emphasis added); *id*. at 10:29-31 (system measures "which of a predetermined plurality of time duration ranges the measured time duration is within").

289. A POSITA would have found it obvious to use a touch duration, as taught by Astala, to indicate that the user's touch was intended to initiate the "drag" operation in Korhonen's system (movement of a touch-selected item in correspondence with movement of the finger touch). As discussed for claim 2, it was well-known as of the priority date to use the duration of a touch to identify user intent. Requiring an extended touch before initiating a drag would have prevented the system from incorrectly "guessing" the user's intent—by cycling through the list elements when the user intended to move the entire list horizontally, for example. Using a long-touch to initiate a drag would have made particular sense when adding the drag functionality to Korhonen's system. That is because a POSITA

would have expected a drag operation to be used less frequently than a scrolling operation in the list environment. It would have been more efficient, from the user's perspective, to require a more intentional and difficult gesture (the long touch) to trigger a drag, and allow users to rely on the simpler short touch for scrolling.

3. Independent Claim 8

290. Independent claim 8 recites limitations that are largely analogous to those of independent claims 1, 7, and 9 and dependent claim 2. *See* para. 189. The limitations of claim 8 are therefore obvious for the reasons discussed for the corresponding limitations of claims 1 and 2.

291. The preamble of claim 8 is identical to the preamble of claim 1. I am informed that the preamble of a claim may not be limiting. Even so, for the reasons I have discussed in connection with the preamble of claim 1, the preamble of claim 8 is disclosed by Korhonen.

292. Element 8[a] is identical to element 1[a]. Korhonen discloses this limitation for the same reasons discussed in connection with element 1[a].

293. Element 8[b] is identical to element 1[b]. A POSITA would have understood Korhonen to disclose this limitation for the same reasons discussed in connection with element 1[b].

294. Element 8[c] is identical to element 1[c]. A POSITA would have understood Korhonen to disclose this limitation and have rendered it obvious for the same reasons discussed in connection with element 1[c].

295. Element 8[d] is identical to element 1[d]. A POSITA would have understood Korhonen to disclose this limitation for the same reasons discussed in connection with element 1[d].

296. Element 8[e] is identical to element 1[e]. A POSITA would have understood Korhonen to disclose this limitation for the same reasons discussed in connection with element 1[e].

297. Element 8[f] is identical to element 1[f]. A POSITA would have understood Korhonen to have disclosed this limitation and rendered it obvious, alone or in combination with Narutaka, for the same reasons discussed in connection with element 1[f].

298. Element 8[g] is similar to element 1[g], except that 8[g] recites the additional element of a duration of a touch that causes scrolling to start "is less than a second given preset minimum that is greater than said first minimum." This is substantively identical to a limitation of claim 2 (which provides that this first touch duration is "greater than said first preset given minimum time and less than a second given preset minimum time"). A POSITA would have understood Korhonen to have rendered element 8[g] obvious in combination with Astala or with

Narutaka and Astala for the same reasons discussed in connection with element 1[g] and claim 2.

299. Element 8[h] ("said scrolling motion program instructions still further comprising instructions to move said display in correspondence with movement of the finger touch, in response to motion following a touch having a stationary duration greater than said first given preset minimum time and less than said second given preset minimum time") is identical to claim 2 ("wherein said scrolling motion program instructions further comprise instructions to move said display in correspondence with movement of the finger touch, in response to movement following a touch having a stationary duration greater than said first preset given minimum time and less than a second given preset minimum time"). A POSITA would have understood Korhonen to have rendered element 8[h] obvious in combination with Astala and Narutaka and Astala for the same reasons discussed in connection with claim 2.

300. Element 8[i] is identical to element 1[h]. A POSITA would have understood Korhonen to have rendered this limitation obvious, alone or in combination with Narutaka, for the same reasons discussed in connection with element 1[h].

301. Element 8[j] is identical to element 1[i]. A POSITA would have understood Korhonen to have rendered this limitation obvious, alone or in combination with Westerman, for the same reasons discussed with element 1[i].

302. In sum, Korhonen in combination with Astala, Korhonen in combination with Narutaka and Astala, Korhonen in combination with Westerman and Astala, and Korhonen in combination with Narutaka, Westerman, and Astala render claim 8 obvious.

4. Dependent Claim 11

303. Claim 11 is a method claim that depends from claim 9. Claim 11 recites limitations that are generally analogous to those of claim 2 (*see* ¶ 202)), and it is obvious over the Korhonen/Astala, Korhonen/Narutaka/Astala, Korhonen/Westerman/Astala, and Korhonen/Narutaka/Westerman/Astala combinations for the reasons discussed for claims 9 and 2.

5. Dependent Claim 12

304. Claim 12 is a method claim that depends from claim 9. Claim 12 recites limitations that are generally analogous to those of claim 3 (*see* ¶ 203), and it is obvious over the Korhonen/Astala, Korhonen/Narutaka/Astala, Korhonen/Westerman/Astala, and Korhonen/Narutaka/Westerman/Astala combinations for the reasons discussed for claims 9 and 3.

X. UNPATENTABILITY OF THE ’064 PATENT CLAIMS

305. Based upon my experience in the fields of user interface design and computer science, my review of the ’064 patent and claims, as well as other materials cited herein and attached as exhibits to the Petitions, it is my opinion that

challenged claims 1-9 of the ’064 patent would have been obvious to a POSITA at the time of the earliest possible priority date of the ’064 patent (December 28, 2001) in view the combinations of Anwar, Korhonen, Narutaka, Westerman, and/or Astala discussed below. More specifically, it is my opinion that claims 1-9 of the ’064 patent are unpatentable as follows:

Challenged Claims	**Unpatentable As Obvious Over**
1, 5, 6, 7	Anwar and Narutaka
1, 5, 6, 7	Anwar, Narutaka, and Westerman
2, 3, 8	Anwar, Narutaka, and Astala
2, 3, 8	Anwar, Narutaka, Westerman, and Astala
4, 9	Anwar, Narutaka, and Korhonen
4, 9	Anwar, Narutaka, Westerman, and Korhonen
1, 4-7, 9	Korhonen and Narutaka
1, 4-7, 9	Korhonen, Narutaka, and Westerman
2, 3, 8	Korhonen, Narutaka, and Astala
2, 3, 8	Korhonen, Narutaka, Westerman, and Astala

306. A detailed discussion of my opinions regarding the unpatentability of claims 1-9 follows. While I discuss specific portions of the prior art references and ’064 patent in this Declaration to exemplify my analysis, I am prepared to use any or all of these references and the ’064 patent to support my opinions.

307. Notably, the claims of the ’064 patent are nearly identical to claims of the ’387 patent. Exhibit 1003 depicts the claims of the ’387 and ’064 patents side

by side. As evident from the exhibit, these claims recite nearly identical limitations, with corresponding subject matter as follows:

Challenged ’064 Patent Claims	Challenged ’387 Patent Claims
Element 1[pre]	Element 1[pre]
Element 1[a]	Element 1[a]
Element 1[b]	Element 1[b]
Element 1[c]	Element 1[c]
Element 1[d]	Element 1[d]
Element 1[e]	Element 1[f]
Element 1[f]	Element 1[g]
Element 1[g]	Element 1[h]
Element 1[h]	Element 1[i]
Claim 2	Claim 2
Claim 3	Claim 3
Claim 4	Claim 4
Claim 5	Claim 5
Claim 6	Claim 6
Element 7[pre]	Element 7[pre]
Element 7[a]	Element 7[a]
Element 7[b]	Element 7[b]
Element 7[c]	Element 7[c]
Element 7[d]	Element 7[d]
Element 7[e]	Element 7[f]
Element 7[f]	Element 7[g]
Element 7[g]	Element 7[h]
Element 7[h]	Element 7[i]
Element 8[pre]	Element 8[pre]
Element 8[a]	Element 8[a]
Element 8[b]	Element 8[b]

Challenged ’064 Patent Claims	Challenged ’387 Patent Claims
Element 8[c]	Element 8[c]
Element 8[d]	Element 8[d]
Element 8[e]	Element 8[f]
Element 8[f]	Element 8[g]
Element 8[g]	Claim 2
Element 8[h]	Element 8[h]
Element 8[i]	Element 8[i]
Element 8[j]	Element 8[j]
Claim 9	Claim 10

308. The claims of the ’064 and ’387 patents differ in only one main respect: the ’064 patent independent claims omit the limitation of “a keyboard coupled to said microprocessor to provide input control signals thereto,” which the independent claims of the ’387 patent recite. The claims of the ’064 patent are therefore broader than those of the ’387 patent, and the same grounds of unpatentability apply to both, as itemized below.

A. Claims 1, 5, 6, and 7 are Obvious Over the Combinations of Anwar and Narutaka and Anwar, Narutaka, and Westerman.

309. It is my opinion that claims 1, 5, 6, and 7 of the ’064 patent would have been obvious to a POSITA in view of the combination of Anwar and Narutaka and Anwar, Narutaka, and Westerman. A POSITA would have been motivated to combine Anwar, Narutaka, and Westerman in the manners specifically

described, and would have seen the value in doing so, for the reasons discussed in Section IX(A).

1. Independent Claim 1

310. Claim 1 of the ’064 patent reads as follows (with labeling added in brackets for ease of discussion):

> 1[pre]. An improved touch-screen image scrolling system, comprising:
>
> 1[a] an electronic image display screen;
>
> 1[b] a microprocessor coupled to said display screen to display information thereon and to receive interactive signals therefrom;
>
> 1[c] timer means associated with said microprocessor to provide timing capacity therefor;
>
> 1[d] a source of scroll format data capable of display on said display screen;
>
> 1[e] finger touch program instructions associated with said microprocessor for sensing the speed, direction and time duration of a finger touch contact with said display screen;
>
> 1[f] scrolling motion program instructions associated with said microprocessor responsive to said duration of said finger touch contact such that, when said duration exceeds a first given preset minimum time and is accompanied by motion along the surface of said screen followed by separation of said finger touch from said screen, a scroll format display on said screen is caused to begin to scroll in said sensed direction and at said sensed initial speed;

> 1[g] time decay program instructions associated with said microprocessor for reducing the rate of scrolling displacement on said display screen at a given rate until motion is terminated;
>
> 1[h] stopping motion program instructions associated with said microprocessor for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (b) an end-of-scroll signal received from said scroll format data source.

Below, I explain the bases for my opinion that claim 1 of the ’064 patent would have been obvious in view of Anwar and Narutaka, and Anwar, Narutaka, and Westerman.

311. The preamble of claim 1 reads: “An improved touch-screen image scrolling system, comprising: ….” The language of this preamble is identical to the preamble of claim 1 of the ’387 patent. Thus, for the reasons I have discussed in Section IX(A) in connection with element 1[pre] of the ’387 patent, Anwar discloses element 1[pre] of the ’064 patent.

312. Element 1[a] recites "an electronic image display screen." The language of this element is identical to the element 1[a] of the '387 patent. Thus, for the reasons I have discussed in Section IX(A) in connection with element 1[a] of the '387 patent, Anwar discloses element 1[a] of the '064 patent.

313. Element 1[b] recites "a microprocessor coupled to said display screen to display information thereon and to receive interactive signals therefrom." The language of this element is identical to the element 1[b] of the '387 patent. Thus, for the reasons I have discussed in Section IX(A) in connection with element 1[b] of the '387 patent, Anwar discloses element 1[b] of the '064 patent.

314. Element 1[c] recites "timer means associated with said microprocessor to provide timing capacity therefor." The language of this element is identical to the element 1[c] of the '387 patent. Thus, for the reasons I have discussed in Section IX(A) in connection with element 1[c] of the '387 patent, Anwar discloses element 1[c] of the '064 patent and renders it obvious.

315. Element 1[d] recites "a source of scroll format data capable of display on said display screen." The language of this element is identical to the element 1[d] of the '387 patent. Thus, for the reasons I have discussed in Section IX(A) in connection with element 1[d] of the '387 patent, Anwar discloses element 1[d] of the '064 patent.

316. Element 1[e] recites "finger touch program instructions associated with said microprocessor for sensing the speed, direction and time duration of a finger touch contact with said display screen." The language of this element is identical to the element 1[f] of the '387 patent. Thus, for the reasons I have discussed in Section IX(A) in connection with element 1[f] of the '387 patent, Anwar discloses element 1[e] of the '064 patent and renders it obvious in combination with Narutaka.

317. Element 1[f] recites "scrolling motion program instructions associated with said microprocessor responsive to said duration of said finger touch contact such that, when said duration exceeds a first given preset minimum time and is accompanied by motion along the surface of said screen followed by separation of said finger touch from said screen, a scroll format display on said screen is caused to begin to scroll in said sensed direction and at said sensed initial speed." The language of this element is identical to the element 1[g] of the '387 patent. Thus, for the reasons I have discussed in Section IX(A) in connection with element 1[g] of the '387 patent, Anwar renders element 1[f] of the '064 patent obvious in combination with Narutaka.

318. Element 1[g] discloses "time decay program instructions associated with said microprocessor for reducing the rate of scrolling displacement on said display screen at a given rate until motion is terminated." The language of this el-

ement is identical to the element 1[h] of the ’387 patent. Thus, for the reasons I have discussed in Section IX(A) in connection with element 1[h] of the ’387 patent, Anwar discloses element 1[g] of the ’064 patent.

319. Element 1[h] recites “stopping motion program instructions associated with said microprocessor for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (b) an end-of-scroll signal received from said scroll format data source.” The language of this element is identical to the element 1[i] of the ’387 patent. Thus, for the reasons I have discussed in Section IX(A) in connection with element 1[i] of the ’387 patent, Anwar renders element 1[h] of the ’064 patent obvious, alone or in combination with Westerman.

2. Dependent Claim 5

320. Claim 5 recites “[t]he improved touch-screen image scrolling system of claim 1, wherein said microprocessor, and said timer means together comprise a processing unit of a conventional computer.” The language of this claim is identical to claim 5 of the ’387 patent. Thus, for the reasons I have discussed in Section IX(A) in connection with claim 5 of the ’387 patent, Anwar discloses claim 5 of the ’064 patent.

3. Dependent Claim 6

321. Claim 6 recites "[t]he improved touch-screen image scrolling system of claim 5, wherein said source of scroll format data capable of display on said display screen comprises part of the memory of said conventional computer." The language of this claim is identical to claim 6 of the '387 patent. Thus, for the reasons I have discussed in Section IX(A) in connection with claim 6 of the '387 patent, Anwar discloses claim 6 of the '064 patent.

4. Independent Claim 7

322. The limitations of claim 7 are nearly identical to those of claim 1. The only change is that claim 7 recites a "computer apparatus" instead of a "microprocessor." A POSITA would have considered a "computer apparatus" that could perform all of the recited limitations to include a microprocessor. At a minimum, a POSITA would have understood a "computer apparatus" to be broader than, and to include, a "microprocessor." Claim 7 of the '064 patent reads as follows (with labeling added in brackets for ease of discussion):

> 7[pre]. An improved touch-screen image scrolling system, comprising:
>
> 7[a] an electronic image display screen;
>
> 7[b] a computer apparatus coupled to said display screen to display information thereon and to receive interactive signals therefrom;

> 7[c] timer means within said computer apparatus to provide timing capacity therefor;
>
> 7[d] said computer apparatus having capacity to store scroll format data capable of display on said display screen;
>
> 7[e] finger touch program instructions associated with said computer apparatus for sensing the speed, direction and time duration of a finger touch contact with said display screen;
>
> 7[f] scrolling motion program instructions associated with said computer apparatus responsive to said duration of said finger touch contact such that, when said duration exceeds a preset minimum time and is accompanied by motion along the surface of said screen, a scroll format display on said screen is caused to begin to scroll in the sensed direction and at the sensed initial speed;
>
> 7[g] time decay program instructions associated with said computer apparatus for reducing the rate of scrolling displacement on said display screen at a given rate until motion is terminated;
>
> 7[h] stopping motion program instructions associated with said computer apparatus for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (b) an end-of-scroll signal received from said scroll format data source.

323. The preamble of claim 7 reads: "An improved touch-screen image scrolling system, comprising: …." The language of this preamble is identical to the preamble of claim 7 of the '387 patent. Thus, for the reasons I have discussed

in Section IX(A) in connection with element 7[pre] of the ’387 patent, Anwar discloses element 7[pre] of the ’064 patent.

324. Element 7[a] recites “an electronic image display screen.” The language of this element is identical to the element 7[a] of the ’387 patent. Thus, for the reasons I have discussed in Section IX(A) in connection with element 7[a] of the ’387 patent, Anwar discloses element 7[a] of the ’064 patent.

325. Element 7[b] recites “a computer apparatus coupled to said display screen to display information thereon and to receive interactive signals therefrom.” The language of this element is identical to the element 7[b] of the ’387 patent. Thus, for the reasons I have discussed in Section IX(A) in connection with element 7[b] of the ’387 patent, Anwar discloses element 7[b] of the ’064 patent.

326. Element 7[c] recites “timer means within said computer apparatus to provide timing capacity therefor.” The language of this element is identical to the element 7[c] of the ’387 patent. Thus, for the reasons I have discussed in Section IX(A) in connection with element 7[c] of the ’387 patent, Anwar discloses element 7[c] of the ’064 patent and renders it obvious.

327. Element 7[d] recites “said computer apparatus having capacity to store scroll format data capable of display on said display screen.” The language of this element is identical to the element 7[d] of the ’387 patent. Thus, for the

reasons I have discussed in Section IX(A) in connection with element 7[d] of the ’387 patent, Anwar discloses element 7[d] of the ’064 patent.

328. Element 7[e] recites “finger touch program instructions associated with said computer apparatus for sensing the speed, direction and time duration of a finger touch contact with said display screen.” The language of this element is identical to the element 7[f] of the ’387 patent. Thus, for the reasons I have discussed in Section IX(A) in connection with element 7[f] of the ’387 patent, Anwar discloses element 7[e] of the ’064 patent and renders it obvious in combination with Narutaka.

329. Element 7[f] recites “scrolling motion program instructions associated with said computer apparatus responsive to said duration of said finger touch contact such that, when said duration exceeds a preset minimum time and is accompanied by motion along the surface of said screen, a scroll format display on said screen is caused to begin to scroll in the sensed direction and at the sensed initial speed.” The language of this element is identical to the element 7[g] of the ’387 patent. Thus, for the reasons I have discussed in Section IX(A) in connection with element 7[g] of the ’387 patent, Anwar renders element 7[f] of the ’064 patent obvious in combination with Narutaka.

330. Element 7[g] discloses “time decay program instructions associated with said computer apparatus for reducing the rate of scrolling displacement on

said display screen at a given rate until motion is terminated." The language of this element is identical to the element 7[h] of the ’387 patent. Thus, for the reasons I have discussed in Section IX(A) in connection with element 7[h] of the ’387 patent, Anwar discloses element 7[g] of the ’064 patent.

331. Element 7[h] recites "stopping motion program instructions associated with said computer apparatus for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (b) an end-of-scroll signal received from said scroll format data source." The language of this element is identical to the element 7[i] of the ’387 patent. Thus, for the reasons I have discussed in Section IX(A) in connection with element 7[i] of the ’387 patent, Anwar renders element 7[h] of the ’064 patent obvious, alone or in combination with Westerman.

332. In sum, Anwar in combination with Narutaka and Anwar in combination with Narutaka and Westerman renders claim 7 obvious.

B. Claims 2, 3, and 8 are Obvious Over the Combinations of Anwar, Narutaka, and Astala and Anwar, Narutaka, Westerman, and Astala.

333. It is my opinion that claims 2, 3, and 8 of the ’064 patent would have been obvious to a POSITA in view of the combination of Anwar, Narutaka, and Astala, and Anwar, Narutaka, Westerman, and Astala.

1. Dependent Claim 2

334. Claim 2 recites "[t]he improved touch-screen image scrolling system of claim 1, wherein said scrolling motion program instructions further comprise instructions to move said display in correspondence with movement of the finger touch, in response to movement following a touch having a stationary duration greater than said first preset given minimum time and less than a second given preset minimum time." The language of this claim is identical to claim 2 of the '387 patent. Thus, for the reasons I have discussed in Section IX(B) in connection with claim 2 of the '387 patent, claim 2 of the '064 patent is obvious over the combination of Astala with Anwar and Narutaka, and separately over the combination of Astala with Anwar, Narutaka, and Westerman.

2. Dependent Claim 3

335. Claim 3 recites "[t]he improved touch-screen image scrolling system of claim 1, wherein said scrolling motion program instructions further comprise instructions to move a touch-selected item relative to the stationary display in correspondence with movement of said finger touch, in response to motion following a touch having a stationary duration greater than said second given preset minimum time." The language of this claim is identical to claim 3 of the '387 patent. Thus, for the reasons I have discussed in Section IX(B) in connection with claim 3 of the '387 patent, claim 3 of the '064 patent is obvious over the combination of

Astala with Anwar and Narutaka, and separately over the combination of Astala with Anwar, Narutaka, and Westerman.

3. Independent Claim 8

336. Independent claim 8 recites limitations similar to those of independent claims 1 and 7. The majority of its language is a nearly verbatim replica of claim 1. *See* Appendix B. In addition to the limitations recited by claim 1, claim 8 recites limitations similar to those of claims 2 and 3: instructions for "mov[ing] said display in correspondence with movement of the finger touch, in response to movement following a touch having a stationary duration greater than said first preset given minimum time and less than a second given preset minimum time" and for "mov[ing] a touch-selected item relative to the stationary display in correspondence with movement of said finger touch, in response to motion following a touch having a stationary duration greater than said second given preset minimum time." *Id*. The limitations of claim 8 are therefore obvious for the reasons discussed for the corresponding limitations of claims 1, 2, and 3.

337. Independent claim 8 of the '064 patent reads as follows (with labeling added in brackets for ease of discussion):

> 8[pre]. An improved touch-screen image scrolling system, comprising:
>
> 8[a] an electronic image display screen;

8[b] a microprocessor coupled to said display screen to display information thereon and to receive interactive signals therefrom;

8[c] timer means associated with said microprocessor to provide timing capacity therefor;

8[d] a source of scroll format data capable of display on said display screen;

8[e] finger touch program instructions associated with said microprocessor for sensing the speed, direction and time duration of a finger touch contact with said display screen:

8[f] scrolling motion program instructions associated with said microprocessor responsive to said duration of said finger touch contact such that, when said duration exceeds a first given preset minimum time, and is less than a second given preset minimum that is greater than said first minimum, and is accompanied by motion along the surface of said screen, a scroll format display on said screen is caused to begin to scroll in the sensed direction and at the sensed initial speed;

8[g] said scrolling motion program instructions further comprising instructions to move a touch-selected item relative to the stationary display in correspondence with movement of the finger touch, in response to motion following a touch having a stationary duration greater than said second given preset minimum time;

8[h] said scrolling motion program instructions still further comprising instructions to move said display in correspondence with movement of the finger touch, in response to motion following a touch having a stationary duration greater than said first given preset minimum time and less than said second given preset minimum time;

> 8[i] time decay program instructions associated with said microprocessor for reducing the rate of scrolling displacement on said display screen at a given rate until motion is terminated;
>
> 8[j] stopping motion program instructions associated with said microprocessor for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (b) an end-of-scroll signal received from said scroll format data source.

338. The preamble of claim 8 reads: "An improved touch-screen image scrolling system, comprising: …." The language of this preamble is identical to the preamble of claim 8 of the '387 patent. Thus, for the reasons I have discussed in Section IX(B) in connection with element 8[pre] of the '387 patent, Anwar discloses element 8[pre] of the '064 patent.

339. Element 8[a] recites "an electronic image display screen." The language of this element is identical to the element 8[a] of the '387 patent. Thus, for the reasons I have discussed in Section IX(B) in connection with element 8[a] of the '387 patent, Anwar discloses element 8[a] of the '064 patent.

340. Element 8[b] recites "a microprocessor coupled to said display screen to display information thereon and to receive interactive signals therefrom." The language of this element is identical to the element 8[b] of the '387 patent. Thus,

for the reasons I have discussed in Section IX(B) in connection with element 8[b] of the ’387 patent, Anwar discloses element 8[b] of the ’064 patent.

341. Element 8[c] recites “timer means associated with said microprocessor to provide timing capacity therefor.” The language of this element is identical to the element 8[c] of the ’387 patent. Thus, for the reasons I have discussed in Section IX(B) in connection with element 8[c] of the ’387 patent, Anwar discloses element 8[c] of the ’064 patent and renders it obvious.

342. Element 8[d] recites “a source of scroll format data capable of display on said display screen.” The language of this element is identical to the element 8[d] of the ’387 patent. Thus, for the reasons I have discussed in Section IX(B) in connection with element 8[d] of the ’387 patent, Anwar discloses element 8[d] of the ’064 patent.

343. Element 8[e] recites “finger touch program instructions associated with said microprocessor for sensing the speed, direction and time duration of a finger touch contact with said display screen.” The language of this element is identical to the element 8[f] of the ’387 patent. Thus, for the reasons I have discussed in Section IX(B) in connection with element 8[f] of the ’387 patent, Anwar discloses element 8[e] of the ’064 patent and renders it obvious in combination with Narutaka.

344. Element 8[f] recites "scrolling motion program instructions associated with said microprocessor responsive to said duration of said finger touch contact such that, when said duration exceeds a first given preset minimum time, and is less than a second given preset minimum that is greater than said first minimum, and is accompanied by motion along the surface of said screen, a scroll format display on said screen is caused to begin to scroll in the sensed direction and at the sensed initial speed." The language of this element is identical to the element 8[g] of the '387 patent. Thus, for the reasons I have discussed in Section IX(B) in connection with element 8[g] of the '387 patent, Anwar renders element 8[f] of the '064 patent obvious in combination with Narutaka.

345. Element 8[g] recites "said scrolling motion program instructions further comprising instructions to move a touch-selected item relative to the stationary display in correspondence with movement of the finger touch, in response to motion following a touch having a stationary duration greater than said second given preset minimum time." The language of this element is identical to the limitation of claim 3 of the '387 patent. Thus, for the reasons I have discussed in Section IX(B) in connection with claim 3 of the '387 patent, Anwar renders element 8[g] of the '064 patent obvious in combination with Astala and Narutaka, and separately in combination with Astala, Narutaka, and Westerman.

346. Element 8[h] discloses "said scrolling motion program instructions still further comprising instructions to move said display in correspondence with movement of the finger touch, in response to motion following a touch having a stationary duration greater than said first given preset minimum time and less than said second given preset minimum time." The language of this element is identical to the element 8[h] of the '387 patent. Thus, for the reasons I have discussed in Section IX(B) in connection with element 8[h] of the '387 patent, Anwar discloses element 8[h] of the '064 patent.

347. Element 8[i] discloses "time decay program instructions associated with said microprocessor for reducing the rate of scrolling displacement on said display screen at a given rate until motion is terminated." The language of this element is identical to the element 8[i] of the '387 patent. Thus, for the reasons I have discussed in Section IX(B) in connection with element 8[i] of the '387 patent, Anwar discloses element 8[i] of the '064 patent.

348. Element 8[j] recites "stopping motion program instructions associated with said microprocessor for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (b) an end-of-scroll signal received from said scroll format data source." The language of this element is identical to the element

8[j] of the ’387 patent. Thus, for the reasons I have discussed in Section IX(B) in connection with element 8[j] of the ’387 patent, Anwar renders element 8[j] of the ’064 patent obvious, alone or in combination with Westerman.

349. In sum, claim 8 is obvious over the combination of Astala with Anwar and Narutaka, and separately over the combination of Astala with Anwar, Narutaka, and Westerman.

C. Claims 4 and 9 are Obvious Over Anwar and Narutaka and Anwar, Narutaka, and Westerman in Further Combination with Korhonen.

350. It is my opinion that claims 4 and 9 of the ’064 patent would have been obvious to a POSITA in view of the combination of Anwar, Narutaka, and Korhonen, and Anwar, Narutaka, Westerman, and Korhonen.

1. Dependent Claim 4

351. Claim 4 recites “<u>[t]he improved touch-screen image scrolling system of claim 1, wherein said group of signals for terminating scrolling displacement of the image on said display screen further comprises (a) a signal indicating that the rate of scrolling displacement on said screen has decayed to a value below a predetermined given value.</u>” The language of this claim is identical to claim 4 of the ’387 patent. Thus, for the reasons I have discussed in Section IX(C) in connection with claim 4 of the ’387 patent, claim 4 of the ’064 patent is obvious over the

combination of Korhonen with Anwar and Narutaka, and separately over the combination of Korhonen with Anwar, Narutaka, and Westerman.

2. Dependent Claim 9

352. Claim 9 recites "[t]he improved method of controlling the scroll-like display of data on an electronic display screen, in accordance with claim 7, wherein said group of conditions to be sensed for terminating said scrolling motion further comprises: the speed of said scrolling motion on said screen slows to a value below a predetermined given value." The language of this claim is identical to claim 10 of the '387 patent. Thus, for the reasons I have discussed in Section IX(C) in connection with claim 10 of the '387 patent, claim 9 of the '064 patent is obvious over the combination of Korhonen with Anwar and Narutaka, and separately over the combination of Korhonen with Anwar, Narutaka, and Westerman.

D. Claims 1, 4, 5, 6, 7, and 9 Are Obvious Over Korhonen, Korhonen and Narutaka, Korhonen and Westerman, and Korhonen, Narutaka, and Westerman

353. It is my opinion that claims 1, 4, 5, 6, 7, and 9 of the '064 patent would have been obvious in view of Korhonen and the combinations of Korhonen and Narutaka, Korhonen and Westerman, and Korhonen, Narutaka, and Westerman render claim 7 obvious. A POSITA would have been motivated to combine Korhonen, Narutaka, and Westerman in the manners specifically described, and would have seen the value in doing so, for the reasons discussed in Section IX(D).

1. Independent Claim 1

354. The preamble of claim 1 reads: "An improved touch-screen image scrolling system, comprising: …." The language of this preamble is identical to the preamble of claim 1 of the '387 patent. Thus, for the reasons I have discussed in Section IX(D) in connection with element 1[pre] of the '387 patent, Korhonen discloses element 1[pre] of the '064 patent.

355. Element 1[a] recites "an electronic image display screen." The language of this element is identical to the element 1[a] of the '387 patent. Thus, for the reasons I have discussed in Section IX(D) in connection with element 1[a] of the '387 patent, Korhonen discloses element 1[a] of the '064 patent.

356. Element 1[b] recites "a microprocessor coupled to said display screen to display information thereon and to receive interactive signals therefrom." The language of this element is identical to the element 1[b] of the '387 patent. Thus, for the reasons I have discussed in Section IX(D) in connection with element 1[b] of the '387 patent, Korhonen discloses element 1[b] of the '064 patent.

357. Element 1[c] recites "timer means associated with said microprocessor to provide timing capacity therefor." The language of this element is identical to the element 1[c] of the '387 patent. Thus, for the reasons I have discussed in Section IX(D) in connection with element 1[c] of the '387 patent, Korhonen discloses element 1[c] of the '064 patent and renders it obvious.

358. Element 1[d] recites "a source of scroll format data capable of display on said display screen." The language of this element is identical to the element 1[d] of the '387 patent. Thus, for the reasons I have discussed in Section IX(D) in connection with element 1[d] of the '387 patent, Korhonen discloses element 1[d] of the '064 patent.

359. Element 1[e] recites "finger touch program instructions associated with said microprocessor for sensing the speed, direction and time duration of a finger touch contact with said display screen." The language of this element is identical to the element 1[f] of the '387 patent. Thus, for the reasons I have discussed in Section IX(D) in connection with element 1[f] of the '387 patent, Korhonen discloses element 1[e] of the '064 patent and renders it obvious in combination with Narutaka.

360. Element 1[f] recites "scrolling motion program instructions associated with said microprocessor responsive to said duration of said finger touch contact such that, when said duration exceeds a first given preset minimum time and is accompanied by motion along the surface of said screen followed by separation of said finger touch from said screen, a scroll format display on said screen is caused to begin to scroll in said sensed direction and at said sensed initial speed." The language of this element is identical to the element 1[g] of the '387 patent. Thus, for the reasons I have discussed in Section IX(D) in connection with element 1[g]

of the ’387 patent, Korhonen renders element 1[f] of the ’064 patent obvious in combination with Narutaka.

361. Element 1[g] discloses “time decay program instructions associated with said microprocessor for reducing the rate of scrolling displacement on said display screen at a given rate until motion is terminated.” The language of this element is identical to the element 1[h] of the ’387 patent. Thus, for the reasons I have discussed in Section IX(D) in connection with element 1[h] of the ’387 patent, Korhonen discloses element 1[g] of the ’064 patent.

362. Element 1[h] recites “stopping motion program instructions associated with said microprocessor for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (b) an end-of-scroll signal received from said scroll format data source.” The language of this element is identical to the element 1[i] of the ’387 patent. Thus, for the reasons I have discussed in Section IX(D) in connection with element 1[i] of the ’387 patent, Korhonen renders element 1[h] of the ’064 patent obvious, alone or in combination with Westerman. Thus, Korhonen and the combinations of Korhonen and Narutaka, Korhonen and Westerman, and Korhonen, Narutaka, and Westerman render claim 1 obvious.

2. Dependent Claim 4

363. Claim 4 recites "[t]he improved touch-screen image scrolling system of claim 1, wherein said group of signals for terminating scrolling displacement of the image on said display screen further comprises (a) a signal indicating that the rate of scrolling displacement on said screen has decayed to a value below a predetermined given value." The language of this claim is identical to claim 4 of the '387 patent. Thus, for the reasons I have discussed in Section IX(D) in connection with claim 4 of the '387 patent, claim 4 of the '064 patent is obvious over the combination of Korhonen with Korhonen and Narutaka, and separately over the combination of Korhonen with Korhonen, Narutaka, and Westerman.

3. Dependent Claim 5

364. Claim 5 recites "[t]he improved touch-screen image scrolling system of claim 1, wherein said microprocessor, and said timer means together comprise a processing unit of a conventional computer." The language of this claim is identical to claim 5 of the '387 patent. Thus, for the reasons I have discussed in Section IX(D) in connection with claim 5 of the '387 patent, Korhonen discloses claim 5 of the '064 patent.

4. Dependent Claim 6

365. Claim 6 recites "[t]he improved touch-screen image scrolling system of claim 5, wherein said source of scroll format data capable of display on said

display screen comprises part of the memory of said conventional computer." The language of this claim is identical to claim 6 of the '387 patent. Thus, for the reasons I have discussed in Section IX(D) in connection with claim 6 of the '387 patent, Korhonen discloses claim 6 of the '064 patent.

5. Independent Claim 7

366. The preamble of claim 7 reads: "An improved touch-screen image scrolling system, comprising: …." The language of this preamble is identical to the preamble of claim 7 of the '387 patent. Thus, for the reasons I have discussed in Section IX(D) in connection with element 7[pre] of the '387 patent, Korhonen discloses element 7[pre] of the '064 patent.

367. Element 7[a] recites "an electronic image display screen." The language of this element is identical to the element 7[a] of the '387 patent. Thus, for the reasons I have discussed in Section IX(D) in connection with element 7[a] of the '387 patent, Korhonen discloses element 7[a] of the '064 patent.

368. Element 7[b] recites "a computer apparatus coupled to said display screen to display information thereon and to receive interactive signals therefrom." The language of this element is identical to the element 7[b] of the '387 patent. Thus, for the reasons I have discussed in Section IX(D) in connection with element 7[b] of the '387 patent, Korhonen discloses element 7[b] of the '064 patent.

369. Element 7[c] recites "timer means within said computer apparatus to provide timing capacity therefor." The language of this element is identical to the element 7[c] of the '387 patent. Thus, for the reasons I have discussed in Section IX(D) in connection with element 7[c] of the '387 patent, Korhonen discloses element 7[c] of the '064 patent and renders it obvious.

370. Element 7[d] recites "said computer apparatus having capacity to store scroll format data capable of display on said display screen." The language of this element is identical to the element 7[d] of the '387 patent. Thus, for the reasons I have discussed in Section IX(D) in connection with element 7[d] of the '387 patent, Korhonen discloses element 7[d] of the '064 patent.

371. Element 7[e] recites "finger touch program instructions associated with said computer apparatus for sensing the speed, direction and time duration of a finger touch contact with said display screen." The language of this element is identical to the element 7[f] of the '387 patent. Thus, for the reasons I have discussed in Section IX(D) in connection with element 7[f] of the '387 patent, Korhonen discloses element 7[e] of the '064 patent and renders it obvious in combination with Narutaka.

372. Element 7[f] recites "scrolling motion program instructions associated with said computer apparatus responsive to said duration of said finger touch contact such that, when said duration exceeds a preset minimum time and is accompa-

nied by motion along the surface of said screen, a scroll format display on said screen is caused to begin to scroll in the sensed direction and at the sensed initial speed." The language of this element is identical to the element 7[g] of the '387 patent. Thus, for the reasons I have discussed in Section IX(D) in connection with element 7[g] of the '387 patent, Korhonen renders element 7[f] of the '064 patent obvious in combination with Narutaka.

373. Element 7[g] discloses "time decay program instructions associated with said computer apparatus for reducing the rate of scrolling displacement on said display screen at a given rate until motion is terminated." The language of this element is identical to the element 7[h] of the '387 patent. Thus, for the reasons I have discussed in Section IX(D) in connection with element 7[h] of the '387 patent, Korhonen discloses element 7[g] of the '064 patent.

374. Element 7[h] recites "stopping motion program instructions associated with said computer apparatus for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (b) an end-of-scroll signal received from said scroll format data source." The language of this element is identical to the element 7[i] of the '387 patent. Thus, for the reasons I have discussed in Section

IX(D) in connection with element 7[i] of the ’387 patent, Korhonen renders element 7[h] of the ’064 patent obvious, alone or in combination with Westerman.

375. In sum, Korhonen and the combinations of Korhonen and Narutaka, Korhonen and Westerman, and Korhonen, Narutaka, and Westerman render claim 7 obvious.

6. Dependent Claim 9

376. Claim 9 recites “[t]he improved method of controlling the scroll-like display of data on an electronic display screen, in accordance with claim 7, wherein said group of conditions to be sensed for terminating said scrolling motion further comprises: the speed of said scrolling motion on said screen slows to a value below a predetermined given value.” The language of this claim is identical to claim 10 of the ’387 patent. Thus, for the reasons I discussed in Section IX(D) in connection with claim 10 of the ’387 patent, claim 9 of the ’064 patent is obvious over the combination of Korhonen with Korhonen and Narutaka, and separately over the combination of Korhonen with Korhonen, Narutaka, and Westerman.

E. Claims 2, 3, and 8 are Obvious Over Korhonen, Korhonen and Narutaka, Korhonen and Westerman, and Korhonen, Narutaka, and Westerman in Further Combination with Astala.

377. It is my opinion that claims 2, 3, and 8 of the ’064 patent would have been obvious to a POSITA in view of Korhonen and the combinations of Korho-

nen and Narutaka, Korhonen and Westerman, and Korhonen, Narutaka, and Westerman, further combined with Astala.

1. Dependent Claim 2

378. Claim 2 recites "[t]he improved touch-screen image scrolling system of claim 1, wherein said scrolling motion program instructions further comprise instructions to move said display in correspondence with movement of the finger touch, in response to movement following a touch having a stationary duration greater than said first preset given minimum time and less than a second given preset minimum time." The language of this claim is identical to claim 2 of the '387 patent. Thus, for the reasons I have discussed in Section IX(E) in connection with claim 2 of the '387 patent, claim 2 of the '064 patent is obvious over the combination of Astala with Korhonen, and separately over the combinations of Astala with Korhonen and Narutaka, Astala with Korhonen and Westerman, and Astala with Korhonen, Narutaka, and Westerman.

2. Dependent Claim 3

379. Claim 3 recites "[t]he improved touch-screen image scrolling system of claim 1, wherein said scrolling motion program instructions further comprise instructions to move a touch-selected item relative to the stationary display in correspondence with movement of said finger touch, in response to motion following a touch having a stationary duration greater than said second given preset mini-

mum time." The language of this claim is identical to claim 3 of the '387 patent. Thus, for the reasons I have discussed in Section IX(E) in connection with claim 3 of the '387 patent, claim 3 of the '064 patent is obvious over the combination of Astala with Korhonen, and separately over the combinations of Astala with Korhonen and Narutaka, Astala with Korhonen and Westerman, and Astala with Korhonen, Narutaka, and Westerman.

3. Independent Claim 8

380. The preamble of claim 8 reads: "An improved touch-screen image scrolling system, comprising: …." The language of this preamble is identical to the preamble of claim 8 of the '387 patent. Thus, for the reasons I have discussed in Section IX(E) in connection with element 8[pre] of the '387 patent, Korhonen discloses element 8[pre] of the '064 patent.

381. Element 8[a] recites "an electronic image display screen." The language of this element is identical to the element 8[a] of the '387 patent. Thus, for the reasons I have discussed in Section IX(E) in connection with element 8[a] of the '387 patent, Korhonen discloses element 8[a] of the '064 patent.

382. Element 8[b] recites "a microprocessor coupled to said display screen to display information thereon and to receive interactive signals therefrom." The language of this element is identical to element 8[b] of the '387 patent. Thus, for

the reasons I have discussed in Section IX(E) in connection with element 8[b] of the ’387 patent, Korhonen discloses element 8[b] of the ’064 patent.

383. Element 8[c] recites “timer means associated with said microprocessor to provide timing capacity therefor.” The language of this element is identical to element 8[c] of the ’387 patent. Thus, for the reasons I have discussed in Section IX(E) in connection with element 8[c] of the ’387 patent, Korhonen discloses element 8[c] of the ’064 patent and renders it obvious.

384. Element 8[d] recites “a source of scroll format data capable of display on said display screen.” The language of this element is identical to element 8[d] of the ’387 patent. Thus, for the reasons I have discussed in Section IX(E) in connection with element 8[d] of the ’387 patent, Korhonen discloses element 8[d] of the ’064 patent.

385. Element 8[e] recites “finger touch program instructions associated with said microprocessor for sensing the speed, direction and time duration of a finger touch contact with said display screen.” The language of this element is identical to element 8[f] of the ’387 patent. Thus, for the reasons I have discussed in Section IX(E) in connection with element 8[f] of the ’387 patent, Korhonen discloses element 8[e] of the ’064 patent and renders it obvious in combination with Narutaka.

386. Element 8[f] recites "scrolling motion program instructions associated with said microprocessor responsive to said duration of said finger touch contact such that, when said duration exceeds a first given preset minimum time, and is less than a second given preset minimum that is greater than said first minimum, and is accompanied by motion along the surface of said screen, a scroll format display on said screen is caused to begin to scroll in the sensed direction and at the sensed initial speed." The language of this element is identical to element 8[g] of the '387 patent. Thus, for the reasons I have discussed in Section IX(E) in connection with element 8[g] of the '387 patent, Korhonen renders element 8[f] of the '064 patent obvious in combination with Narutaka.

387. Element 8[g] recites "said scrolling motion program instructions further comprising instructions to move a touch-selected item relative to the stationary display in correspondence with movement of the finger touch, in response to motion following a touch having a stationary duration greater than said second given preset minimum time." The language of this element is identical to the limitation of claim 3 of the '387 patent. Thus, for the reasons I have discussed in Section IX(E) in connection with claim 3 of the '387 patent, Korhonen renders element 8[g] of the '064 patent obvious in combination with Astala and Narutaka, and separately in combination with Astala, Narutaka, and Westerman.

388. Element 8[h] discloses "said scrolling motion program instructions still further comprising instructions to move said display in correspondence with movement of the finger touch, in response to motion following a touch having a stationary duration greater than said first given preset minimum time and less than said second given preset minimum time." The language of this element is identical to element 8[h] of the '387 patent. Thus, for the reasons I have discussed in Section IX(E) in connection with element 8[h] of the '387 patent, Korhonen discloses element 8[h] of the '064 patent.

389. Element 8[i] discloses "time decay program instructions associated with said microprocessor for reducing the rate of scrolling displacement on said display screen at a given rate until motion is terminated." The language of this element is identical to element 8[i] of the '387 patent. Thus, for the reasons I have discussed in Section IX(E) in connection with element 8[i] of the '387 patent, Korhonen discloses element 8[i] of the '064 patent.

390. Element 8[j] recites "stopping motion program instructions associated with said microprocessor for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (b) an end-of-scroll signal received from said scroll format data source." The language of this element is identical to element

8[j] of the ’387 patent. Thus, for the reasons I have discussed in Section IX(E) in connection with element 8[j] of the ’387 patent, Korhonen renders element 8[j] of the ’064 patent obvious, alone or in combination with Westerman.

391. In sum, claim 8 is obvious over the combination of Astala with Korhonen, and separately over the combinations of Astala with Korhonen and Narutaka, Astala with Korhonen and Westerman, and Astala with Korhonen, Narutaka, and Westerman.

XI. CONCLUSION

392. For all of these reasons set forth in this Declaration, it is my opinion that all claims of the ’387 and ’064 patents are unpatentable.

I declare under penalty of perjury that the foregoing is true and correct to the best of my knowledge and that this Declaration was executed by me on the 5th day of October, 2017 at Minneapolis, MN.

Loren Terveen

APPENDIX A

APPENDIX A - CLAIM ELEMENT CHART FOR U.S. PATENT NO. 6,690,387 B2

’387 Patent Claim Language			
1[pre]. An improved touch-screen image scrolling system, comprising:	7[pre]. An improved touch-screen image scrolling system, comprising:	8[pre]. An improved touch-screen image scrolling system, comprising:	9[pre]. An improved method of controlling the scroll-like display of data on an electronic display screen, said method comprising the steps of:
1[a] an electronic image display screen;	7[a] an electronic image display screen;	8[a] an electronic image display screen;	
1[b] a microprocessor coupled to said display screen to display information thereon and to receive interactive signals therefrom;	7[b] a computer apparatus coupled to said display screen to display information thereon and to receive interactive signals therefrom;	8[b] a microprocessor coupled to said display screen to display information thereon and to receive interactive signals therefrom;	
1[c] timer means associated with said microprocessor to provide timing capacity therefor;	7[c] timer means within said computer apparatus to provide timing capacity therefor;	8[c] timer means associated with said microprocessor to provide timing capacity therefor;	
1[d] a source of scroll format data capable of display on said display screen;	7[d] said computer apparatus having capacity to store scroll format data capable of display on said display screen;	8[d] a source of scroll format data capable of display on said display screen;	

’387 Patent Claim Language			
1[e] a keyboard coupled to said microprocessor to provide input control signals thereto;	7[e] a keyboard coupled to said computer apparatus to provide input control signals thereto;	8[e] a keyboard coupled to said microprocessor to provide input control signals thereto;	
1[f] finger touch program instructions associated with said microprocessor for sensing the speed, direction and time duration of a finger touch contact with said display screen;	7[f] finger touch program instructions associated with said computer apparatus for sensing the speed, direction and time duration of a finger touch contact with said display screen;	8[f] finger touch program instructions associated with said microprocessor for sensing the speed, direction and time duration of a finger touch contact with said display screen;	9[a] sensing the duration of finger touch contact time with an electronic display screen having scrollable data displayed thereon; 9[b] sensing the speed and direction of motion of said finger touch contact with said display screen;
1[g] scrolling motion program instructions associated with said microprocessor responsive to said duration of said finger touch contact such that, when said duration exceeds a first given preset minimum time and is accompanied by motion	7[g] scrolling motion program instructions associated with said computer apparatus responsive to said duration of said finger touch contact such that, when said duration exceeds a preset minimum time and is accompanied by motion	8[g] scrolling motion program instructions associated with said microprocessor responsive to said duration of said finger touch contact such that, when said duration exceeds a first given preset minimum time, and is less than a second given	9[c] initiating scrolling motion of said scrollable data on said display screen in said sensed direction and at said sensed speed;

’387 Patent Claim Language			
along the surface of said screen followed by separation of said finger touch from said screen, a scroll format display on said screen is caused to begin to scroll in said sensed direction and at said sensed initial speed;	along the surface of said screen, a scroll format display on said screen is caused to begin to scroll in the sensed direction and at the sensed initial speed;	preset minimum that is greater than said first minimum, and is accompanied by motion along the surface of said screen, a scroll format display on said screen is caused to begin to scroll in the sensed direction and at the sensed initial speed;	
		8[h] said scrolling motion program instructions still further comprising instructions to move said display in correspondence with movement of the finger touch, in response to motion following a touch having a stationary duration greater than said first given preset minimum time and less than said second given preset minimum time;	
1[h] time decay program	7[h] time decay program	8[i] time decay program	9[d] slowing the speed of

’387 Patent Claim Language			
instructions associated with said microprocessor for reducing the rate of scrolling displacement on said display screen at a given rate until motion is terminated;	instructions associated with said computer apparatus for reducing the rate of scrolling displacement on said display screen at a given rate until motion is terminated;	instructions associated with said microprocessor for reducing the rate of scrolling displacement on said display screen at a given rate until motion is terminated;	said scrolling motion from the initiated speed thereof, at a predetermined rate;
1[i] stopping motion program instructions associated with said microprocessor for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (b) an end-of-scroll signal received from said scroll format data	7[i] stopping motion program instructions associated with said computer apparatus for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (b) an end-of-scroll signal received from said scroll	8[j] stopping motion program instructions associated with said microprocessor for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a preset minimum time, and (b) an end-of-scroll signal received from said scroll	9[e] and terminating said scrolling motion when one of the conditions comprising the following group of conditions is sensed: (a) a substantially stationary finger touch having a finite duration is sensed; (b) an end-of-scroll signal is sensed.

’387 Patent Claim Language			
source.	format data source.	format data source.	
2. The improved touch-screen image scrolling system of claim 1, wherein said scrolling motion program instructions further comprise instructions to move said display in correspondence with movement of the finger touch, in response to movement following a touch having a stationary duration greater than said first preset given minimum time and less than a second given preset minimum time.			11. The improved method of controlling the scroll-like display of data on an electronic display screen in accordance with claim 9, wherein said method comprises the further step of sensing a finger touch on said screen having a duration greater than said first given preset minimum time and less than a second given preset minimum time which is greater than said first given time and then moving said display in correspondence with movement of the finger touch.
3. The improved touch-screen image scrolling system of claim 1, wherein said scrolling motion			12. The improved method of controlling the scroll-like display of data on an electronic display screen.

’387 Patent Claim Language			
program instructions further comprise instructions to move a touch-selected item relative to the stationary display in correspondence with movement of said finger touch, in response to motion following a touch having a stationary duration greater than said second given preset minimum time.			in accordance with claim 9, wherein said method comprises the further step of sensing a stationary finger touch on said screen having a duration greater than a second preset given minimum time which is greater than said first given preset time and then moving a touch-selected item relative to the stationary display in correspondence with movement of the finger touch.
4. The improved touch-screen image scrolling system of claim 1, wherein said group of signals for terminating scrolling, displacement of the image on said display screen further comprises (a) a signal indicating that the	10. The improved method of controlling the scroll-like display of data on an electronic display screen, in accordance with claim 7, wherein said group of conditions to be sensed for terminating said scrolling motion further comprises:		

’387 Patent Claim Language			
rate of scrolling displacement on said screen has decayed to a value below a predetermined given value.	the speed of said scrolling motion on said screen slows to a value below a predetermined given value.		
5. The improved touch-screen image scrolling system of claim 1, wherein said microprocessor, and said timer means together comprise a processing unit of a conventional computer.			
6. The improved touch-screen image scrolling system of claim 5, wherein said source of scroll format data capable of display on said display screen comprises part of the memory of said conventional computer.			

APPENDIX B

Appendix B - Claim Element Chart for U.S. Patent No. 7,184,064 B2

’064 Patent Claim Language		
1[pre]. An improved touch-screen image scrolling system, comprising:	7[pre]. An improved touch-screen image scrolling system, comprising:	8[pre]. An improved touch-screen image scrolling system, comprising:
1[a] an electronic image display screen;	7[a] an electronic image display screen;	8[a] an electronic image display screen;
1[b] a microprocessor coupled to said display screen to display information thereon and to receive interactive signals therefrom;	7[b] a computer apparatus coupled to said display screen to display information thereon and to receive interactive signals therefrom;	8[b] a microprocessor coupled to said display screen to display information thereon and to receive interactive signals therefrom;
1[c] timer means associated with said microprocessor to provide timing capacity therefor;	7[c] timer means within said computer apparatus to provide timing capacity therefor;	8[c] timer means associated with said microprocessor to provide timing capacity therefor;
1[d] a source of scroll format data capable of display on said display screen;	7[d] said computer apparatus having capacity to store scroll format data capable of display on said display screen;	8[d] a source of scroll format data capable of display on said display screen;
1[e] finger touch program instructions associated with said microprocessor for sensing the speed, direction and time duration of a finger touch contact with said	7[e] finger touch program instructions associated with said computer apparatus for sensing the speed, direction and time duration of a finger touch contact with said	8[e] finger touch program instructions associated with said microprocessor for sensing the speed, direction and time duration of a finger touch contact with said

’064 Patent Claim Language		
display screen:	display screen;	display screen:
1[f] scrolling motion program instructions associated with said microprocessor responsive to said duration of said finger touch contact such that, when said duration exceeds a first given preset minimum time and is accompanied by motion along the surface of said screen followed by separation of said finger touch from said screen, a scroll format display on said screen is caused to begin to scroll in said sensed direction and at said sensed initial speed;	7[f] scrolling motion program instructions associated with said computer apparatus responsive to said duration of said finger touch contact such that, when said duration exceeds a preset minimum time and is accompanied by motion along the surface of said screen, a scroll format display on said screen is caused to begin to scroll in the sensed direction and at the sensed initial speed;	8[f] scrolling motion program instructions associated with said microprocessor responsive to said duration of said finger touch contact such that, when said duration exceeds a first given preset minimum time, and is less than a second given preset minimum that is greater than said first minimum, and is accompanied by motion along the surface of said screen, a scroll format display on said screen is caused to begin to scroll in the sensed direction and at the sensed initial speed;
		8[g] said scrolling motion program instructions further comprising instructions to move a touch-selected item relative to the stationary display in correspondence with movement of the finger touch, in response to motion following a touch having a stationary duration greater than said second given preset minimum time;

’064 Patent Claim Language		
		8[h] said scrolling motion program instructions still further comprising instructions to move said display in correspondence with movement of the finger touch, in response to motion following a touch having a stationary duration greater than said first given preset minimum time and less than said second given preset minimum time;
1[g] time decay program instructions associated with said microprocessor for reducing the rate of scrolling displacement on said display screen at a given rate until motion is terminated;	7[g] time decay program instructions associated with said computer apparatus for reducing the rate of scrolling displacement on said display screen at a given rate until motion is terminated;	8[i] time decay program instructions associated with said microprocessor for reducing the rate of scrolling displacement on said display screen at a given rate until motion is terminated;
1[h] stopping motion program instructions associated with said microprocessor for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a	7[h] stopping motion program instructions associated with said computer apparatus for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a	8[j] stopping motion program instructions associated with said microprocessor for terminating scrolling displacement of the image on said screen upon first occurrence of any signal in the group of signals comprising: (a) a substantially stationary finger touch on the screen enduring for a period longer than a

’064 Patent Claim Language		
preset minimum time, and (b) an end-of-scroll signal received from said scroll format data source.	preset minimum time, and (b) an end-of-scroll signal received from said scroll format data source.	preset minimum time, and (b) an end-of-scroll signal received from said scroll format data source.
2. The improved touch-screen image scrolling system of claim 1, wherein said scrolling motion program instructions further comprise instructions to move said display in correspondence with movement of the finger touch, in response to movement following a touch having a stationary duration greater than said first preset given minimum time and less than a second given preset minimum time.		
3. The improved touch-screen image scrolling system of claim 1, wherein said scrolling motion program instructions further comprise instructions to move a touch-selected item relative to the stationary display in correspondence with movement of said finger touch, in response to motion following a touch having a stationary duration		

’064 Patent Claim Language		
greater than said second given preset minimum time.		
4. The improved touch-screen image scrolling system of claim 1, wherein said group of signals for terminating scrolling displacement of the image on said display screen further comprises (a) a signal indicating that the rate of scrolling displacement on said screen has decayed to a value below a predetermined given value.	9. The improved method of controlling the scroll-like display of data on an electronic display screen, in accordance with claim 7, wherein said group of conditions to be sensed for terminating said scrolling motion further comprises: the speed of said scrolling motion on said screen slows to a value below a predetermined given value.	
5. The improved touch-screen image scrolling system of claim 1, wherein said microprocessor, and said timer means together comprise a processing unit of a conventional computer.		
6. The improved touch-screen image scrolling system of claim 5, wherein said source of scroll format data capable of display on said display screen comprises part of the memory of said conventional computer.		

APPENDIX C

Loren G. Terveen

terveen@cs.umn.edu University of Minnesota (612) 624-8310

Address

Department of Computer Science and Engineering
University of Minnesota
Minneapolis, MN 55455

Education

Ph.D. in Computer Sciences	University of Texas	1991
M.S. in Computer Sciences	University of Texas	1988
B.A. in Computer Science, Mathematics, History	University of South Dakota	1984

Affiliations Member of ACM, ACM/SIGCHI

Research Interests Human-Computer Interaction, Computer-Supported Collaborative Work

Professional Experience

Professor	University of Minnesota	2002 --
Distinguished McKnight University Professor		2017 --
Principal Member of Technical Staff	AT&T Labs - Research	1996-2002
Member of Technical Staff	AT&T Bell Labs	1991-1996
Graduate Research Intern	Microelectronics and Computer Corporation	1986-19991
Teaching Assistant	University of Texas at Austin	1985

AWARDS AND MAJOR ROLES

ACM Distinguished Scientist (awarded 2009)

ACM SIGCHI President (2015-2018)

ACM Council, SIG Governing Board Representative (2016 –)

RESEARCH

External Funding

NSF: “PFI:BIC: Smart Human-Centered Collision Warning System: sensors, intelligent algorithms and human-computer interfaces for safe and minimally intrusive car-bicycle interactions” (Co-PI, with Rajesh Rajamani (PI), Max Donath, and Nichole Morris), $999,773 for September 1, 2016 to August 31, 2019.

NSF: “Computer-Supported Cooperative Work Doctoral Colloquium” (**PI**), $25,000 for 03/01/2016 to 02/28/2017.

National Institute on Drug Abuse: “A Technology-Delivered Peer-to-Peer Support ARB Adherence Intervention for HIV+ Adults”, (Co-PI, with Keith Horvath(PI) and Darin Erickson), $3,302,62 for 07/01-2015 to 05/31/2020.

NSF: “HCC: Tools and Mechanisms to Support Social Participation Efforts”, (**PI)**, $499,399 for 10/01/2012 to 09/30/2015.

NSF: “SoCS: Collaborative Research: Novel Algorithms and Interaction Mechanisms to Enhance Social Production”, (**PI**), $527,140, for 7/01/2012 to 06/30/2015 (recommended for funding).

NSF: “Collaborative Research: Supporting Newcomer Socialization in Online Production Communities”, (**PI**), $301,135.00 for 08/2011 to 08/31/2015.

Minnesota Department of Transportation: “Statewide Cycloplan: A Bicycle Planning Tool with Participatory GIS”, (**PI**), $130,000, for 10/01/2011 to 06/30/2013.

Metropolitan Council: “Cycloplan II”, (**PI**), $71,350, for 08/15/2011 to 05/31/2012.

NSF: “Wikisym Doctoral Consortium”, (**PI**), $13,163, for 05/01/2011 to 04/30/2012.

IBM: “Mobile Crowdsensing”, (**PI**), $100,000 awarded 04/01/2011

NSF: “Social-Computational Systems (SoCS) Community Meeting” (**PI**), $48,801 for 09/01/2010 to 08/31/2011

NSF: “SoCS: Collaborative Research: Information Framing: Intelligent Interfaces for an Online Production Community”, (**PI)**, $375,000 for 09/15/2010 to 08/31/2015.

NSF: “Collaborative Research: Guiding Folksonomy Development to Enable Novel Tagging Applications” (**PI,** with J. Riedl and S. Sen (Macalester College)). $949, 788 for 04/1/2010 to 03/312014 (UMN Share).

Minnesota Department of Transportation: Bike, Bus, and Beyond: Extending Cyclopath to Enable MultiModal Routing (**PI**), $60,627, for 07/08/2010 to 01/31/2012

NIH: An Interactive Website to Promote Communication about Sexual Health and Dating Relationships between Parents and Teens (**Co-PI**, with Sonya Brady (PI), Simon Rosser, and Renee Sieving), $679,500 for 09/30/2009 to 08/31/2011.

Metropolitan Council: “Cycloplan” (**PI**), $185,000, October 2009 – March 2011.

NSF “Collaborative Research: Understanding Online Volunteer Communities: Toward Theory-Based Design” (**co-PI,** with J. Riedl, J. Konstan, M. Snyder, & Y. Ren; R. Kraut (CMU) $2,400,000 for 08/01/2008 to 07/31/2013.

NSF: “Recommender Systems Doctoral Consortium” (**PI**) $15,415, 2007-2008.

NSF: “Mining Spatiotemporal Data: From Personal Use to Community Knowledge” (**PI**) $449,570 for 12/1/2005 to 11/30/2009.

NSF: “Collaborative Research: Mark This! - Operationalizing the notion of "place" for interactive community systems” (**PI**, with Q. Jones (NJIT) and S. Whittaker (Univ. of Sheffield)). $173,411 for 6/1/2003 to 5/31/2007 (UMN share).

NSF: “Being There: Mobile Devices for Community and Commerce” (**PI**, with J. Konstan, J. Riedl, and S. Shekhar). $120,000 for 9/1/2002 to 8/31/2005.

NSF: “ITR: Collaborative Research: Designing On-Line Communities to Enhance Participation” (**co-PI**, with J. Konstan & J. Riedl, R. Kraut & S. Kiesler (CMU), P. Resnick and Y. Chen (Univ. of Michigan)). $1,246,017 for 9/1/2003 to 8/31/2009 (UMN share).

AT&T: “VURI: Collaborative Filtering and Intelligent Interface Design for Enhanced TV Applications” (**PI**). $35,000 for November 1, 2008 to October 31, 2009.

AT&T: “VURI: Collaborative Filtering and Intelligent Interface Design for Enhanced TV Applications” (**PI**). $35,000 for June 1, 2007 to May 31, 2008.

Internal Funding

Minnesota/China Collaborative Research Grant: "Expertise Oriented Mining for Web Community". $10,000 for July 1, 2007 to June 30, 2008 (PI, with Jie Tang, Tsinghua University, Beijing China).

University of Minnesota TEL grant: “The Next Generation Online Learning Environment: Designing for Community and Collaboration”. $10,00 for September 2006 to May 2007. (co-PI, with Joan Hughes, David Ernst, and Ann Ooms, College of Education and Human Development).

University of Minnesota Digital Technology Center: “Indoor Navigation Aids for Visually

Impaired People". $25,67 for June 2005 to December 2006. (co-PI, with S. Shekhar and G. Legge).

University of Minnesota Digital Technology Center: "Eye-Tracking Research on Community Websites: Photo Directories and Building Social Networks". $19,300 for June 2004 to June 2005. (co-PI, with J. Konstan).

University of Minnesota Grant-In-Aid: "Facilitating Participation in Online Communities". $20,397 for 1/1/2003 to 6/30/2004.

Books

1. Bickhard, M.H. and Terveen, L.G. Foundational Issues in Artificial Intelligence and Cognitive Science: Impasse and Solution, (1995), Elsevier Science.

Refereed Journal Papers

2. Nguyen, T.T., Harper, F.M., Terveen, L., and Konstan, J. User Personality and User Satisfaction with Recommender Systems. In *Information Systems Frontiers* (2017).
3. Filson Moses, J., Dwyer, P.C., Gulestad, P.T., Kim, J.S., Maki, A., Synder, M., and Terveen, L. Encouraging Online Engagement: The Role of Interdependent Self-Construal and Social Motives in Fostering Online Participation. In *Personality and Individual Differences* (2017).
4. Thebault-Spieker, J., Terveen, L., and Hecht, B. Towards a Geographic Understanding of the Sharing economy: Systemic Biases in UberX and TaskRabbit. In *ACM Transactions on Computer-Human Interaction* (2017).
5. Brady, S.S., Sieving, R.E., Terveen, L.G., Rosser, R. S., Kodet, A.J., and Rothberg, V.D. An Interactive Website to Reduce Sexual Risk Behavior: Process Evaluation of TeensTalkHealth, JMIR Research Protocols (2015).
6. Ren, Y., Harper, F.M., Drenner, S., Terveen, L., Kiesler, S., Riedl, J., and Kraut, R.E. (2012). Building Member Attachment in Online Communities: Applying Theories of Group Identity and Interpersonal Bonds. *Management Information Systems Quarterly*.
7. Jones, Q., Grandhi, S., Karam, S., Whittaker, S., Zhou, C., and Terveen, L. Geographic 'Place' and Community Information Preferences, in *Computer-Supported Cooperative Work*.
8. Zhou, C., Frankowski, D., Ludford, P., Shekhar, S., Terveen, L., Discovering Personally Meaningful Places from Location Data: An Interactive Clustering Approach. *ACM Transactions on Information System,* 25, 3 (July 2007).
9. Ling, K., Beenen, G., Ludford, P.J., Wang, X., Chang, K., Li, X., Cosley, D., Frankowski, D., Terveen, L., Rashid, A.M., Resnick, P., and Kraut, R.E. Using Psychology to Motivate Contributions to Online Communities. *Journal of Computer-Mediated Communication*, 10, 4 (June 2005).
10. Terveen, L. and McDonald, D. Social Matching: A Framework and Research Agenda. *ACM Transactions on Computer-Human Interaction*, 12, 3 (2005), 401-434.
11. Whittaker, S., Jones, Q., Nardi, B., Creech, M., Terveen, L., Isaacs, E., and Hainsworth, J. ContactMap: organizing communication in a social desktop, in *ACM Transactions on Computer-Human Interaction*, 11, 4 (December 2004), 445-471.
12. Jones, Q., Grandhi, S., Terveen, L., and Whittaker, S. People-To-People-to-Geographical-Places: The P3 Framework for Location-Based Community Systems. in *Computer-Supported Cooperative Work*, 13, 3-4 (August 2004), 249-282.

13. Herlocker, J.L., Konstan, J.A., Terveen, L.G., and Riedl, J.T. Evaluating Collaborative Filtering Recommender Systems, *ACM Transactions on Information Systems* (2004).
14. Amento, B., Terveen, L., Hill, W., Hix, D., and Schulman, R. Experiments in Social Data Mining: The TopicShop System, in *ACM Transactions on Computer-Human Interaction*, 10, 1 (March 2003), 54-85.
15. Whittaker, S., Terveen, L.G, and Nardi, B.A. Let's stop pushing the envelope and start addressing it, in *Human-Computer Interaction*, 15, 2-3 (Sep 2000), 75-106.
16. Terveen, L.G., Hill, W.C., and Amento, B. Constructing, Organizing, and Visualizing Collections of Topically Related Web Resources, in *ACM Transactions on Computer-Human Interaction*, 6, 1 (Mar. 1999), 67-94.
17. Selfridge, P.G. and Terveen, L.G. Knowledge Management Tools for Business Process Support and Reengineering, in *Journal of Intelligent Systems in Accounting, Finance, and Management* (Jan. 1996).
18. Terveen, L.G. An Overview of Human-Computer Collaboration, in *Knowledge-Based Systems*, 8, 2-3 (1995), 67-81.
19. Terveen, L.G., Selfridge, P.G., and Long, M.D. Living Design Memory: Framework, System, and Lessons Learned, in *Human-Computer Interaction*, 10, 1 (1995), 1-37.
20. Terveen, L.G. Intelligent Systems as Cooperative Systems, in *International Journal of Intelligent Systems*, 3, 2-4 (1993), 217-250.
21. Brachman, R.J., Selfridge, P.G., Terveen, L.G., Altman, B., Borgida, A., Halper, F., Kirk, T., Lazar, A., McGuinness, D.L., and Resnick, L.A. Integrated Support for Data Archaeology, in *International Journal of Intelligent and Cooperative Information Systems*, 2, 2 (1993), 159-185.

Refereed Conference Papers

22. Kang, J., Condiff, K., Chang, S., Terveen, L., Konstan, J., and Harper, F.M. Understanding How People Use Natural Language to Ask for Recommendations, in *RecSys 2017.*
23. Miller, H., Kluver, D., Thebault-Spieker, J., Terveen, L., and Hecht, B. Understanding Emoji Ambiguity in Context: The Role of Text in Emoji-Related Miscommunication. In *Proceedings of ICWSM 2017, AAAI Conference on the Web and Social Media.*
24. Hall, A., McRoberts, S., Thebault-Spieker, J., Lin, A.Y., Sen, S., Hecht, B., and Terveen, L. Freedom versus Standardization: Structured Data Generation in a Peer Production Community. In *Proceedings of CHI 2017, the ACM Conference on Human Factors in Computing Systems (CHI 2017)*.
25. Yu, B., Ren, Y., Terveen, L., and Zhu, H. Predicting Member Productivity and Withdrawal from Pre-Joining Attachments in Online Production Groups. In *Proceedings of CSCW 2017, The ACM Conference on Computer-Supported Cooperative Work and Social Computing.*
26. Chang, S., Harper, M., and Terveen, L Crowd-Based Personalized Natural Language Explanations for Recommendations In *Proceedings of RecSys 2016, The ACM Conference on Recommender Systems.*
27. Chang, S., Harper, M., He, L., and Terveen, L CrowdLens: Experimenting with Crowd-Powered Recommendation and Explanation. In *Proceedings of ICWSM 2016, AAAI Conference on the Web and Social Media.*

28. Miller, H., Thebault-Spieker, J., Chang, S., Johnson, I., Terveen, L., and Hecht, B. "Blissfully happy" or "ready to fight": Varying Interpretations of Emoji. In *Proceedings of ICWSM 2016, AAAI Conference Web and Social Media.*
29. Zhao, Q., Huang, Z., Harper, F.M., Terveen, L., and Konstan, J. "Precision Crowdsourcing: Closing the Loop to Turn Information Consumers into Information Producers". In *Proceedings of CSCW 2016, The ACM Conference on Computer Supported Cooperative Work and Social Computing.*
30. Harper, F.M., Xu, F., Kaur, H., Condiff, K., Chang, S., and Terveen, L. Putting Users in Control of their Recommendations. In *Proceedings of RecSys2015, The ACM Conference on Recommender Systems.*
31. Kapoor, K., Kumar, V., Terveen, L., Konstan, J.A., and Schrater, P. "I like to explore sometimes" – Adapting to Dynamic User Novelty Preferences. In *Proceedings of RecSys2015, The ACM Conference on Recommender Systems.*
32. Warncke-Wang, M., Ranjan, V., Terveen, L., and Hecht, B. Misalignment Between Supply and Demand of Quality Content in Peer Production Communities. In *Proceedings of the International Conference on Weblogs and Social Media (ICWSM'15).*
33. Chang, S., Harper, F.M., and Terveen, L. Using Groups of Items to Bootstrap New Users in Recommender Systems. In *Proceedings of the 18th ACM conference on Computer supported cooperative work & social computing* (CSCW '15). ACM, New York, NY, USA.
34. Miller, H., Chang, S., and Terveen, L. "I LOVE THIS SITE!" vs. "It's a little girly": Perceptions of and Initial User Experience with Pinterest. In *Proceedings of the 18th ACM conference on Computer supported cooperative work & social computing* (CSCW '15). ACM, New York, NY, USA.
35. #Thebault-Spieker, J., Terveen, L., and Hecht, B. 2015. Avoiding the South Side and the Suburbs: The Geography of Mobile Crowdsourcing Markets. In *Proceedings of the 18th ACM conference on Computer supported cooperative work & social computing* (CSCW '15). ACM, New York, NY, USA.
36. Warncke-Wang, M., Hecht, B., and Terveen, L. The Success and Failure of Quality Improvement Projects in Peer Production Communities. In *Proceedings of the 18th ACM conference on Computer supported cooperative work & social computing* (CSCW '15). ACM, New York, NY, USA.
37. Panciera, K., Masli, M., and Terveen, L. Crème de la Crème: Elite contributors in an online community, in *Proceedings of the 10th International Symposium on Open Collaboration* (OpenSym 2014).
38. Kumar, V., Kluver, D., Terveen, L., and Riedl, J. More Efficient Tagging Systems with Tag Seeding, in Proceedings of IEEE SocialCom 2014.
39. Nguyen, T., Hui, P., Harper, F.M., Terveen, L., and Konstan, J. Exploring the Filter Bubble: The Effect of Using Recommender Systems on Content Diversity, in Proceedings of WWW 2014.
40. Halfaker, A., Geiger, S., and Terveen, L. Snuggle: Designing for Efficient Socialization and Ideological Critique, in Proceedings of CHI 2014.
41. Chang, S., Kumar, V., Gilbert, E., and Terveen, L. Specialization, Homophily, and Gender in a Social Curation Site Findings from Pinterest, in Proceedings of CSCW 2014.
42. Grevet, C., Terveen, L., and Gilbert, E. Managing Political Differences in Social Media, in Proceedings of CSCW 2014.

43. Masli, M., and Terveen, L. Leveraging the Contributory Potential of User Feedback, in Proceedings of CSCW 2014.
44. Sheppard, S.A., Wiggins, A., and Terveen, L. Capturing Quality: Retaining Provenance for Curated Volunteer Monitoring Data, in Proceedings of CSCW 2014.
45. Delong, C., Terveen, L., and Srivastava, J. (2013). TeamSkill and the NBA: Applying Lessons from the Virtual World to the Real World, in Proceedings of ASONAM 2013.
46. Torre, F., Liu, Y., Liu, Z., and Terveen, L. (2013). Local Knowledge Matters for Crowdsourcing Systems: Experience from Transferring an American Site to China, in Proceedings of ICWSM 2013.
47. Gilbert, E., Bakshi, S, Chang, S., and Terveen, L. (2013). "I Need to Try This!": A Statistical Overview of Pinterest, in Proceedings of CHI 2013.
48. Torre, F., Pitchford, D., Brown, P., and Terveen, L. (2012). Matching GPS Traces to (Possibly) Incomplete Map Data: Bridging Map Building and Map Matching, in ACM SIGSPATIAL GIS 2012.
49. Dunne, L, Zhang, J., and Terveen, L. (2012). An Investigation of Contents and Use of the Home Wardrobe, in UbiComp 2012.
50. Dong, Z., Shi, C., Sen, S., Terveen, L., and Riedl, J. (2012). War Versus Inspirational in Forrest Gump: Cultural Effects in Tagging Communities, in ICWSM 2012.
51. Masli, M. and Terveen, L. (2012) Evaluating Compliance-Without-Pressure Techniques for Increasing Participation in Online Communities, in CHI 2012.
52. Fuglestad P.T., Dwyer, P.C., Filson Moses, J., Kim, J.S., Mannino, C.A., Terveen, L., and Snyder, M. (2012) What Makes Users Rate (Share, Tag, Edit…)? Predicting Patterns of Participation in Online Communities, in CSCW 2012.
53. Nathan, M., Topkara, M., Lai, J., Pan, S., Wood, S., Boston, J., and Terveen, L. (2012) In Case You Missed It: Benefits of Attendee-Shared Annotations for Non-attendees of Remote Meetings, in CSCW 2012.
54. Priedhorsky, R., Pitchford, D., Sen, S., and Terveen, L. (2012), Recommending Routes in the Context of Bicycling: Algorithms, Evaluation, and the Value of Personalization, in CSCW 2012.
55. Lam, S.K., Uduwage, A., Dong, Z., Sen, S., Musicant, D.R., Terveen, L., and Riedl, J. (2011). WP:Clubhouse? An Exploration of Wikipedia's Gender Imbalance, in Wikisym 2011. **Best Paper Winner.**
56. Panciera, K., Masli, M., and Terveen, L.G. (2011). "How Should I Go from __ to __ without Getting Killed? Motivations and Benefits in Open Collaboration", in Wikisym 2011.
57. Priedhorsky, R., and Terveen, L.G. (2011). Wiki Grows Up: Arbitrary Data Models, Access Control, and Beyond, in Wikisym 2011.
58. Sheppard, S.A. and Terveen, L.G. (2011). Quality is a Verb: The Operationalization of Data Quality in a Citizen Science Community, in Wikisym 2011.
59. Masli, M., Priedhorsky, R., and Terveen, L. (2011). Task Specialization in Social Production Communities: The Case of Geographic Volunteer Work, in the Proceedings the 4th International AAAI Conference on Weblogs and Social Media (ICWSM 2011).
60. Torre, F., Sheppard, S.A., Priedhorsky, R., and Terveen, L. (2010) bumpy, caution with merging: An Exploration of Tagging in a Geowiki, in GROUP 2010.
61. Panciera, K., Priedhorsky, R., Erickson, T., and Terveen, L. (2010). Lurking? Cyclopaths? A Quantitative Analysis of User Behavior in a Geowiki, in CHI 2010. **Best of CHI Nominee**.

62. Priedhorsky, R., Masli, M., and Terveen, L. (2010). Eliciting and Focusing Geographic Volunteer Work, in CSCW 2010.
63. Panciera, K., Halfaker, A., and Terveen, L. (2009). Wikipedians are Born, Not Made: A Study of Power Editors on Wikipedia, in GROUP 2009. *36% Acceptance Rate.*
64. Reily, K., Ludford Finnerty, P., and Terveen, L. Two Peers are Better than One: Aggregating Peer Reviews for Computing Assignments is Surprisingly Accurate, in GROUP 2009. *36% Acceptance Rate.*
65. Ludwig, M., Priedhorsky, R., and Terveen, L. (2009). Path Selection: A Novel Interaction Technique for Mapping Applications, to appear in CHI 2009. *24% acceptance rate*.
66. Priedhorsky, R., and Terveen, L. (2008). The Computational Geowiki: What, Why, and How, in CSCW 2008. *23% acceptance rate*. **Best of CSCW Nominee**.
67. Drenner, S., Sen, S., and Terveen, L (2008). Crafting the Initial User Experience to Achieve Community Goals, in RecSys 2008. *30% acceptance rate*.
68. Reily, K., Ludford, P., and Terveen, L. (2008). Sharescape: An Interface for Place Annotation, in NordiCHI 2008. *30% acceptance rate*.
69. Nathan, M., Harrison, C., Yarosh, S., Terveen, L., Stead, L., and Amento, B. (2008), CollaboraTV: Making Television Viewing Social Again, in uxTV 2008.
70. Priedhorsky, R., Jordan, B., and Terveen, L. (2007), How a Personalized Geowiki Can Help Bicyclists Share Information More Effectively, in WikiSym 2007. 50% acceptance rate.
71. Priedhorsky, R., Chen, J., Lam, A., Panciera, K., Terveen, L., and Riedl, J. (2007), Creating, Destroying, and Restoring Value in Wikipedia, in Group 2007.
72. Rouben, A. and Terveen, L. (2007), Speech and Non-Speech Audio: Navigational Information and Cognitive Load, in International Conference on Auditory Displays (ICAD).
73. Ludford, P., Priedhorsky, R., Reily, K., and Terveen, L. (2007), Capturing, Sharing, and Using Local Place Information, in CHI 2007. *25% acceptance rate*.
74. Cosley, D., Frankowski, D., Terveen, L., and Riedl, J. (2007), SuggestBot: Using Intelligent Task Routing to Help People Find Work in Wikipedia, in IUI 2007. *22% acceptance rate*.
75. Harper, F.M., Frankowski, D., Drenner, S., Ren, Y., Kiesler, S., Terveen, L., Kraut, R., and Riedl, J. (2007), Talk Amongst Yourselves: Inviting Users To Participate In Online Conversations, in IUI 2007. *22% acceptance rate*.
76. Frankowski, D., Cosley, D., Sen, S., Terveen, L., and Riedl, J. You Are What You Say: Privacy Risks of Public Mentions, in *SIGIR 2006*. *19% acceptance rate.*
77. Ludford, P.J., Frankowski, D., Reily, K., Wilms, K., and Terveen, L., Because I Carry My Cell Phone Anyway: Functional Location-Based Reminder Applications, in *Proceedings of CHI 2006*. *23% acceptance rate*.
78. Cosley, D., Frankowski, D., Terveen, L, and Riedl, J., Using Intelligent Task Routing and Contribution Review to Help Communities Build Artifacts of Lasting Value, in *Proceedings of CHI 2006*. *23% acceptance rate*.
79. Drenner, S., Harper, M., Frankowski, D., Riedl, J., and Terveen, L. Insert Movie Reference Here: A System to Bridge Conversation and ItemOriented Web Sites, in *Proceedings of CHI 2006* (Tech Note).
80. Zhou, C., Ludford, P., Frankowski, D., and Terveen, L. How Do People's Concepts of Place Relate to Physical Locations? In *Proceedings of INTERACT 2005*. *27% acceptance rate.*
81. D. Cosley, D. Frankowski, S. Kiesler, L. Terveen, J. Riedl. How Oversight Improves Member-Maintained Communities. In *Proceedings of CHI 2005*, Portland, OR, 2005. *25% acceptance rate*.

82. Jones, Q., Grandhi, S., Whittaker, S., Chivakula, K, and Terveen, L. Putting Systems into Place: A Qualitative Study of Design Requirements for Location Aware Community Systems, in *Proceedings of CSCW 2004. 30% acceptance rate*.
83. Zhou, C., Ludford, P., Shekhar, S., and Terveen, L. Discovering Personal Gazetteers: An Interactive Clustering Approach, in *ACM GIS 2004* (12th International Symposium on Geographic Information Systems). *33% acceptance rate.*
84. Ludford, P., Cosley, D., Frankowski, D., and Terveen, L.G. Think Different: Increasing Online Community Participation Using Uniqueness and Group Dissimilarity, in *Proceedings of CHI 2004. 16% acceptance rate*.
85. Cosley, D., Ludford, P. and Terveen, L.G. Studying the Effect of Similarity in Online Task-Focused Interactions, *Proceedings of GROUP 2003. 35% acceptance rate*.
86. Ludford, P. and Terveen, L.G. Does an Individual's Myers-Briggs Type Indicator Preference Influence Task-Oriented Technology Use?, *Proceedings of Interact 2003. 34% acceptance rate*.
87. Whittaker, S., Jones, Q., and Terveen, L.G. Contact Management: Identifying Contacts to Support Long-Term Communication, *Proceedings of CSCW 2002*, 216-225. *20% acceptance rate*.
88. Terveen, L.G., McMackin, J., Amento, B., and Hill, W. Specifying Preferences Based On User History, *Proceedings of CHI 2002*, 315-322. *15% acceptance rate.*
89. Whittaker, S., Jones, Q., and Terveen, L.G. Managing Long Term Conversations: Conversation and Contact Management, *Proceedings of HICSS 2002. 50% acceptance rate.*
90. Amento, B., Terveen, L., Hill, W., and Hix, D. TopicShop: Enhanced Support for Evaluating and Organizing Collections of Web Sites, *Proceedings of UIST 2000. 26% acceptance rate.*
91. #Amento, B., Terveen, L., and Hill, W. Does 'Authority' Mean Quality? Predicting Expert Quality Ratings of Web Documents, *Proceedings of SIGIR 2000. 27% acceptance rate*.
92. Amento, B., Hill, W., Terveen, L., Hix, D., and Ju, P. An Empirical Evaluation of User Interfaces for Topic Management of Web Sites, *Proceedings of CHI 1999*, 552-559. *25% acceptance rate.*
93. Terveen, L.G and Hill, W.C. Evaluating Emergent Collaboration on the Web, *Proceedings of CSCW 1998*, 355-362. *19% acceptance rate.*
94. Whittaker, S., Terveen, L.G, Hill, W.C., and Cherny, L. The Dynamics of Mass Interaction, *Proceedings of CSCW 1998*, 257-264. *19% acceptance rate.*
95. Terveen, L.G and Hill, W.C. Finding and Visualizing Inter-site Clan Graphs, *Proceedings of CHI 1998*, 448-455. *23% acceptance rate.*
96. Terveen, L.G and Hill, W.C. Involving Users in Continuous Design of Web Content, in *Proceedings of DIS 1997*, ACM Press, 137-145. *41% acceptance rate*
97. Terveen, L.G., Hill, W.C., Amento, B., McDonald, D., and Creter, J. Building Task-Specific Interfaces to High Volume Conversational Data, *Proceedings of CHI 1997*, 226-233. *23% acceptance rate.*
98. Hill, W.C. and Terveen, L.G Using Frequency-of-Mention in Public Conversations for Social Filtering, *Proceedings of CSCW 1996*, 106-112.
99. Terveen, L.G. and Murray, L. Helping Users Program Their Personal Agents, *Proceedings of CHI 1996*, 355-361. *23% acceptance rate.*
100. Terveen, L.G. and Tuomenoksa, M.L. DynaDesigner: A Tool for Rapid Creation of Device-Independent, *Proceedings of INTERACT 1995*, 386-389.

101. Terveen, L.G. and Selfridge, P.G. Intelligent Assistance for Software Construction: A Case Study, *Proceedings of Knowledge-Based Software Engineering 1994*, 14-21.

102. Terveen, L.G., Selfridge, P.G., and Long, M.D. From 'Folklore' to 'Living Design Memory', in *Proceedings of INTERCHI 1993*, 15-22. *19% Acceptance Rate.*

103. Terveen, L.G. Interface Support for Data Archaeology, *ISMM International Conference on Information and Knowledge Management* (CIKM'93). *(acceptance rate for 1992 unknown; average is 24%*).

104. Brachman, R. J., Selfridge, P.G., Terveen, L.G., Altman, B., Borgida, A., Helper, F., Kirk, T., Lazar, A., McGuinness, D.L., and Resnick, L. A., Knowledge Representation Support for Data Archaeology, *ISMM International Conference on Information and Knowledge Management* (CIKM'92). *(acceptance rate for 1992 unknown; average is 24%).*

105. Selfridge, P.G., Terveen, L.G., and Long, M.D. Managing Design Knowledge to Provide Assistance to Large-Scale Software Development, *Proceedings of Knowledge-Based Software Engineering 1992*, 154-162.

106. Terveen, L.G. and Wroblewski, D.A. A Tool for Achieving Consensus in Knowledge Editing*, Proceedings of AAAI 1991*, 74-79*. 24% acceptance rate*.

107. Terveen, L.G., Wroblewski, D.A., and Tighe, S.N. Intelligent Assistance Through Collaborative Manipulation, *Proceedings of IJCAI 1991, 9-14*. (*acceptance rate for 1991 unknown; typically in low 20s*).

108. Terveen, L.G. and Wroblewski, D.A. A Collaborative Interface for Browsing and Editing Large Knowledge Bases, *Proceedings of AAAI 1990*, 491-496. *18% acceptance rate*.

Book Chapters

109. Terveen, L., Riedl, J., Konstan, J., and Lampe, C. (2014) "Study, Build, Repeat: Using Online Communities as a Research Platform", in *Human Computer Interaction Ways of Knowing*, edited by Judith S. Olson and Wendy Kellogg, New York: Springer.

110. Amento, B., Harrison, C., Nathan, M., and Terveen, L. (2009), Asynchronous Communication – Fostering Social Interaction with CollaboraTV, in Cesar, P., Geerts, D., and Chorianopoulos, K. (ed.), *Social Interactive Television: Immersive Shared Experiences and Perspectives* (2009), Information Science Reference.

111. Amento, B., Terveen, L.G and Hill, W. From PHOAKS to TopicShop: Experiments in Social Data Mining, in Lueg, C. and Fisher, D. (ed.), *From Usenet to CoWebs: Interacting with Social Information Spaces* (2002), Springer.

112. Whittaker, S., Terveen, L.G, Hill, W.C., and Cherny, L. The Dynamics of Mass Interaction, in Lueg, C. and Fisher, D. (ed.), *From Usenet to CoWebs: Interacting with Social Information Spaces* (2002), (this chapter is a reprint of Whittaker et al 1998), Springer.

113. Terveen, L.G and Hill, W. Beyond Recommender Systems: Helping People Help Each Other, in Carroll, J. (ed.), *HCI in the New Millennium* (2001), Addison Wesley.

114. Terveen, L.G. Computer-Mediated Collaboration, in *More than Screen Deep: Toward Every-Citizen Interfaces to the Nation's Information Infrastructure* (1997), National Academy Press.

Other Publications

115. Masli, M. and Terveen, L. Geographical Social Production: Lessons from Cyclopath, in CHI 2013 GeoHCI Workshop.

116. Masli, M., Bouman, L., Owen, A., and Terveen, L. Geowiki + route analysis = improved transportation planning, CSCW 2013 Interactive Poster.

117. Brady, S. S., Sieving, R. E., Terveen, L. G., Rosser, B. R. S., Kodet, A. J., & Rothberg, V. D. (2012, October). *TeensTalkHealth: An interactive website to promote healthy relationships and prevent STIs*. Paper presented at the annual meeting of the American Public Health Association, San Francisco, CA.

118. Society for Personality and Social Psychology, San Antonio, TX. January, 2011. "The role of community orientation in promoting online participation" (with J. S. Kim, P. C. Dwyer, J. Filson Moses, P. T. Fuglestad, C. A. Mannino, R Davies, & M. Snyder)

119. Society for Personality and Social Psychology, San Antonio, TX. January, 2011. "Applying a functional approach to participation in online groups" (with P. T. Fuglestad, P. C. Dwyer, J. Filson Moses, J. S. Kim, C. A. Mannino, R Davies, & M. Snyder)

120. Society for Personality and Social Psychology, San Antonio, TX. January, 2011. "Past volunteerism predicts amount of content contributed in an online community" (with P. C. Dwyer, J. Filson Moses, P. T. Fuglestad, J. S. Kim, C. A. Mannino, R Davies, & M. Snyder)

121. Society for Personality and Social Psychology, San Antonio, TX. January, 2011. "Social motives and personality as predictors of online participation" (with J. Filson Moses, P. C. Dwyer, P. T. Fuglestad, J. S. Kim, C. A. Mannino, R Davies, & M. Snyder)

122. Kapoor, N., Frankowski, D., Konstan, J., and Terveen, L. Lessons Learned in Implementing the CHIplace Online Community, in *Proceedings of Human-Computer Interaction International 2005*.

123. Zhou, C., Ludford, P., Frankowski, D., and Terveen, L. An Experiment in Exploring How People Describe Places, Short Paper in *Proceedings of Pervasive 2005*.

124. Zhou, C., Ludford, P., Frankowski, D., and Terveen, L. An Experiment in Discovering Personally Meaningful Places from Location Data, Short Paper in *Proceedings of CHI 2005*.

125. Kapoor, N., Konstan, J., and Terveen, L. How Peer Photos Influence Member Participation in Online Communities, Short Paper in *Proceedings of CHI 2005*.

126. Amento, B., Hill, W., and Terveen, L. The Sound of One Hand: A Wrist-mounted Bio-acoustic Fingertip Gesture Interface, Short Paper in *Proceedings of CHI 2002*, 724-725. *33% acceptance rate.*

127. Terveen, L., Hill, W., and Amento, B. Collaborative Filtering to Locate, Comprehend, and Organize Collections of Websites, in SIGART Bulletin, 9, 3&4 (1998), 10-17.

128. Terveen, L., Hill, W., Amento, B., McDonald, D., and Creter, J. 1997. PHOAKS: a system for sharing recommendations. *Commun. ACM* 40, 3 (Mar. 1997), 59-62.

129. Terveen, L.G., Stolze, M., and Hill, W. From 'Model World' to 'Magic World', in *SIGCHI Bulletin*, 27, 4 (1995), 31-34.

130. Terveen, L.G., Papavero, E., and Tuomenoksa, M. DynaDesigner: A Tool for Rapid Design and Deployment of Device-Independent Interactive Services, Refereed Formal Demonstration in *Adjunct Proceedings of CHI'95*, 29-30.

131. Terveen, L.G. Person-Computer Cooperation through Collaborative Manipulation. Ph.D. Thesis, University of Texas Department of Computer Sciences, 1991.

Invited Presentations

Carnegie Mellon University

Northwestern University

The University of California Irvine
Carleton College
Twin Cities MetroGIS Policy Board
Hennepin County Bicycle Advisory Committee
University of Minnesota Digital Humanities Collaborative
University of Minnesota Advanced Transportation Technologies Seminar Series
Georgia Tech
University of Minnesota Urban Ecosystems Symposium
University of Minnesota New Media
IBM T.J. Watson Research
University of Illinois
University of Maryland
University of Washington
Carnegie-Mellon University
Swedish Institute of Computer Science / Royal Institute of Technology
Uppsala University, Sweden
New Jersey Institute of Technology
Microsoft Research
Vassar University
Rensselaer Polytechnic Institute
University of Colorado
Virginia Tech
University of Nebraska

Conference Presentations

- ACM Conference on Computer-Supported Cooperative Work (CSCW), 2014.
- ACM Conference on Computer-Supported Cooperative Work (CSCW), 2012.
- ACM Conference on Computer-Supported Cooperative Work (CSCW), 2002.
- ACM Conference on Computer-Supported Cooperative Work (CSCW), 1998.
- ACM Conference on Human Factors in Computing Systems (CHI), 1998.
- ACM Conference on Design of Interactive Systems, 1997.
- ACM Conference on Human Factors in Computing Systems (CHI), 1997.
- ACM Conference on Human Factors in Computing Systems (CHI), 1996.
- IFIP TC13 International Conference on Human-Computer Interaction (INTERACT), 1995.
- IEEE Conference on Knowledge-Based Software Engineering, 1994.
- ACM Conference on Human Factors in Computing Systems (CHI), 1993.

- Conference on Information and Knowledge Management, 1993.
- Conference on Information and Knowledge Management, 1992.
- National Conference on Artificial Intelligence (AAAI), 1991.
- International Joint Conference on Artificial Intelligence (IJCAI), 1991.
- National Conference on Artificial Intelligence (AAAI), 1990.

Workshop Presentations

- CHI 2013 GeoHCI Workshop
    - "Geographical Social Production: Lessons from Cyclopath"
- CSCW 2002 Workshop on "The Role of Place in Shaping Virtual Communities":
    - "Place-Based Community Information Systems".
- CHI 99 Workshop on "Interacting with Recommender Systems":
    - "Visualization Interfaces for Recommender Systems".
- 1999 Human-Computer Interaction Consortium Workshop:
    - "A Reference Task Agenda for Human-Computer Interaction".
- 1998 AAI Workshop on "Recommender Systems":
    - "The PHOAKS Recommender System".
- 1997 Human-Computer Interaction Consortium Workshop:
    - "The PHOAKS Recommender System".
- 1995 Lifelike Computer Characters Workshop:
    - "Hidden Hands, not Talking Heads: The Magic World Interaction Paradigm".
- 1995 Lifelike Computer Characters Workshop:
    - "Moving Agent-User Voice Dialogue towards Natural Conversation".
- CHI 95 Workshop on "'Model World' to 'Magic World': Making Visual Objects the Medium for Intelligent Design Assistance":
    - "The 'Magic World' Approach to Human-Computer Collaboration".
- 1993 AAAI Fall Symposium: "Human-Computer Collaboration: Reconciling Theory, Synthesizing Practice":
    - "A Framework for Human-Computer Collaboration"
- 1993 AI-ED Workshop on "Collaborative Problem Solving: Theoretical Frameworks and Innovative Systems":
    - "Collaborative Problem Solving in Interactive Systems".
- 1992 CAIA Workshop on "Applying AI To Software Problems: Assessing Promises and Pitfalls":
    - "Representing and Disseminating Software Design Knowledge".
- 1992 AAAI Spring Symposium on "Cognitive Aspects of Knowledge Acquisition":
    - "In The Footprints of The Masters: Embedding Knowledge Acquisition in

Organizational Activity".

- AAAI 90 Workshop on "Complex Systems, Ethnomethodology, and Interaction Analysis":
  - "Resources for Person-Computer Collaboration".
- 1990 AAAI Spring Symposium on "Knowledge-Based Human-Computer Communication":
  - "Tools for Human-Computer Collaboration".

Patents

U.S. Patent # 5,388,188. Apparatus and methods for providing design advice. (with P. Selfridge). Issued February 7, 1995.

U.S. Patent #5,659,724. Interactive data analysis apparatus employing a knowledge base. (with A. Borgida, R.J. Brachman, T. Kirk, and P. Selfridge). Issued August 19, 1997.

U.S. Patent #5,680,530. Graphical environment for interactively specifying a target system (with P. Selfridge). Issued October 21, 1997.

U.S. Patent #5,806,060. Interactive data analysis employing a knowledge base. (with A. Borgida, R.J. Brachman, T. Kirk, and P. Selfridge). Issued September 8, 1998.

U.S. Patent # 5,809,492. Apparatus and method for defining rules for personal agents. (with L. Murray). Issued September 15, 1998.

U.S. Patent # 5,953,393. Personal Telephone Agent. (With P. Culbreth, P. Danielsen, R.J. Hall, E. Papavero, and M. Tuomenoksa). Issued September 14, 1999.

U.S. Patent #6,029,192. System and method for locating resources on a network using resource evaluations derived from electronic messages. (with W.C. Hill). Issued February 22, 2000.

U.S. Patent #6,244,873. Wireless myoelectric control apparatus and methods. (with W.C. Hill, F.C. Pereira, and Y. Singer). Issued June 12, 2001.

U.S. Patent #6,256,648. System and method for selecting and displaying hyperlinked information resources. (with W.C. Hill). Issued July 3, 2001.

CONSULTING

- Sound View Innovations, LLC v. LinkedIn Corp.
  - Retained as expert on behalf of LinkedIn by Klarquist Sparkman (2016)
- TiVo Inc. v. Samsung Electronics
  - Retained as expert on behalf of TiVo by Irell & Manella (2016)
  - Case No. 2:15-cv-1503 (E.D. Tex.)
- Netflix v. Rovi 4:11-cv-06591
  - Retained as expert on behalf of Nextflix by Keker Van Nest (2014-2015)
  - Presented technology tutorial to Judge, March 2015
- Clear with Computers vs. Vermeer

  Clear with Computers, LLC vs. Vermeer Corporation
  Civil Action No. 13-cv-00167 (E.D. Texas)
  - Retained by Faegre, Baker, and Daniels
- Clear with Computer vs. Manitowoc Cranes, LLC.

  Clear with Computers, LLC vs. Vermeer Corporation
  Civil Action No. 13-cv-00167 and related cases (E.D. Texas)

    - Retained by Baker Botts
- Clear with Computers vs. Terex Corporation
  Clear with Computers, LLC v. Terex Corporation
  Civil Action No. 6:12-cv-00634 (E.D. Texas)
    - Retained by Fish & Richardson
- Clear with Computers vs. Valmont Industries, Inc.
  Clear with Computers, LLC v. Valmont Industries, Inc.
  Civil Action No. 6:13-cv-00166 (E.D. Texas)
    - Retained by Fish & Richardson
    - During the first half of 2014 I was retained by three different firms on behalf of four different clients to work on invalidity. I also worked on non-infringement on behalf of Vermeer.

- Intellectual Ventures vs. U.S. Bancorp and U.S. Bank
  Intellectual Ventures II LLC v. U.S. Bancorp and U.S. Bank
  Civil Action No. 13-2071 (U.S. District of Minnesota, Fourth Division)
    - Retained by Winthrop & Weinstine on behalf of U.S. Bancorp and U.S. Bank
    - Consulted on non-infringement and invalidity from late 2013 through mid 2014

- "Products Containing Interactive Program Guide and Parental Control Technology": Rovi vs. LG Electronics, Mitsubishi, Netflix, Roku, and Vizio.
    - Case No. ITC 337-TA-845
    - International Trade Commission
    - Retained as expert on behalf of respondents Nextflix and Roku by Keker Van Nest (2012-2013)
    - Deposed: 2013
    - Testified in court: 2013

- Motorola Mobility vs. Microsoft and Microsoft vs. Motorola Mobility (Counterclaim)
    - Case No. 1:10-CV-24063-MORENO
    - United States District Court Southern District of Florida
    - Retrained as expert on behalf of Microsoft (2011-2012) by Sidley Austin
    - Deposed: 2011

- Interval Licensing LLC vs. Aol, Inc.; Apple, Inc., eBAY, Inc.; Facebook, Inc.; Google, Inc.; Netflix, Inc.; Office Depot Inc.; OfficeMax Inc.; Staples, Inc.; Yahoo, Inc.; and YouTube, LLC.
    - Case No. 2:10-cv-01385-MJP
    - United States District Court Western District of Washington at Seattle
    - Retained as expert of behalf of four of the respondents – eBay Inc.; Netflix, Inc.; Office Depot, Inc.; and Staples, Inc. – by Klarquist Sparkman (2011)

- Microsoft vs. Tivo
    - Case No. 5:10-cv-00240-RS
    - United States District Court Northern District of California
    - Retained as expert on behalf of plaintiff by Perkins Coie (2010-2011)

- Elsevier B.V., Elsevier Inc., and Mosby, Inc. vs. UnitedHealth Group, Inc. etc.
    - 09 Civ. 2124
    - United States District Court Southern District of New York
    - Retained as expert on behalf of respondent by Dorsey & Whitney (2010)

- NORTHBROOK DIGITAL LLC vs. Vendio Services, Inc.
    - Civil File No. 07-CV-2250
    - United States District Court District of Minnesota
    - Retained as expert on behalf of plaintiff by Dorsey & Whitney (2009)

SERVICE

External Professional Activities

ACM Special Interest Group on Computer Human Interaction

- President: 2015-2018
- Executive Committee: Adjunct Chair for Awards, 2012 – 2015.
- Executive Committee: Vice President for Membership and Communication, 2009 – 2012.

Conference Chair

- CHI 2002: ACM Conference on Human Factors in Computing Systems
- IUI 1998: ACM Conference on Intelligent User Interfaces.

Program Committee Chair

- CSCW 2004: ACM Conference on Computer-Supported Cooperative Work.
- CSCW 2013: ACM Conference on Computer-Supported Cooperative Work.

Computer Supported Cooperative Work (CSCW) Steering Committee

- Chair, 2012-2014 (First elected Chair of the committee that oversees the CSCW Conference and the general CSCW and Social Computing research community.)

Awards Committee Chair

- CSCW 2008: ACM Conference on Computer-Supported Cooperative Work.

Journals Edited

- Special Issue of *Knowledge-Based Systems* on Human-Computer Collaboration, Vol. 8, No. 2-3, 1995.

Proceedings Edited

- Proceedings of the 2002 ACM Conference on Human Factors in Computing Systems (CHI

2002).

- Proceedings of the 1998 ACM Conference on Intelligent User Interfaces (IUI 1998).

Editorial Boards

- Communications of the ACM, 2009-present.
- ACM Transactions on CHI, 2000-2006.
- Knowledge-Based Systems, 1993-present.
- ACM *intelligence*, 1998-2001.

Program Committees

- ACM Conference on Human Factors in Computing Systems (CHI): 1999-2004, 2006, 2016.
- ACM Conference on Computer-Supported Cooperative Work (CSCW): 2000, 2006, 2008, 2010, 2011, 2012, 2015.
- ACM Conference on Intelligent User Interfaces (IUI): 1997-2000, 2004.
- ACM Recommender Systems Conference: 2007, 2008, 2009, 2010, 2013.
- ACM GROUP Conference: 2007, 2009, 2010.
- ACM SIGIR Conference: 2008.
- AAAI Conference on the Web and Social Media: 2011, 2012, 2016.
- User Modeling, Adaptation, and Personalization: 2011, 2016.
- NordiCHI 2008, 2010.
- Communities and Technology: 2009.
- Computer-Supported Cooperative Learning 2002.
- User Modeling: 2001.
- International Conference on Knowledge Capture 2001.
- Knowledge-Based Software Engineering (KBSE): 1994-1999.
- Intelligent Data Analysis: 1997.
- National Conference on Artificial Intelligence (AAAI): 1996-1997.
- IEEE Conference on Artificial Intelligence for Applications (CAIA): 1994.
- International Workshop on Privacy-Aware Location-based Mobile Services: 2007.
- SIGIR Workshop on Future Challenges in Expertise Retrieval: 2008.

Other conference leadership positions

- ACM Conference on Computer-Supported Cooperative Work Doctoral Consortium Co-Chair, 2016 and 2011.
- The 2nd AAAI Conference on Human Computation and Crowdsourcing (HCOMP 2014) Doctoral Consortium Co-Chair.
- International Symposium on Wikis and Open Collaboration Doctoral Symposium Chair,

2011.

- ACM Conference on Recommender Systems Doctoral Consortium, Co-Chair, 2007.
- International Joint Conference for Artificial Intelligence Doctoral Consortium, Co-Chair, 1997.
- American Association for Artificial Intelligence Doctoral Consortium, Co-Chair, 1997.
- American Association for Artificial Intelligence Doctoral Consortium, Co-Chair, 1996.

Reviewer

Journals

- ACM Computing Surveys.
- IEEE Transactions on Data and Knowledge Engineering.
- IEEE Expert.
- Information Systems.
- International Journal of Human-Computer Studies.
- Journal of Computer-Supported Cooperative Work.

Conferences

- Ninth IFIP International Conference on Human-Computer Interaction (INTERACT): 2003.
- ACM Conference on Human Factors in Computing Systems (CHI): 1995-1998.
- ACM Conference on Computer-Supported Cooperative Work (CSCW):1998.
- ACM Symposium on User Interface Software and Technology (UIST): 1996.
- ISSM Conference on Information and Knowledge Management: 1993.

National Science Foundations Panels

Served on panels in 1996, 1998, 2001, 2003, 2005, 2010, 2012, 2013.

Workshops Organized

- ACM Wikis and Open Collaboration Doctoral Symposium, 2011.
- Social Computational Systems Community Workshop, 2011.
- ACM Computer Supported Cooperative Work Doctoral Consortium, 2011.
- International Conference on Ubiquitous Computing (UbiComp): “Multi-device Interfaces for Ubiquitous Peripheral Interaction”, 2003.
- ACM Conference on Human Factors in Computer Systems (CHI): “Interacting with Recommender Systems”, 1999.
- International Joint Conference for Artificial Intelligence Doctoral Consortium, 1997.
- American Association for Artificial Intelligence Doctoral Consortium, 1997.
- American Association for Artificial Intelligence Doctoral Consortium, 1996.

- ACM Conference on Human Factors in Computer Systems (CHI): "'Model World' to 'Magic World': Making Visual Objects the Medium for Intelligent Design Assistance", 1995.
- ACM Conference on Human Factors in Computer Systems (CHI): "New Uses and Abuses of Interaction History", 1994.
- AAAI Fall Symposium: "Human-Computer Collaboration: Reconciling Theory, Synthesizing Practice", 1993.
- World Conference on Artificial Intelligence and Education: "Collaborative Problem Solving: Theoretical Frameworks and Innovative Systems", 1993.
- Conference on AI for Applications: "Applying AI To Software Problems: Assessing Promises and Pitfalls", 1992.

Other Professional Service

- Member of SIGCHI Publications Board and Conference Management Committee, 2002-2004.
- ACM Special Interest Group on Artificial Intelligence Conference Chair (1995-1999); originated and co-organized Doctoral Consortia held in conjunction with AAAI and IJCAI.

Internal Service

- Chair, Department Head Search Committee, 2015.
- Chair, Strategic Planning Committee, 2013-2014.
- Chair, Social Computing Faculty Search Committee, 2013-2014.
- Curriculum Committee, 2012-2013.
- Chair, Strategic Planning Committee / Faculty Recruiting Committee, 2011-2012.
- Director of Graduate Studies: 2007-2010.
- Communications Committee: 2007
- Newsletter/Brochure Committee (chair): 2006-2007
- Research Opportunities Committee: 2005-2007
- Curriculum Committee: 2003-2005.
- Hosted Robert Kraut, Cray Colloquium speaker: October 2003.
- Participated in recruiting and admission activities.
- Information, Technology, and Everyday Life Initiative: Committee Member.

TEACHING AND ADVISING

Ph.D. Students Advised

- Dan Cosley (co-advised with John Riedl): completed PhD July 2006, currently an Associate Professor at Cornell University; on leave as Program Director at NSF.
- Pamela Ludford: completed PhD September 2007, currently an independent consultant.

- Reid Priedhorksy: completed PhD August 2010, currently a Postdoctoral Research Associate at Los Alamos National Laboratories.
- Mikhil Masli: completed PhD July 2013, currently employed at IBM.
- Aaron Halfaker (co-advised with John Riedl): completed PhD September 2013, currently employed at the Wikimedia Foundation.
- Katie Panciera: completed PhD August 2014; currently employed at Google.
- Fernando Torre: completed PhD September 2014; founder of a startup.
- Shuo (Steven) Chang: completed PhD August 2016; currently employed at Quora.
- Tien Nguyen (co-advised with Joe Konstan): completed PhD August 2016; currently employed at Pinterest.
- Morten Warncke-Wang (co-advised with Brent Hecht): completed PhD December 2016.
- Jacob Thebault-Spieker (entered program Fall 2011).
- Vikas Kumar (entered program Fall 2011)
- Hannah Miller (entered program Fall 2013)
- Andrew Hall (entered program Fall 2014)
- Bowen Yu (entered program Fall 2014; co-advised with Haiyi Zhu).

M.S. Students Advised

- Tyler Danielsen – received degree in 2016
- Jie Kang – received degree in 2016
- Zahra Eslami – received degree in 2015
- Yanjie Liu – received degree in 2013
- Renji Yu – received degree in 2012
- Carol Drysdale – received degree in 2011
- Jingwen Zhang – received degree in 2011
- Jisu Oh – received degree in 2010
- Sara Drenner – received degree in 2008
- Anna Rouben – received degree in 2006
- Arjun Sundararajan – received degree in 2006
- Pamela Ludford – received degree in 2005
- Rahul Akolkar – graduated May 2004

Undergraduate Honors / Senior Thesis Students Advised

- Harmanprett Kaur, 2016
- Arlo Siemsen, 2014
- Johnathan Frenz, 2013

- David Pitchford - 2012
- Michael Ludwig - 2010
- Jordan Focht – 2010
- Kurt Wilms - 2005
- John Murphy – 2004

Other Committees

- Catherine Grevet, PhD.: member of preliminary and final examination committees (Gerogia Tech).
- Loxley Wang, PhD: member of preliminary and final examination committees.
- Abigail Bakke, PhD: member of preliminary and final examination committees
- Tahir Sousa, MS: member of final examination committee
- Tony Lam, PhD: member of preliminary examination committee.
- Michael Janseen, PhD: member of preliminary examination committee.
- Haleh Hagh Shenas, PhD: member of preliminary & final examination committees.
- Julie Beilfuss, MS: member of final examination committee.
- Liv Knatterud, MS: member of final examination committee.
- Sean McNee, MS: member of final examination committee.
- Shankar Subrahmanian, MS: member of final examination committee.
- Vamsee Venuturumilli, MS: member of final examination committee.
- Eric Gilbert, Ph.D. Preliminary and Final Examination Committees (UIUC, 2009/2010): member of thesis committee.
- Yi (Jenny) Zhang, Ph.D. (New Jersey Institute of Technology, 2004): member of thesis committee.
- Brian Amento, Ph.D. (Virginia Tech, 2001): member of thesis committee.
- David McDonald, Ph.D. (UC Irvine, 2000): member of thesis committee.

Courses taught

Semester	Course
Fall 2016	CSCI 5115: User Interface Design, Implementation, and Evaluation
Spring 2016	CSCI 5125: Collaborative and Social Computing
Fall 2015	CSCI 1133H: Introduction to Computer Science (Honors)
	CSCI 8115: Human-Computer Interaction and UI Technology
Spring 2015	CSCI 5125: Collaborative and Social Computing
Fall 2014	CSCI 5115: User Interface Design, Implementation, and Evaluation
	HSEM 2519H: Honors Seminar on Crowdsourcing
Spring 2014	CSCI 8115: Human-Computer Interaction and UI Technology
Fall 2013	CSCI 1901H: Introduction to Computer Science (Honors)
	CSCI 5115: User Interface Design, Implementation, and Evaluation
Spring 2013	CSCI 5125: Collaborative and Social Computing
	SEng 5115: User Interface Design and Evaluation
Fall 2012	CSCI 5115: User Interface Design, Implementation, and Evaluation
Spring 2012	CSCI 8115: Human-Computer Interaction and UI Technology
	SEng 5115: User Interface Design and Evaluation
Fall 2011	CSCI 1902: Structure of Computer Programming II
	CSCI 5115: User Interface Design, Implementation, and Evaluation
Spring 2011	CSCI 5125: Collaborative and Social Computing
	SEng 5115: User Interface Design and Evaluation
Fall 2010	CSCI 5115: User Interface Design, Implementation, and Evaluation
Spring 2010	CSCI 8115: Human-Computer Interaction and UI Technology
Fall 2009	CSCI 5115: User Interface Design, Implementation, and Evaluation
Spring 2009	CSCI 5125: Collaborative and Social Computing
	SEng 5115: User Interface Design and Evaluation
Fall 2008	CSCI 5115: User Interface Design, Implementation, and Evaluation
Spring 2008	CSCI 1902: Structure of Computer Programming II
	SEng 5115: User Interface Design and Evaluation
Fall 2007	CSCI 5115: User Interface Design, Implementation, and Evaluation
Spring 2007	SEng 5115: User Interface Design and Evaluation
Spring 2007	CSCI 5125: Collaborative and Social Computing
Fall 2006	CSCI 1902: Structure of Computer Programming II
	CSCI 5115: User Interface Design, Implementation, and Evaluation
Spring 2006	CSCI 8115: Human-Computer Interaction and UI Technology
Fall 2005	CSCI 1902: Structure of Computer Programming II
Spring 20005	CSCI 5116: GUI Toolkits and Their Implementation

Fall 2004	CSCI 5115: User Interface Design, Implementation, and Evaluation
Spring 2004	CSCI 5980: Collaborative Computing
Fall 2003	CSCI 5115: User Interface Design, Implementation, and Evaluation
Spring 2003	CSCI 5116: GUI Toolkits and Their Implementation
Fall 2002	CS 8115: Human-Computer Interaction and UI Technology

Tutorials

"Intelligent User Interfaces: Issues, Approaches, Evaluation", offered at 1993 Conference on Artificial Intelligence for Applications.

APPENDIX D

Direct manipulation systems offer the satisfying experience of operating on visible objects. The computer becomes transparent, and users can concentrate on their tasks.

Direct Manipulation: A Step Beyond Programming Languages

Ben Shneiderman, University of Maryland

> Leibniz sought to make the form of a symbol reflect its content. "In signs," he wrote, "one sees an advantage for discovery that is greatest when they express the exact nature of a thing briefly and, as it were, picture it; then, indeed, the labor of thought is wonderfully diminished."
>
> Frederick Kreiling, "Leibniz," *Scientific American*, May 1968

Certain interactive systems generate glowing enthusiasm among users—in marked contrast with the more common reaction of grudging acceptance or outright hostility. The enthusiastic users' reports are filled with positive feelings regarding

- mastery of the system,
- competence in the performance of their task,
- ease in learning the system originally and in assimilating advanced features,
- confidence in their capacity to retain mastery over time,
- enjoyment in using the system,
- eagerness to show it off to novices, and
- desire to explore more powerful aspects of the system.

These feelings are not, of course, universal, but the amalgam does convey an image of the truly pleased user. As I talked with these enthusiasts and examined the systems they used, I began to develop a model of the features that produced such delight. The central ideas seemed to be visibility of the object of interest; rapid, reversible, incremental actions; and replacement of complex command language syntax by direct manipulation of the object of interest—hence the term "direct manipulation."

Examples of direct manipulation systems

No single system has all the attributes or design features that I admire—that may be impossible—but those described below have enough to win the enthusiastic support of many users.

Display editors. "Once you've used a display editor, you'll never want to go back to a line editor. You'll be spoiled." This reaction is typical of those who use full-page display editors, who are great advocates of their systems over line-oriented text editors. I heard similar comments from users of stand-alone word processors such as the Wang system and from users of display editors such as EMACS on the MIT/Honeywell Multics system or "vi" (for visual editor) on the Unix system. A beaming advocate called EMACS "the one true editor."

Roberts[1] found that the overall performance time of display editors is only half that of line-oriented editors, and since display editors also reduce training time, the evidence supports the enthusiasm of display editor devotees. Furthermore, office automation evaluations consistently favor full-page display editors for secretarial and executive use.

The advantages of display editors include

Display of a full 24 to 66 lines of text. This full display enables viewing each sentence in context and simplifies reading and scanning the document. By contrast, the

A portion of this article was derived from the author's keynote address at the NYU Symposium on User Interfaces, "The Future of Interactive Systems and the Emergence of Direct Manipulation," published in *Human Factors in Interactive Computer Systems*, Y. Vassiliou, ed., Ablex Publishing Co., Norwood, N.J., 1983.

0018-9162/83/0800-0057$01.00 © 1983 IEEE

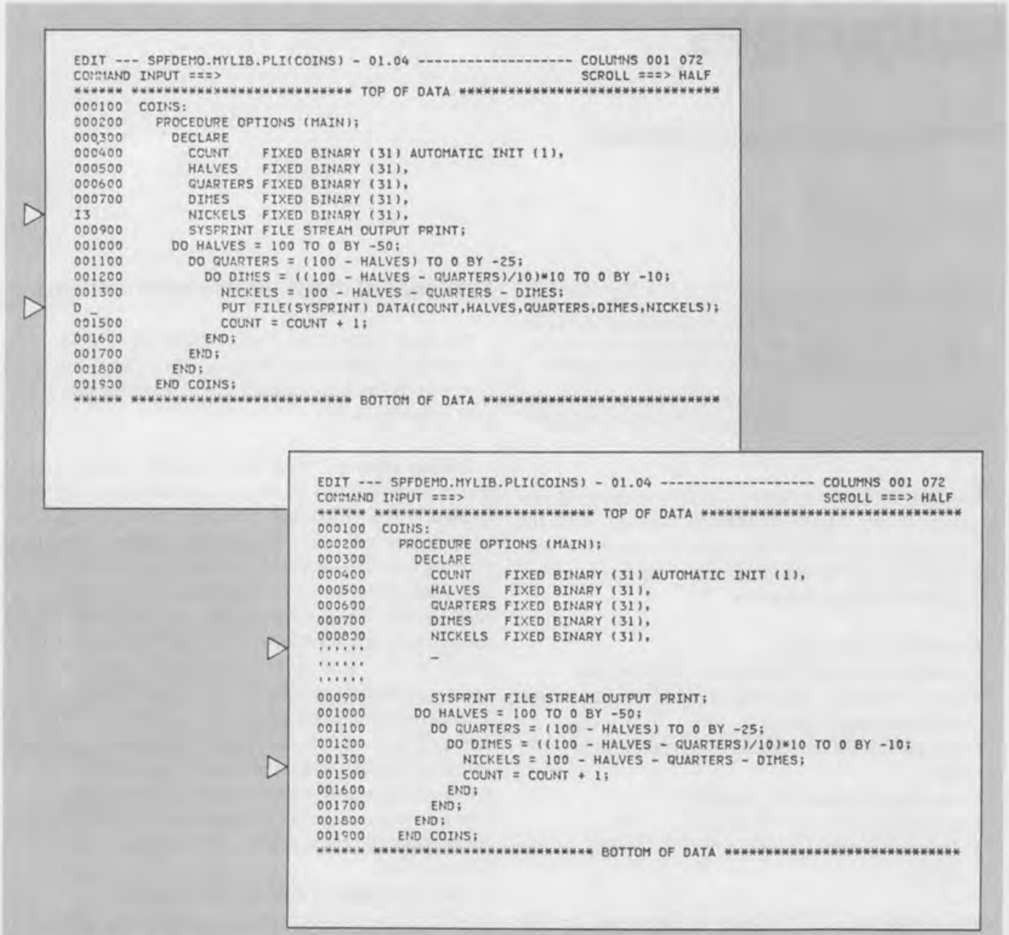
one-line-at-a-time view offered by line editors is like seeing the world through a narrow cardboard tube.

Display of the document in its final form. Eliminating the clutter of formatting commands also simplifies reading and scanning the document. Tables, lists, page breaks, skipped lines, section headings, centered text, and figures can be viewed in the form that will be printed. The annoyance and delay of debugging the format commands is eliminated because the errors are immediately apparent.

Cursor action that is visible to the user. Seeing an arrow, underscore, or blinking box on the screen gives the operator a clear sense of where to focus attention and apply action.

Cursor motion through physically obvious and intuitively natural means. Arrow keys or devices such as a mouse, joystick, or graphics tablet provide natural physical mechanisms for moving the cursor. This is in marked contrast with commands such as UP 6, which require an operator to convert the physical action into correct syntactic form and which may be difficult to learn, hard to recall, and a source of frustrating errors.

Labeled buttons for action. Many display editors have buttons etched with commands such as INSERT, DELETE, CENTER, UNDERLINE, SUPERSCRIPT, BOLD, or LOCATE. They act as a permanent menu selection display, reminding the operator of the features and obviating memorization of a complex command-lan-



```
EDIT --- SPFDEMO.MYLIB.PLI(COINS) - 01.04 ------------------- COLUMNS 001 072
COMMAND INPUT ===>                                              SCROLL ===> HALF
****** ************************** TOP OF DATA *******************************
000100  COINS:
000200    PROCEDURE OPTIONS (MAIN);
000300      DECLARE
000400        COUNT    FIXED BINARY (31) AUTOMATIC INIT (1),
000500        HALVES   FIXED BINARY (31),
000600        QUARTERS FIXED BINARY (31),
000700        DIMES    FIXED BINARY (31),
I3            NICKELS  FIXED BINARY (31),
000900        SYSPRINT FILE STREAM OUTPUT PRINT;
001000      DO HALVES = 100 TO 0 BY -50;
001100        DO QUARTERS = (100 - HALVES) TO 0 BY -25;
001200          DO DIMES = ((100 - HALVES - QUARTERS)/10)*10 TO 0 BY -10;
001300            NICKELS = 100 - HALVES - QUARTERS - DIMES;
D _               PUT FILE(SYSPRINT) DATA(COUNT,HALVES,QUARTERS,DIMES,NICKELS);
001500            COUNT = COUNT + 1;
001600          END;
001700        END;
001800      END;
001900    END COINS;
****** ************************** BOTTOM OF DATA ****************************
```

```
EDIT --- SPFDEMO.MYLIB.PLI(COINS) - 01.04 ------------------- COLUMNS 001 072
COMMAND INPUT ===>                                              SCROLL ===> HALF
****** ************************** TOP OF DATA *******************************
000100  COINS:
000200    PROCEDURE OPTIONS (MAIN);
000300      DECLARE
000400        COUNT    FIXED BINARY (31) AUTOMATIC INIT (1),
000500        HALVES   FIXED BINARY (31),
000600        QUARTERS FIXED BINARY (31),
000700        DIMES    FIXED BINARY (31),
000800        NICKELS  FIXED BINARY (31),
''''''        _
''''''
''''''
000900        SYSPRINT FILE STREAM OUTPUT PRINT;
001000      DO HALVES = 100 TO 0 BY -50;
001100        DO QUARTERS = (100 - HALVES) TO 0 BY -25;
001200          DO DIMES = ((100 - HALVES - QUARTERS)/10)*10 TO 0 BY -10;
001300            NICKELS = 100 - HALVES - QUARTERS - DIMES;
001500            COUNT = COUNT + 1;
001600          END;
001700        END;
001800      END;
001900    END COINS;
****** ************************** BOTTOM OF DATA ****************************
```

Figure 1. This example from the IBM SPF display editor shows 19 lines of a PL/I program. The commands to insert three lines (I3) and to delete one line (D or D1) are typed on the appropriate lines in the first screen display. Pressing ENTER causes commands to be executed and the cursor to be placed at the beginning of the inserted line. New program statements can be typed directly in their required positions. Control keys move the cursor around the text to positions where changes are made by overstriking. A delete key causes the character under the cursor to be deleted and the text to the left to be shifted over. After pressing an insert key, the user can type text in place. Programmed function keys allow movement of the window forwards, backwards, left, and right over the text. (Examples courtesy of IBM.)

guage syntax. Some editors provide basic functionality with only 10 or 15 labeled buttons, and a specially marked button may be the gateway to advanced or infrequently used features offered on the screen in menu form.

Immediate display of the results of an action. When a button is pressed to move the cursor or center the text, the results appear on the screen immediately. Deletions are apparent at once, since the character, word, or line is erased and the remaining text rearranged. Similarly, insertions or text movements are shown after each keystroke or function button press. Line editors, on the other hand, require a print or display command before the results of a change can be seen.

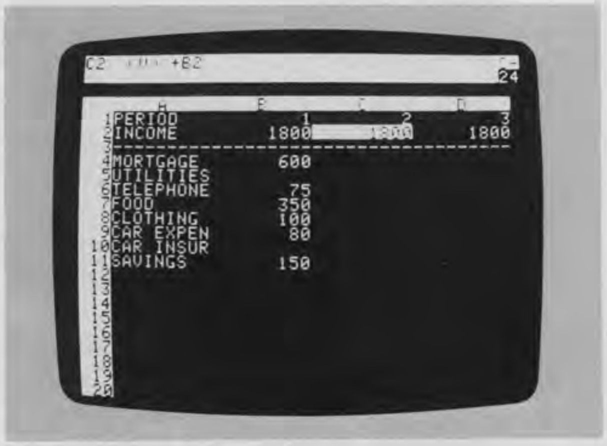
Rapid action and display. Most display editors are designed to operate at high speeds: 120 characters per second (1200 baud), a full page in a second (9600 baud), or even faster. This high display rate coupled with short response time produces a thrilling sense of power and speed. Cursors can be moved quickly, large amounts of text can be scanned rapidly, and the results of commands can be shown almost instantaneously. Rapid action also reduces the need for additional commands, thereby simplifying product design and decreasing learning time. Line editors operating at 30 characters per second with three- to eight-second response times seem sluggish in comparison. Speeding up line editors adds to their attractiveness, but they still lack features such as direct overtyping, deletion, and insertion.

Easily reversible commands. Mistakes in entering text can be easily corrected by backspacing and overstriking. Simple changes can be made by moving the cursor to the problem area and overstriking, inserting, or deleting characters, words, or lines. A useful design strategy is to include natural inverse operations for each operation. Carroll[2] has shown that congruent pairs of operations are easy to learn. As an alternative, many display editors offer a simple UNDO command that cancels the previous command or command sequence and returns the text to its previous state. This easy reversibility reduces user anxiety about making mistakes or destroying a file.

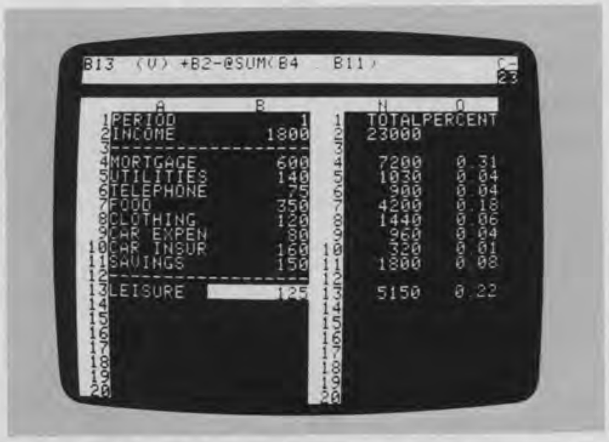
The large market for display editors generates active competition, which accelerates evolutionary design refinements. Figure 1 illustrates the current capabilities of an IBM display editor.

Visicalc. Visicorp's innovative financial forecasting program, called Visicalc, was the product of a Harvard MBA student, who was frustrated by the time needed to carry out multiple calculations in a graduate business course. Described as an "instantly calculating electronic worksheet" in the user's manual, it permits computation and display of results across 254 rows and 63 columns and is programmed without a traditional procedural control structure. For example, positional declarations can prescribe that column 4 displays the sum of columns 1 through 3; then every time a value in the first three columns changes, the fourth column changes as well. Complex dependencies among manufacturing costs, distribution costs, sales revenue, commissions, and profits can be stored for several sales districts and months so that the impact of changes on profits is immediately apparent.

Since Visicalc simulates an accountant's worksheet, it is easy for novices to comprehend. The display of 20 rows and up to nine columns, with the provision for multiple windows, gives the user sufficient visibility to easily scan information and explore relationships among entries (see Figure 2). The command language for setting up the worksheet can be tricky for novices to learn and for infrequent users to remember, but most users need learn only the basic commands. According to Visicalc's distributor, "It jumps," and the user's delight in watching this propagation of changes cross the screen helps explain its appeal.



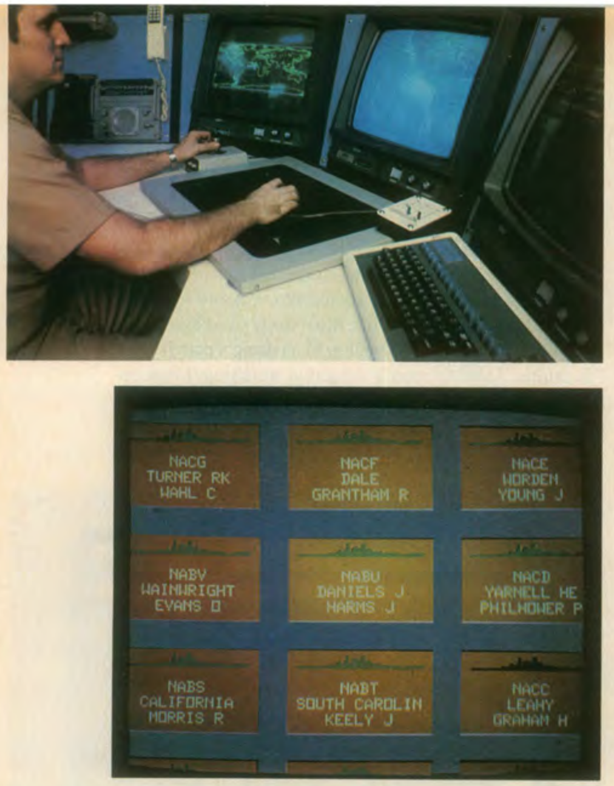
```
C2  (V) +B2                                        C-
                                                   24
        A            B          C          D
 1PERIOD             1          2          3
 2INCOME          1800       1800       1800
 3---------------------------------------------
 4MORTGAGE         600
 5UTILITIES
 6TELEPHONE         75
 7FOOD             350
 8CLOTHING         100
 9CAR EXPEN         80
10CAR INSUR
11SAVINGS          150
12
13
14
15
16
17
18
19
20
```



```
B13  (V) +B2-@SUM(B4 . B11)                        C-
                                                   23
        A            B              N       O
 1PERIOD             1     1   TOTALPERCENT
 2INCOME          1800     2   23000
 3-----------------------  3
 4MORTGAGE         600     4    7200    0.31
 5UTILITIES        140     5    1030    0.04
 6TELEPHONE         75     6     900    0.04
 7FOOD             350     7    4200    0.18
 8CLOTHING         120     8    1440    0.06
 9CAR EXPEN         80     9     960    0.04
10CAR INSUR        160    10     320    0.01
11SAVINGS          150    11    1800    0.08
12-----------------------  12
13LEISURE          125    13    5150    0.22
14                        14
15                        15
16                        16
17                        17
18                        18
19                        19
20                        20
```

Figure 2. This simple Visicalc program display (top) shows four columns and 20 rows of home budget information. The cursor, an inverse video light bar controlled by key presses, is in position C2. The top command line shows that C2 is a value (as opposed to a text string) that has been set up to have the same value as position B2.

The second display (above) shows two windows over the home budget data with row sums to the right. The last row shows leisure dollar amounts, which are established by the top command line formula as the income minus the sum of expenses. A change to the income or expense values would immediately propagate to all affected values. (Displays reproduced by permission of Visicorp.)

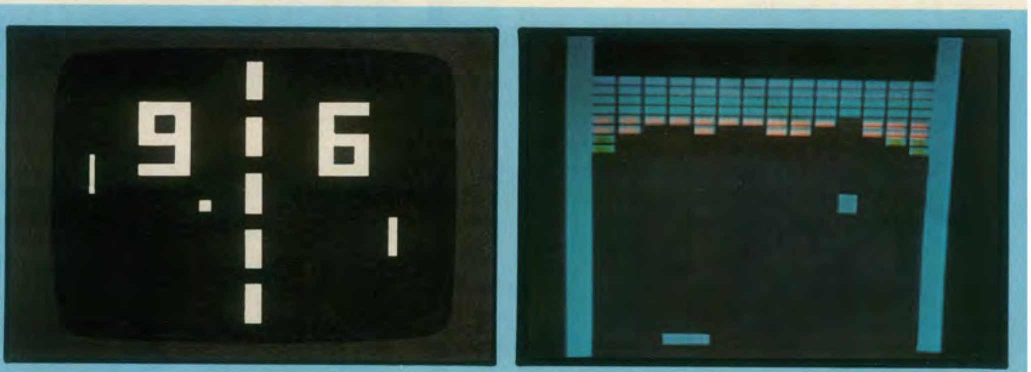


Spatial data management. The developers of the prototype spatial data management system[3] attribute the basic idea to Nicholas Negroponte of MIT.

In one scenario, a user seated before a color graphics display of the world zooms in on the Pacific to see markers for military ship convoys. Moving a joystick fills the screen with silhouettes of individual ships, which can be zoomed in on to display structural details or, ultimately, a full-color picture of the captain. (See Figure 3.)

In another scenario, icons representing different aspects of a corporation, such as personnel, organization, travel, production, or schedules, are shown on a screen. Moving the joystick and zooming in on objects takes users through complex "information spaces" or "I-spaces" to locate the item of interest. For example, when they select a department from a building floor

Figure 3. A spatial data management system has been installed on the aircraft carrier USS *Carl Vinson*. In the photo at top left, the operator has a world map on the left screen and a videodisc map of selected areas on the center screen. After some command selections with the data tablet and puck, the operator can zoom in on specific data such as the set of ships shown in the second photo. With further selections the operator can get detailed information about each ship, such as the length, speed, and fuel. (Photos courtesy of Computer Corporation of America.)



In 1971, about the only people playing video games were students in computer science laboratories. By 1973, however, millions of people were familiar with at least one video game—Pong (above left). A few years later came Breakout (above right), which, according to many designers, was the first true video game and the best one ever invented. Pong and other early games imitated real life, but Breakout could not have existed in any medium other than video. In the game, a single paddle directed a ball toward a wall of color bricks; contact made a brick vanish and changed the ball's speed.

When the first arcade video game, Computer Space, went on location in a Sears store, its joystick was torn off before the end of the first day. As a result, game designers have sought controls that were both easy to use and hard to destroy. Centipede (above left) uses simple controls—a trackball and one button. On the other hand, Defender (above right) has five buttons and a joystick; novice players are confused by these relatively complex controls and usually give up after a few seconds.

plan, individual offices become visible. Moving the cursor into a room brings the room's details onto the screen. If they choose the wrong room, they merely back out and try another. The lost effort is minimal, and no stigma is attached to the error.

The success of a spatial data management system depends on the designer's skill in choosing icons, graphical representations, and data layouts that are natural and easily understood. Even anxious users enjoy zooming in and out or gliding over data with a joystick, and they quickly demand additional power and data.

Video games. Perhaps the most exciting, well-engineered—certainly, the most successful—application of direct manipulation is in the world of video games. An early, but simple and popular, game called Pong required the user to rotate a knob, which moved a white rectangle on the screen. A white spot acted as a Ping-Pong ball, which ricocheted off the wall and had to be hit back by the movable white rectangle. The user developed skill involving speed and accuracy in placement of the "paddle" to keep the increasingly speedy ball from getting by, while the speaker emitted a ponging sound when the ball bounced. Watching someone else play for 30 seconds was all the training needed to become a competent novice, but many hours of practice were required to become a skilled expert.

Contemporary games such as Missile Command, Donkey Kong, Pac Man, Tempest, Tron, Centipede, or Space Invaders are far more sophisticated in their rules, color graphics, and sound effects (see sidebar below and on facing page). The designers of these games have provided stimulating entertainment, a challenge for novices and experts, and many intriguing lessons in the human factors of interface design—somehow they have found a way to get people to put coins into the sides of computers. The strong attraction of these games contrasts markedly with the anxiety and resistance many users experience toward office automation equipment.

Because their fields of action are abstractions of reality, these games are easily understood—learning is by analogy. A general idea of the game can be gained by watching the on-line automatic demonstration that runs continuously on the screen, and the basic principles can be learned in a few minutes by watching a knowledgeable player. But there are ample complexities to entice many hours and quarters from experts. The range of skill accommodated is admirable.

The commands are physical actions, such as button presses, joystick motions, or knob rotations, whose results appear immediately on the screen. Since there is no syntax, there are no syntax error messages. If users move their spaceships too far left, then they merely use the natural inverse operation of moving back to the right. Error messages are unnecessary because the results of ac-

ınkey Kong, Space Invaders, and Tron (clockwise from above) emplify the lively variety of video games now inviting the er's loose change. As of mid-1981, according to Steve Bloom, thor of *Video Invaders*, more than four billion quarters had en dropped into Space Invaders games around the ırld—that's roughly "one game per earthling."

!eo game photos reprinted courtesy of *IEEE Spectrum*. For a more complete ıort on the topic, see "Video Games: The Electronic Big Bang" by Tekla 'ry, Carol Truxal, and Paul Wallich in *IEEE Spectrum*, Vol. 19, No. 12, Dec. 82, pp. 20-33.

tions are so obvious and easily reversed. These principles can be applied to office automation, personal computing, and other interactive environments.

Every game that I have seen keeps a continuous score so that users can measure their progress and compete with their previous performance, with friends, or with the highest scorers. Typically, the 10 highest scorers get to store their initials in the game for regular display, a form of positive reinforcement that encourages mastery. Malone's[4] and our own studies with elementary school children have shown that continuous display of scores is extremely valuable. Machine-generated value judgments —"Very good" or "You're doing great!"—are not as effective, since the same score means different things to different people. Users prefer to make their own subjective judgments and may perceive machine-generated messages as an annoyance and a deception.

Carroll and Thomas[5] draw productive analogies between game-playing environments and application systems. However, game players seek entertainment and the challenge of mastery, while application-system users focus on the task and may resent forced learning of system constraints. The random events that occur in most games are meant to challenge the user, but predictable system behavior is preferable in nongame designs. Game players compete with the system, but application-system users apparently prefer a strong internal locus of control, which gives them the sense of being in charge.

The pleasure in using these systems stems from the capacity to manipulate the object of interest directly and to generate multiple alternatives rapidly.

Computer-aided design/manufacturing. Many computer-aided design systems for automobiles, electronic circuitry, architecture, aircraft, or newspaper layout use direct manipulation principles. The operator may see a schematic on the screen and with the touch of a lightpen can move resistors or capacitors into or out of the proposed circuit. When the design is complete, the computer can provide information about current, voltage drops, fabrication costs, and warnings about inconsistencies or manufacturing problems. Similarly, newspaper layout artists or automobile body designers can try multiple designs in minutes and record promising approaches until a better one is found.

The pleasure in using these systems stems from the capacity to manipulate the object of interest directly and to generate multiple alternatives rapidly. Some systems have complex command languages, but others have moved to cursor action and graphics-oriented commands.

Another, related application is in computer-aided manufacturing and process control. Honeywell's process control system provides an oil refinery, paper mill, or power utility plant manager with a colored schematic view of the plant. The schematic may be on eight displays, with red lines indicating a sensor value that is out of normal range. By pressing a single numbered button (there are no commands to learn or remember), the operator can get a more detailed view of the troublesome component and, with a second press, move the tree structure down to examine individual sensors or to reset valves and circuits.

The design's basic strategy precludes the necessity of recalling complex commands in once-a-year emergency conditions. The plant schematic facilitates problem solving by analogy, since the link between real-world high temperatures or low pressures and screen representations is so close.

Further examples. Driving an automobile is my favorite example of direct manipulation. The scene is directly visible through the windshield, and actions such as braking or steering have become common skills in our culture. To turn to the left, simply rotate the steering wheel to the left. The response is immediate, and the changing scene provides feedback to refine the turn. Imagine trying to turn by issuing a LEFT 30 DEGREES command and then issuing another command to check your position, but this is the operational level of many office automation tools today.

The term direct manipulation accurately describes the programming of some industrial robots. Here, the operator holds the robot's "hand" and guides it through a spray painting or welding task while the controlling computer records every action. The control computer then repeats the action to operate the robot automatically.

A large part of the success and appeal of the Query-by-Example[6] approach to data manipulation is due to its direct representation of relations on the screen. The user moves a cursor through the columns of the relational table and enters examples of what the result should look like. Just a few single-letter keywords supplement this direct manipulation style. Of course, complex Booleans or mathematical operations require knowledge of syntactic forms. Still, the basic ideas and language facilities can be learned within a half hour by many nonprogrammers. Query-by-Example succeeds because novices can begin work with just a little training, yet there is ample power for the expert. Directly manipulating the cursor across the relation skeleton is a simple task, and how to provide an example that shows the linking variable is intuitively clear to someone who understands tabular data. Zloof[7] recently expanded his ideas into Office-by-Example, which elegantly integrates database search with word processing, electronic mail, business graphics, and menu creation.

Designers of advanced office automation systems have used direct manipulation principles. The Xerox Star[8] offers sophisticated text formatting options, graphics, multiple fonts, and a rapid, high-resolution, cursor-based user interface. Users can drag a document icon and drop it into a printer icon to generate a hardcopy printout. Apple's recently announced Lisa system elegantly applies many of the principles of direct manipulation.

Researchers at IBM's Yorktown Heights facility have proposed a future office system, called Pictureworld, in which graphic icons represent file cabinets, mailboxes, notebooks, phone messages, etc. The user could compose a memo on a display editor and then indicate distribution and filing operations by selecting from the menu of icons. In another project, Yedwab et al.[9] have described a generalized office system, which they call the "automated desk."

Direct manipulation can be applied to replace traditional question-and-answer computer-assisted instruction with more attractive alternatives. Several CDC Plato lessons employ direct manipulation concepts, enabling students to trace inherited characteristics by breeding drosophilla, perform medical procedures to save an emergency room patient, draw and move shapes by finger touches, do chemistry lab projects (see Figure 4), or play games.

Explanations of direct manipulation

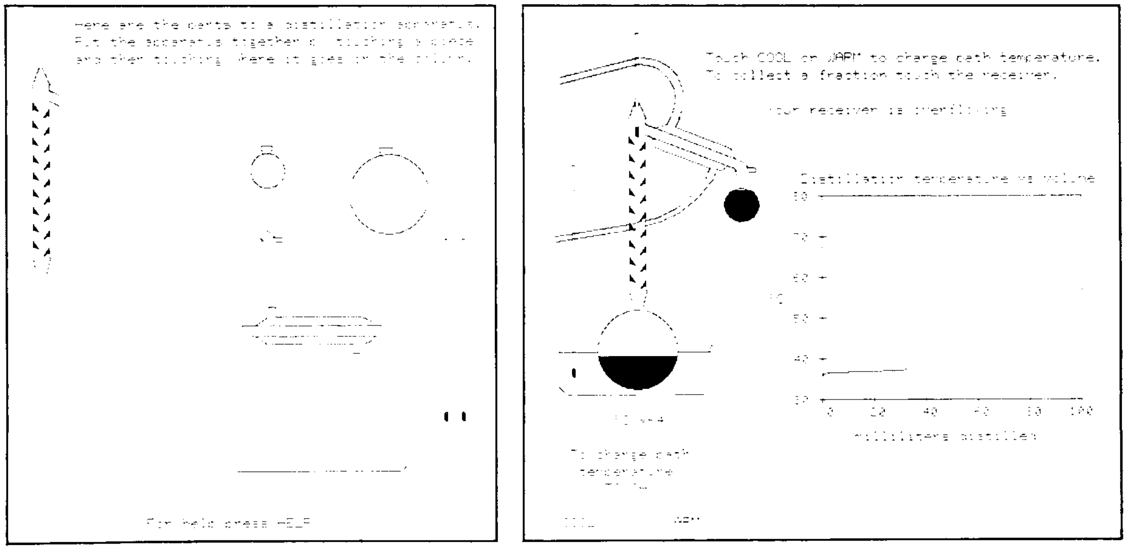
Several people have attempted to describe the component principles of direct manipulation. "What you see is what you get," is a phrase used by Don Hatfield of IBM and others to describe the general approach. Hatfield is applying many direct manipulation principles in his work on an advanced office automation system. Expanding Hatfield's premise, Harold Thimbleby of the University of York, England, suggests, "What you see is what you have got." The display should indicate a complete image of what the current status is, what errors have occurred, and what actions are appropriate, according to Thimbleby.

Another imaginative observer of interactive system designs, Ted Nelson,[10] has noticed user excitement over interfaces constructed by what he calls the principle of "virtuality"—a representation of reality that can be manipulated. Rutkowski[11] conveys a similar concept in his principle of transparency: "The user is able to apply intellect directly to the task; the tool itself seems to disappear." MacDonald[12] proposes "visual programming" as a solution to the shortage of application progammers. He feels that visual programming speeds system construction and allows end users to generate or modify applications systems to suit their needs.

Each of these writers has helped increase awareness of the new form that is emerging for interactive systems. Much credit also goes to individual designers who have created systems exemplifying aspects of direct manipulation.

Problem-solving and learning research. Another perspective on direct manipulation comes from psychology literature on problem solving. It shows that suitable representations of problems are crucial to solution finding and to learning.

Polya[13] suggests drawing a picture to represent mathematical problems. This approach is in harmony with Maria Montessori's teaching methods for children.[14] She proposed use of physical objects such as beads or wooden sticks to convey mathematical principles such as addition, multiplication, or size comparison. Bruner[15] extends the physical representation idea to cover polynomial factoring and other mathematical principles. In a recent experiment, Carroll, Thomas, and Malhotra[16] found that subjects given a spatial representation solved problems more rapidly and successfully than subjects given an isomorphic problem with temporal representa-



Here are the parts to a distillation apparatus. Put the apparatus together by touching a piece and then touching where it goes in the picture.

For help press HELP

Touch COOL or WARM to change bath temperature. To collect a fraction touch the receiver.

Your receiver is overflowing

To change bath temperature

Figure 4. Computer-assisted instruction can become more appealing with direct manipulation, rather than simple question and answer scenarios. This CDC Plato lesson written by Stanley Smith of the Department of Chemistry at the University of Illinois allows students to construct a distillation apparatus by proper finger actions on a touch-sensitive screen (figure at left). Once the student has assembled the apparatus and begun the experiment, the real-time display gives a realistic view of the process with the graph of distillation temperature vs. volume. The student controls the experiment by touching light buttons. The figure at right shows that the student experimenter has gotten into trouble.

tion. (Deeper understanding of visual perception can be obtained from Arnheim[17] and McKim.[18])

Physical, spatial, or visual representations are also easier to retain and manipulate. Wertheimer[19] found that subjects who memorized the formula for the area of a parallelogram, $A = h \times b$, mastered such calculations rapidly. On the other hand, subjects who were given a structural explanation (cut a triangle from one end and place it on the other) retained the knowledge and applied it in similar circumstances more effectively. In plane geometry theorem proving, a spatial representation facilitates discovery of proof procedures more than an axiomatic representation. The diagram provides heuristics that are difficult to extract from the axioms. Similarly, students of algebra are often encouraged to draw a picture to represent a word problem.

Papert's Logo language[20] creates a mathematical microworld in which the principles of geometry are visible. Influenced by the Swiss psychologist Jean Piaget's theory of child development, Logo offers students the opportunity to create line drawings with an electronic turtle displayed on a screen. In this environment, users can receive rapid feedback about their programs, can easily determine what has happened, can quickly spot and repair errors, and can experience creative satisfaction.

Problems with direct manipulation. Some professional programming tasks can be aided by the use of graphic representations such as high-level flowcharts, record structures, or database schema diagrams, but additional effort may be required to absorb the rules of the representation. Graphic representations can be especially helpful when there are multiple relationships among objects and when the representation is more compact than the detailed object. In these cases, selectively screening out detail and presenting a suitable abstraction can facilitate performance.

However, using spatial or graphic representations of the problem does not necessarily improve performance. In a series of studies, subjects given a detailed flowchart did no better in comprehension, debugging, or modification than those given the code only.[21] In a program comprehension task, subjects given a graphic representation of control flow or data structure did no better than those given a textual description.[22] On the other hand, subjects given the data structure documentation consistently did better than subjects given the control flow documentation. This study suggests that the content of graphic representations is a critical determinant of their utility. The wrong information, or a cluttered presentation, can lead to greater confusion.

A second problem is that users must learn the meaning of the components of the graphic representation. A graphic icon, although meaningful to the designer, may require as much—or more—learning time as a word. Some airports serving multilingual communities use graphic icons extensively, but their meaning may not be obvious. Similarly, some computer terminals designed for international use have icons in place of names, but the meaning is not always clear.

A third problem is that the graphic representation may be misleading. The user may rapidly grasp the analogical representation, but then make incorrect conclusions about permissible operations. Designers must be cautious in selecting the displayed representation and the operations. Ample testing must be carried out to refine the representation and minimize negative side effects.

A fourth problem is that graphic representations may take excessive screen display space. For experienced users, a tabular textual display of 50 document names is far more appealing than only 10 document graphic icons with the names abbreviated to fit the icon size. Icons should be evaluated first for their power in displaying static information about objects and their relationship, and second for their utility in the dynamic processes of selection, movement, and deletion.

Choosing the right representations and operations is not easy. Simple metaphors, analogies, or models with a minimal set of concepts seem most appropriate.

Choosing the right representations and operations is not easy. Simple metaphors, analogies, or models with a minimal set of concepts seem most appropriate. Mixing metaphors from two sources adds complexity, which contributes to confusion. The emotional tone of the metaphor should be inviting rather than distasteful or inappropriate[16]—sewage disposal systems are an inappropriate metaphor for electronic message systems. Since users may not share the designer's metaphor, analogy, or conceptual model, ample testing is required.

The syntactic/semantic model. The attraction of systems that use principles of direct manipulation is confirmed by the enthusiasm of their users. The designers of the examples given had an innovative inspiration and an intuitive grasp of what users wanted. Each example has features that could be criticized, but it seems more productive to construct an integrated portrait of direct manipulation:

- Continuous representation of the object of interest.
- Physical actions (movement and selection by mouse, joystick, touch screen, etc.) or labeled button presses instead of complex syntax.
- Rapid, incremental, reversible operations whose impact on the object of interest is immediately visible.
- Layered or spiral approach to learning that permits usage with minimal knowledge. Novices can learn a modest and useful set of commands, which they can exercise till they become an "expert" at level 1 of the system. After obtaining reinforcing feedback from successful operation, users can gracefully expand their knowledge of features and gain fluency.[23]

By using these four principles, it is possible to design systems that have these beneficial attributes:

- Novices can learn basic functionality quickly, usually through a demonstration by a more experienced user.

- Experts can work extremely rapidly to carry out a wide range of tasks, even defining new functions and features.
- Knowledgeable intermittent users can retain operational concepts.
- Error messages are rarely needed.
- Users can immediately see if their actions are furthering their goals, and if not, they can simply change the direction of their activity.
- Users experience less anxiety because the system is comprehensible and because actions are so easily reversible.
- Users gain confidence and mastery because they initiate an action, feel in control, and can predict system responses.

My own understanding of direct manipulation was facilitated by considering the syntactic/semantic model of user behavior. The cognitive model was first developed in the context of programming language experimentation[24,25] and has been applied to database query language questions.[26]

The basic idea is that there are two kinds of knowledge in long-term memory: syntactic and semantic (see Figure 5).

Syntactic knowledge. In a text editor, syntactic knowledge—the details of command syntax—include permissible item delimiters (space, comma, slash, or colon), insertion of a new line after the third line (I3, I 3, or 3I), or the keystroke necessary for erasing a character (delete key, CONTROL-H, or ESCAPE). This knowledge is arbitrary and therefore acquired by rote memorization. Syntactic knowledge is volatile in memory and easily forgotten unless frequently used.[27] This knowledge is system dependent with some possible overlap among systems.

Semantic knowledge. The concepts or functionality—semantic knowledge—are hierarchically structured from low-level functions to higher level concepts. In text editors, lower level functions might be cursor movement, insertion, deletion, changes, text copying, centering, and indentation. These lower level concepts are close to the syntax of the command language. A middle-level semantic concept for text editing might be the process for correcting a misspelling: produce a display of the misspelled word, move the cursor to the appropriate spot, and issue the change command or key in the correct characters. A higher level concept might be the process for moving a sentence from one paragraph to another: move the cursor to the beginning of the sentence, mark this position, move the cursor to the end of the sentence, mark this second position, copy the sentence to a buffer area, clean up the source paragraph, move the cursor to the target location, copy from the buffer, check that the target paragraph is satisfactory, and clear the buffer area.

The higher level concepts in the problem domain (moving a sentence) are decomposed, by the expert user, top-down into multiple, lower level concepts (move cursor, copy from buffer, etc.) closer to the program or syntax domain. Semantic knowledge is largely system independent; text editing functions (inserting/deleting lines, moving sentences, centering, indenting, etc.) are generally available in text editors, although the syntax varies. Semantic knowledge, which is acquired through general explanation, analogy, and example, is easily anchored to familiar concepts and is therefore stable in memory.

The command formulation process in the syntactic/semantic model proceeds from the user's perception of the task in the high-level problem domain to the decomposition into multiple, lower level semantic operations and the conversion into a set of commands. The syntax of text editors may vary, but the decomposition from problem domain into low-level semantics is largely the same. At the syntax level the user must recall whether spaces are permitted, whether program function keys are available, or whether command abbreviations are permitted.

As a user of a half-dozen text editors during a week, I am very aware of the commonality of my thought processes in problem solving and the diversity of syntactic forms with which I must cope. Especially annoying are syntactic clashes such as the different placement of special characters on keyboards, the multiple approaches to backspacing (backspace key, cursor control key, or a mouse), and the fact that one text editor uses "K" for keeping a file while another uses "K" for killing a file.

Implications of the syntactic/semantic model. Novices begin with a close link between syntax and semantics; their attention focuses on the command syntax as they seek to remember the command functions and syntax. In fact, for novice users, the syntax of a precise, concise

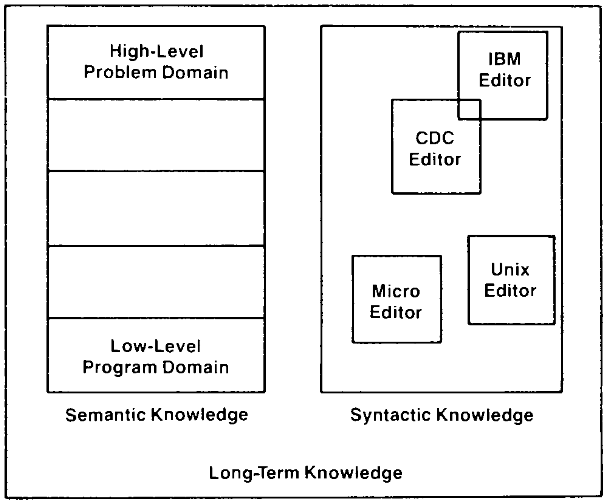


Figure 5. The semantic knowledge in long-term memory goes from high-level problem domain concepts down to numerous low-level program domain details. Semantic knowledge is well-structured, relatively stable, and meaningfully acquired. Syntactic knowledge is arbitrary, relatively volatile unless frequently rehearsed, and acquired by rote memorization. There is usually little overlap between the syntax of different text editors, but they often share semantic concepts about inserting, deleting, and changing lines of text.

command language provides the cues for recalling the semantics. Novices review the command names, in their memory or in a manual, which act as the stimuli for recalling the related semantics. Each command is then evaluated for its applicability to the problem. Novices may have a hard time figuring out how to move a sentence of text, even if they understand each of the commands. Novices using editors that have a "CHANGE /old string/new string/" command must still be taught how to use this command to delete a word or insert a word into a line.

Manuals that have alphabetically arranged sections make it difficult for the novice to anchor material to familiar concepts.

As users gain experience, they increasingly think in higher level semantic terms, which are freer from the syntactic detail and more system independent. In addition to facilitating learning, direct manipulation of a visual representation may aid retention.

The syntactic/semantic model suggests that training manuals should be written from the more familiar, high-level, problem domain viewpoint. The titles of sections should describe problem domain operations that the user deals with regularly. Then the details of the commands used to accomplish the task can be presented, and finally, the actual syntax can be shown. Manuals that have alphabetically arranged sections devoted to each command are very difficult for the novice to learn from, because it is difficult to anchor the material to familiar concepts.

The success of direct manipulation is understandable in the context of the syntactic/semantic model. The object of interest is displayed so that actions are directly in the high-level problem domain. There is little need for decomposition into multiple commands with a complex syntactic form. On the contrary, each command produces a comprehensible action in the problem domain that is immediately visible. The closeness of the problem domain to the command action reduces operator problem-solving load and stress.

Dealing with representations of objects may be more "natural" and closer to innate human capabilities: action and visual skills emerged well before language in human evolution. Psychologists have long known that spatial relationships and actions are more quickly grasped with visual rather than linguistic representations. Furthermore, intuition and discovery are often promoted by suitable visual representations of formal mathematical systems.

Piaget described four stages of growth: sensorimotor (from birth to approximately 2 years), preoperational (2 to 7 years), concrete operational (7 to 11 years), and formal operations (beginning at approximately 11 years).[28] Physical actions on an object are comprehensible during the concrete operational stage, and children acquire the concept of conservation or invariance. At around age 11, children enter the formal operations stage of symbol manipulation to represent actions on objects. Since mathematics and programming require abstract thinking, they are difficult for children, and a greater effort must be made to link the symbolic representation to the actual object. Direct manipulation is an attempt to bring activity to the concrete operational stage or even to the preoperational stage, thus making some tasks easier for children and adults.

It is easy to envision direct manipulation in cases where the physical action is confined to a small number of objects and simple commands, but the approach may be unsuitable for some complex applications. On the other hand, display editors provide impressive functionality in a natural way. The limits of direct manipulation will be determined by the imagination and skill of the designer. With more examples and experience, researchers should be able to test competing theories about the most effective metaphors or analogies. Familiar visual analogies may be more appealing in the early stages of learning the system, while more specific abstract models may be more useful during regular use.

The syntactic/semantic model provides a simple model of human cognitive activity. It must be refined and extended to enhance its explanatory and predictive power. Empirical tests and careful measurements of human performance with a variety of systems are needed to validate the improved model. Cognitive models of user behavior and mental models or system images of computer-supplied functions are rapidly expanding areas of research in computer science and psychology.

Potential applications of direct manipulation

The trick in creating a direct manipulation system is to come up with an appropriate representation or model of reality. I found it difficult to think about information problems in a visual form, but with practice it became more natural. With many applications, the jump to a visual language was initially a struggle, but later I could hardly imagine why anyone would want to use a complex syntactic notation to describe an essentially visual process.

One application that we explored was a personal address list program that displays a Rolodex-like device (see Figure 6). The most recently retrieved address card appears on the screen, and the top line of the next two appear behind, followed by the image of a pack of remaining cards. As the joystick is pushed forward, the Rolodex appears to rotate and successive cards appear in front. As the joystick is pushed further, the cards pass by more quickly; as the joystick is reversed, the direction of movement reverses. To change an entry, users merely move the cursor over the field to be updated and and type the correction. To delete an entry, users merely blank out the fields. Blank cards might be left at the top of the file, but when the fields are filled in, proper alphabetic placement is provided. To find all entries with a specific zip code, users merely type the zip code in the proper field and enter a question mark.

Checkbook maintenance and searching might be done in a similar fashion, by displaying a checkbook register with labeled columns for check number, date, payee, and amount. The joystick might be used to scan earlier entries. Changes could be made in place, new entries could be made at the first blank line, and a check mark could be made to indicate verification against a monthly report. Searches for a particular payee could be made by filling in a blank payee field and then typing a question mark.

Bibliographic searching has more elaborate requirements, but a basic system could be built by first showing the user a wall of labeled catalog index drawers. A cursor in the shape of a human hand might be moved over to the section labeled "Author Index" and to the drawer labeled "F-L." Depressing the button on the joystick or mouse would cause the drawer to open up and reveal an array of index cards with tabs offering a finer index. Moving the cursor-finger and depressing the selection button would cause the actual index cards to appear. Depressing the button while holding a card would cause copying of the card into the user's notebook, also represented on the screen. Entries in the notebook might be edited to create a printed bibliography or combined with other entries to perform set intersections or unions. Copies of entries could be stored on user files or transmitted to colleagues by electronic mail. It is easy to visualize many alternate approaches, so careful design and experimental testing will be necessary to sort out the successful, comprehensible approaches from the idiosyncratic ones.

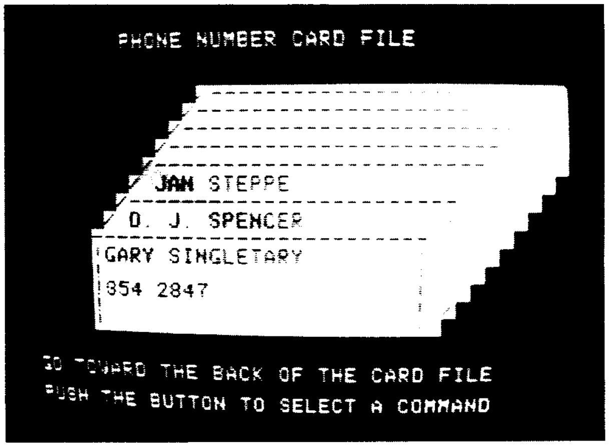
It is possible to apply direct manipulation to environments for which there is no obvious physical parallel. Imagine a job control language that shows the file directory continuously, along with representations of computer components. A new file is created by typing its name into the first free spot in the directory listing. A file name is deleted by blanking it out. Copies are made by locking a cursor onto a file name and dragging it to a picture of a tape drive or a printer. For a hierarchical directory, the roots are displayed until a zoom command causes the next level of the tree to appear. With several presses of the button labeled ZOOM a user should be able to find the right item in the directory, but if he goes down the wrong path, the UNZOOM button will return the previous level. (See Figure 7 for a different approach to hierarchical directories.)

Why not make airline reservations by showing the user a map and prompting for cursor motion to the departing and arriving cities? Then use a calendar to select the date, a clock to indicate the time, and the plane's seating plan (with diagonal lines across already reserved seats) to select a seat.

Why not take inventory by showing the aisles of the warehouse with the appropriate number of boxes on each shelf? McDonald[29] has combined videodisc and computer graphics technology in a medical supply inventory with a visual warehouse display.

Why not teach students about polynomial equations by letting them bend the curves and watch how the coefficients change, where the x-axis intersects, and how the derivative equation reacts?[30]

These ideas are sketches for real systems. Competent designers and implementers must complete the sketches and fill in the details. Direct manipulation has the power



PHONE NUMBER CARD FILE

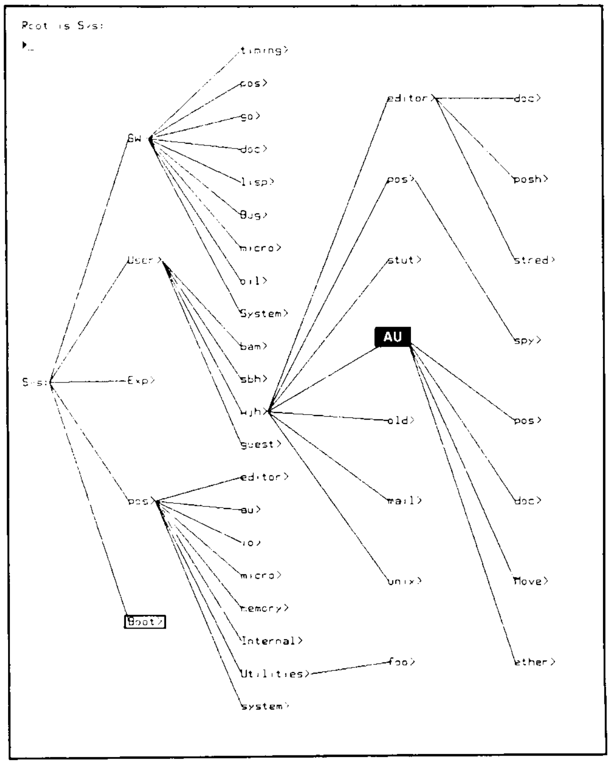
JAN STEPPE

D. J. SPENCER

GARY SINGLETARY
354 2847

GO TOWARD THE BACK OF THE CARD FILE
PUSH THE BUTTON TO SELECT A COMMAND

Figure 6. This electronic Rolodex or phone-number card file gives users rapid control over the card motion by a forward or backward joystick press. Different commands can be displayed by moving the joystick left or right. The lively motion of the cards and the natural commands appeal to many users. Implemented by Gary Patterson in Basic on an Apple II, this system was part of a course project at the University of Maryland.



Root is Sys:

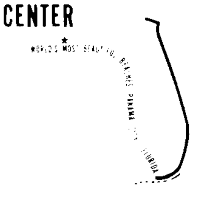
Figure 7. The Dirtree (for directory tree) program on the Perq computer of Three Rivers Computer Corporation is built from left to right by puck selections. The details of lower level directories appear, and the items can then be selected by moving a cursor onto the item. In this figure, the current item is AU, shown in inverse video, but the user has moved the cursor to Boot, which is shown with a box around it. If the button on the puck is pressed, Boot would become the current item. (Figure courtesy of Three Rivers Computer Corporation)

to attract users because it is comprehensible, natural, rapid, and even enjoyable. If actions are simple, reversibility ensured, and retention easy, then anxiety recedes and satisfaction flows in.

The tremendous growth of interest in interactive system design issues in the research community is encouraging. Similarly, the increased concern for improved human engineering in commercial products is a promising sign. Academic and industrial researchers are applying controlled, psychologically oriented experimentation[25] to develop a finer understanding of human performance and to generate a set of practical guidelines. Commercial designers and implementers are eagerly awaiting improved guidelines and increasingly using pilot studies and acceptance tests to refine their designs.

Interactive systems that display a representation of the object of interest and permit rapid, incremental, reversible operations through physical actions rather than command syntax are attracting enthusiastic users. Immediate visibility of the results of operations and a layered or spiral approach to learning contribute to the attraction. Each of these features needs research to refine our understanding of its contributions and limitations. But even while such research is in progress, astute designers can explore this approach.

The future of direct manipulation is promising. Tasks that could have been performed only with tedious command or programming languages may soon be accessible through lively, enjoyable interactive systems that reduce learning time, speed performance, and increase satisfaction. ■



Acknowledgments

I am grateful to the Control Data Corporation for partial support (grant 80M15) of my work and to the University of Maryland Computer Science Center for computer resources to prepare this report. I thank Gordon Braudaway, Jim Foley, John Gannon, Roger Knights, John Lovgren, Harlan Mills, Phyllis Reisner, Sherry Weinberg, and Mark Weiser for their constructive and supportive comments on draft versions. Gio Wiederhold, Stephen Yau, and the reviewers provided useful guidance in shaping the final article.

References

1. Teresa L. Roberts, "Evaluation of Computer Text Editors," PhD dissertation, Stanford University, 1980. Available from University Microfilms, Ann Arbor, Michigan, order number AAD 80-11699.
2. John M. Carroll, "Learning, Using and Designing Command Paradigms," *Human Learning*, Vol. 1, No. 1, 1982, pp. 31-62.
3. Christopher F. Herot, "Spatial Management of Data," *ACM Trans. Database Systems*, Vol. 5, No. 4, Dec. 1980, pp. 493-513.
4. Thomas W. Malone, "What Makes Computer Games Fun?" *Byte*, Vol. 6, No. 12, Dec. 1981, pp. 258-277.
5. John M. Carroll and John C. Thomas, "Metaphor and the Cognitive Representation of Computing Systems, *IEEE Trans. Systems, Man, and Cybernetics*, Vol. SMC-12, No. 2, Mar./Apr. 1982, pp. 107-116.
6. Moshe M. Zloof, "Query-by-Example," *AFIPS Conf. Proc.*, Vol. 44, 1975 NCC, AFIPS Press, Montvale, N.J. 1975.
7. Moshe M. Zloof, "Office-by-Example: A Business Language that Unifies Data and Word Processing and Electronic Mail, *IBM Sys. J.*, Vol. 21, No. 3, 1982, pp. 272-304.
8. Cranfield Smith et al., "Designing the Star User Interface," *Byte*, Vol. 7, No. 4, Apr. 1982, pp. 242-282.
9. Laura Yedwab, Christopher F. Herot, and Ronni L. Rosenberg, "The Automated Desk," *Sigsmall Newsletter*, Vol. 7, No. 2, Oct. 1981, pp. 102-108.
10. Ted Nelson, "Interactive Systems and the Design of Virtuality," *Creative Computing*, Vol. 6, No. 11, Nov. 1980, pp. 56 ff., and Vol. 6, No. 12, Dec. 1980, pp. 94 ff.
11. Chris Rutkowski, "An Introduction to the Human Applications Standard Computer Interface, Part 1: Theory and Principles," *Byte*, Vol. 7, No. 11, Oct. 1982, pp. 291-310.
12. Alan MacDonald, "Visual Programming," *Datamation*, Vol. 28, No. 11, Oct. 1982, pp. 132-140.
13. George Polya, *How to Solve It*, Doubleday, New York, 1957.
14. Maria Montessori, *The Montessori Method*, Schocken, New York, 1964.

15. James Bruner, *Toward a Theory of Instruction,* Harvard University Press, Cambridge, Mass., 1966.

16. John M. Carroll, J. C. Thomas, and A. Malhotra, "Presentation and Representation in Design Problem-Solving," *British J. Psych.*, Vol. 71, 1980, pp. 143-153.

17. Rudolf Arnheim, *Visual Thinking,* University of California Press, Berkeley, Calif., 1972.

18. Robert H. McKim, *Experiences in Visual Thinking,* Brooks/Cole Publishing Co., Monterey, Calif., 1972.

19. Max Wertheimer, *Productive Thinking,* Harper and Row, New York, 1959.

20. Seymour Papert, *Mindstorms: Children, Computers, and Powerful Ideas,* Basic Books, Inc., New York, 1980.

21. Ben Shneiderman, R. Mayer, D. McKay, and P. Heller, "Experimental Investigations of the Utility of Detailed Flowcharts in Programming," *Comm. ACM,* Vol. 20, No. 6, June 1977, pp. 373-381.

22. Ben Shneiderman, "Control Flow and Data Structure Documentation: Two Experiments," *Comm. ACM,* Vol. 25, No. 1, Jan. 1982, pp. 55-63.

23. Michael L. Schneider, "Models for the Design of Static Software User Assistance," *Directions in Human-Computer Interaction,* Albert Badre and Ben Shneiderman, eds., Ablex Publishing Co., Norwood, N.J., 1982.

24. Ben Shneiderman and Richard Mayer, "Syntactic/Semantic Interactions in Programmer Behavior: A Model and Experimental Results," *Int'l J. Computer and Information Sciences,* Vol. 8, No. 3, 1979, pp. 219-239.

25. Ben Shneiderman, *Software Psychology: Human Factors in Computer and Information Systems,* Little, Brown and Co., Boston, Mass., 1980.

26. Ben Shneiderman, "A Note on Human Factors Issues of Natural Language Interaction with Database Systems," *Information Systems,* Vol. 6, No. 2, Feb. 1981, pp. 125-129.

27. D. P. Ausubel, *Educational Psychology: A Cognitive Approach,* Holt, Rinehart and Winston, New York, 1968.

28. Richard W. Copeland, *How Children Learn Mathematics,* third ed., MacMillan, New York, 1979.

29. Nancy McDonald, "Multi-media Approach to User Interface," *Human Factors in Interactive Computer Systems,* Yannis Vassiliou, ed., Ablex Publishing Co., Norwood, N.J., to appear in 1983.

30. Ben Shneiderman, "A Computer Graphics System for Polynomials," *The Mathematics Teacher,* Vol. 67, No. 2, Feb. 1974, pp. 111-113.

Ben Shneiderman is an associate professor of computer science at the University of Maryland, where he is pursuing research in the design of interactive computer systems. He is the head of the recently formed Laboratory for Human-Computer Interaction within the Center for Automation Research.

Shneiderman is the author of *Software Psychology: Human Factors in Computer and Information Systems,* the coauthor of several textbooks, and the editor of three collections of papers. He has published more than 80 research journal and conference articles.

APPENDIX E

HUMAN-COMPUTER INTERACTION, 1985, Volume 1, pp. 311-338

Direct Manipulation Interfaces

Edwin L. Hutchins, James D. Hollan, and
Donald A. Norman
University of California, San Diego

ABSTRACT

Direct manipulation has been lauded as a good form of interface design, and some interfaces that have this property have been well received by users. In this article we seek a cognitive account of both the advantages and disadvantages of direct manipulation interfaces. We identify two underlying phenomena that give rise to the feeling of directness. One deals with the information processing distance between the user's intentions and the facilities provided by the machine. Reduction of this distance makes the interface feel direct by reducing the effort required of the user to accomplish goals. The second phenomenon concerns the relation between the input and output vocabularies of the interface language. In particular, direct manipulation requires that the system provide representations of objects that behave as if they are the objects themselves. This provides the feeling of directness of manipulation.

A version of this paper also appears as a chapter in the **book,** ***User** Centered System* ***De**sign: New Perspectives on Human-Computer Interaction* (Norman & Draper, 1986).

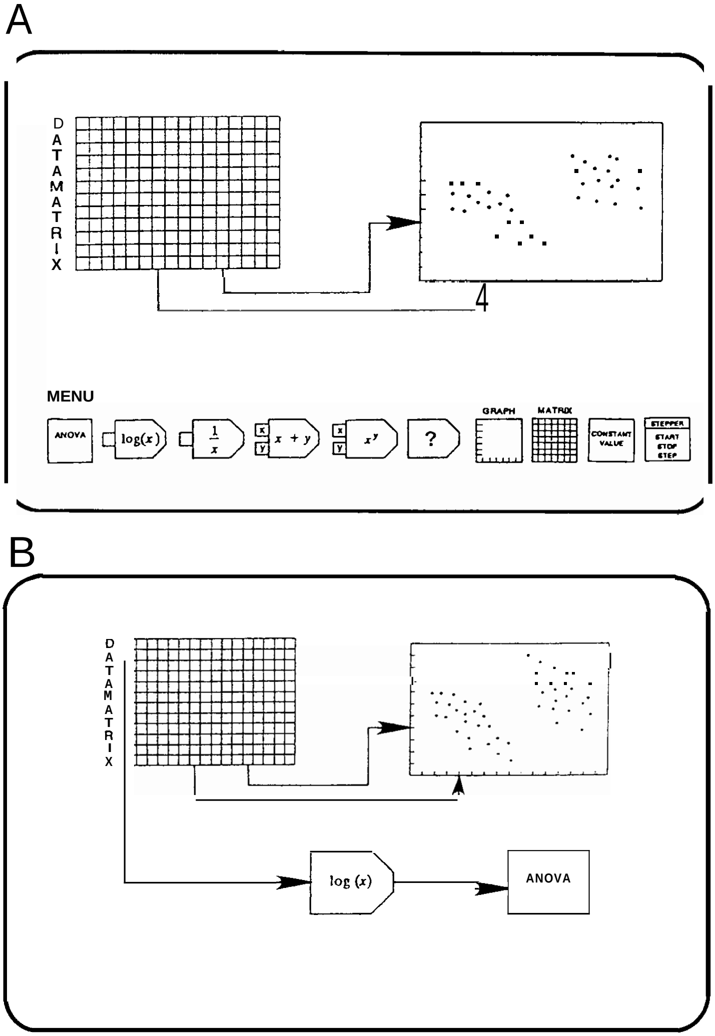
Authors' present address: Edwin L. Hutchins, James D. Hollan, and Donald **A.** Norman, Institute for Cognitive Science, University of California at San Diego, La Jolla, **CA** 92093.

CONTENTS

1. DIRECT MANIPULATION

The best way to describe a direct manipulation interface is by example. Suppose we have a set of data to be analyzed with the numbers stored in matrix form. Their source and meaning are not important for this example. The numbers could be the output of a spreadsheet, a matrix of numerical values from the computations of a conventional programming language, or the results of an experiment. Our goal is to analyze the numbers, to see what relations exist among the rows and columns of the matrix. The matrix of numbers is represented on a computer display screen by an icon. To plot one column against another, simply get a copy of a graph icon, then draw a line from the output of one column to the x-axis input of the graph icon and another line from the output of the second column to the y-axis input (see Figure 1). Not what was wanted? Erase the lines and reconnect them. Want to see other graphs? Make more copies of the graph icons and connect them. Need a logarithmic transformation of one of the axes? Move up a function icon, type in the algebraic function that is desired ($y = log$ x, in this case) and connect it in the desired data stream. Want the analysis of variance of the logarithm of the data? Connect the matrix to the appropriate statistical icons. These examples are illustrated in Figure 1B.

Figure 1. An elementary example of doing simple statistical computations by direct manipulation. (A) The basic components: The data are contained in the matrix, represented by the icon in the upper left corner of the screen. At the bottom of the screen are basic icons that represent possible functions. To use one, a copy of the desired icon is moved to the screen and connected up, as is shown for the graph. **(B)** More complex interconnections, including the use of a logarithmic transformation of the data, a basic statistical package (for means and standard deviations), and an Analysis of Variance Package (ANOVA).



Now consider how we could partition the data. Suppose one result of our analysis was the scatter diagram shown in Figure 2. The straight line that has been fitted through the points is clearly inappropriate. The data fall into two quite different clusters and it would best to analyze each cluster separately. In the actual data matrix, the points that form the two clusters might be scattered randomly throughout the data set. The regularities are apparent only when we plot them. How do we pull out the clusters? Suppose we could simply circle the points of interest in the scatter plot and use each circled set as if it were a new matrix of values, each of which could be analyzed in standard ways, as shown in Figure 2B.

The examples of Figures 1 and 2 illustrate a powerful manipulation medium for computation. The promise of direct manipulation is that instead of an abstract computational medium, all the "programming" is done graphically, in a form that matches the way one thinks about the problem. The desired operations are performed simply by moving the appropriate icons onto the screen and connecting them together. Connecting the icons is the equivalent of writing a program or calling on a set of statistical subroutines, but with the advantage of being able to directly manipulate and interact with the data and the connections. There are no hidden operations, no syntax or command names to learn. What you see is what you get. Some classes of syntax errors are eliminated. For example, you can't point at a nonexistent object. The system requires expertise in the task domain, but only minimal knowledge of the computer or of computing.

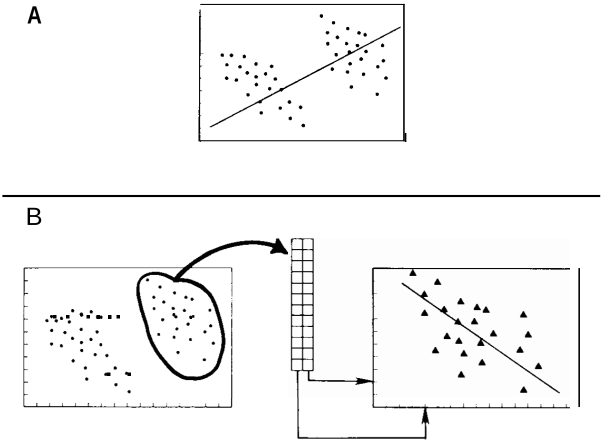
The term ***direct** manipulation* was coined by Shneiderman (1974, 1982, 1983) to refer to systems having the following properties:

1. Continuous representation of the object of interest.
2. Physical actions or labeled button presses instead of complex syntax.
3. Rapid incremental reversible operations whose impact on the object of interest is immediately visible. (Shneiderman, 1982, p. 251)

Direct manipulation interfaces seem remarkably powerful. Shneiderman (1982) has suggested that direct manipulation systems have the following virtues:

1. Novices can learn basic functionality quickly, usually through a demonstration by a more experienced user.
2. Experts can work extremely rapidly to carry out a wide range of tasks, even defining new functions and features.
3. Knowledgeable intermittent users can retain operational concepts.
4. Error messages are rarely needed.
5. Users can see immediately if their actions are furthering their goals, and if not, they can simply change the direction of their activity.

Figure 2. **(A)** The scatter plot formed in Figure 1, along with the best fitting regression line to the data. It is clear that the data really fall into two quite distinct clusters and that it would be best to look at each independently. **(B)** The clusters are analyzed by circling the desired data, then treating the group of circled data as if they were a new matrix of values, which can be treated as a data source and analyzed in standard ways.



6. Users have reduced anxiety because the system is comprehensible and because actions are so easily reversible. (Shneiderman, 1982, p. 251)

Can this really be true? Certainly there must be problems as well as benefits. It turns out that the concept of direct manipulation is complex. Moreover, although there are important benefits there are also costs. Like everything else, direct manipulation systems trade off one set of virtues and vices against another. It is important that we understand these trade-offs. A checklist of surface features is unlikely to capture the real sources of power in direct manipulation interfaces.

1.1. Early Examples of Direct Manipulation

Hints of direct manipulation programming environments have been around for quite some time. The first major landmark is Sutherland's *Sketchpad,* a graphical design program (Sutherland, 1963). Sutherland's goal was *to* devise a program that would make it possible for a person and a computer "to converse rapidly through the medium of line drawings." Sutherland's work is a landmark not only because of historical priority but because of the ideas that he helped develop: He was one of the first to discuss the power of graphical interfaces, the conception of a display as "sheets of paper," the use of pointing devices, the virtues of constraint representations, and the importance of depicting abstractions graphically.

Sutherland's ideas took 20 years to have widespread impact. The lag is perhaps due more to hardware limitations than anything else. Highly interactive, graphical programming requires the ready availability of considerable computational power, and it is only recently that machines capable of supporting this type of computational environment have become inexpensive enough to be generally available. Now we see these ideas in many of the computer-aided design and manufacturing systems, many of which can trace their heritage directly to Sutherland's work. Borning's *ThingLab* program (1979) explored a general programming environment, building upon many of Sutherland's ideas within the Smalltalk programming environment. More recently direct manipulation systems have been appearing with reasonable frequency. For example, *Bill Budge's Pinball Construction Set* (Budge, 1983) permits a user to construct an infinite variety of electronic pinball games by directly manipulating graphical objects that represent the components of the game surface. Other examples exist in the area of intelligent training systems (e.g., the Steamer system of Hollan, Hutchins, & Weitzman, 1984; Hollan, Stevens, & Williams, 1980). Steamer makes use of similar techniques and also provides tools for the construction of interactive graphical interfaces. Finally, spreadsheet programs incorporate many of the essential features of direct manipulation. In the lead article of *Scientific American's* special issue on computer software, Kay (1984) claims that the development of dynamic spreadsheet systems gives strong hints that programming styles are in the offing that will make programming as it has been done for the past 40 years—that is, by composing text that represents instructions—obsolete.

1.2. The Goal: A Cognitive Account of Direct Manipulation

We see promise in the notion of direct manipulation, but as yet we see no explanation of it. There are systems with attractive features, and claims for the benefits of systems that give the user a certain sort of feeling, and even lists of properties that seem to be shared by systems that provide that feeling, but no account of how particular properties might produce the feeling of directness. The purpose of this article is to examine the underlying basis for direct manipulation systems. On the one hand, what is it that provides the feeling of "directness?" Why do direct manipulation systems feel so natural? What is so compelling about the notion? On the other hand, why can using such systems sometimes seem so tedious?

For us, the notion of "direct manipulation" is not a unitary concept, nor even something that can be quantified in itself. It is an orienting notion. "Directness" is an impression or a feeling about an interface. What we seek to do here is to characterize the space of interfaces and see where within that picture the range of phenomena that contribute to the feeling of directness might reside. The goal is to give cognitive accounts of these phenomena. At the root of our approach is the assumption that the feeling of directness results from the commitment of fewer cognitive resources. Or, put the other way around, the need to commit additional cognitive resources in the use of an interface leads to the feeling of indirectness. As we shall see, some of the production of the feeling of directness is due to adaptation by the user, so that the designer can neither completely control the process, nor take full credit for the feeling of directness that may be experienced by the user.

We will not attempt to set down hard and fast criteria under which an interface can be classified as direct or not direct. The sensation of directness is always relative; it is often due to the interaction of a number of factors. There are costs associated with every factor that increases the sensation of directness. At present we know of no way to measure the trade-off values, but we will attempt to provide a framework within which one can say what is being traded off against what.

2. TWO ASPECTS OF DIRECTNESS: DISTANCE AND ENGAGEMENT

There are two distinct aspects of the feeling of directness. One involves a notion of the distance between one's thoughts and the physical requirements of the system under use. A short distance means that the translation is simple and straightforward, that thoughts are readily translated into the physical actions required by the system and that the system output is in a form readily interpreted in terms of the goals of interest to the user. We will use the term ***directness*** to refer to the feeling that results from interaction with an interface. The term *distance* will be used to describe factors which underlie the generation of the feeling of directness.

The second aspect of directness concerns the qualitative feeling of engagement, the feeling that one is directly manipulating the objects of interest. There are two major metaphors for the nature of human-computer interaction, a conversation metaphor and a model-world metaphor. In a system built on the conversation metaphor, the interface is a language medium in which the user and system have a conversation about an assumed, but not explicitly represented world. In this case, the interface is an implied intermediary between the user and the world about which things are said. In a system built on the model-world metaphor, the interface is itself a world where the user can act, and which changes state in response to user actions. The world of interest is explicitly represented and there is no intermediary between user and world. Appropriate use of the model-world metaphor can create the sensation in the user of acting upon the objects of the task domain themselves. We call this aspect of directness *direct engagement.*

2.1. Distance

We call one underlying aspect of directness *distance* to emphasize the fact that directness is never a property of the interface alone, but involves a relationship between the task the user has in mind and the way that task can be accomplished via the interface. Here the critical issues involve minimizing the effort required to bridge the gulf between the user's goals and the way they must be specified to the system.

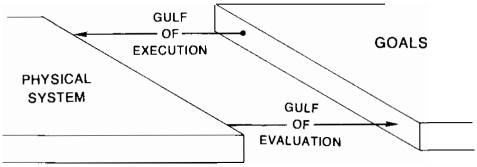
An interface introduces distance to the extent there are gulfs between a person's goals and knowledge and the level of description provided by the systems with which the person must deal. These are referred to as the *gulf of execution* and the *gulf of evaluation* (Figure 3). The gulf of execution is bridged by making the commands and mechanisms of the system match the thoughts and goals of the user. The gulf of evaluation is bridged by making the output displays present a good conceptual model of the system that is readily perceived, interpreted, and evaluated. The goal in both cases is to minimize cognitive effort.

We suggest that the feeling of directness is inversely proportional to the amount of cognitive effort it takes to manipulate and evaluate a system and, moreover, that cognitive effort is a direct result of the gulfs of execution and evaluation. The better the interface to a system helps bridge the gulfs, the less cognitive effort needed and the more direct the resulting feeling of interaction.

2.2. Direct Engagement

The description of the nature of interaction to this point begins to suggest how to make a system less difficult to use, but it misses an important point, a point that is the essence of direct manipulation. The analysis of the execution and evaluation process explains why there is difficulty in using a system, and it says something about what must be done to minimize the mental effort required to use a system. But there is more to it than that. The systems that best exemplify direct manipulation all give the *qualitative* feeling that one is directly engaged with control of the objects—not with the programs, not with the computer, but with the semantic objects of our goals and intentions. This is the feeling that Laurel (1986) discusses: a feeling of first-personness, of direct engagement with the objects that concern us. Are we analyzing data? Then we should be manipulating the data themselves; or if we are designing an analysis of data, we should be manipulating the analytic structures themselves. Are we

Figure 3. **The gulfs of execution and evaluation. Each gulf is unidirectional: The gulf of execution goes from goals to system state; the gulf of evaluation goes from system state to goals.**



playing a game? Then we should be manipulating directly the game world, touching and controlling the objects in that world, with the output of the system responding directly to our actions, and in a form compatible with them.

Historically, most interfaces have been built on the conversation metaphor. There is power in the abstractions that language provides (we discuss some of this later), but the implicit role of interface as an intermediary to a hidden world denies the user direct engagement with the objects of interest. Instead, the user is in direct contact with linguistic structures, structures that can be interpreted as referring to the objects of interest, but that are not those objects themselves. Making the central metaphor of the interface that of the model world supports the feeling of directness. Instead of describing the actions of interest, the user performs those actions. In a conventional interface, the system describes the results of the actions. In a model world the system directly presents the actions taken upon the objects. This change in central metaphor is made possible by relatively recent advances in technology. One of the exciting prospects for the study of direct manipulation is the exploration of the properties of systems that provide for direct engagement.

Building interfaces based on the model-world metaphor requires a special sort of relationship between the input interface language and the output interface language. In particular, the output language must represent its subject of discourse in a way that natural language does not normally do. The expressions of a direct manipulation output language must behave in such a way that the user can assume that they, in some sense, *are* the things they refer to. DiSessa (1985) calls this "naive realism." Furthermore, the nature of the relationship between input and output language must be such that an output expression can serve as a component of an input expression. Draper (1986) has coined the term *inter-referential* 1/0 to refer to relationships between input and output in which an expression in one can refer to an expression in the other. When these conditions are met, it is as if we are directly manipulating the things that the system represents.

Thus, consider a system in which a file is represented by an image on the screen and actions are done by pointing to and manipulating the image. In this case, if we can specify a file by pointing at the screen representation, we have met the goal that an expression in the output language (in this case, an image) be allowed as a component of the input expression (in this case, by pointing at the screen representation). If we ask for a listing of files, we would want the result to be a representation that can, in turn, be used directly to specify the further operations to be done. Notice that this is not how a conversation works. In conversation, one may refer to what has been said previously, but one cannot operate upon what has been said. This requirement does not necessarily imply an interface of pictures, diagrams, or icons. It can be done with words and descriptions. The key properties are that the objects, whatever their form, have behaviors and can be referred to by other objects, and that referring to an object causes it to behave. In the file-listing example, we must be able to use the output expression that represents the file in question as a part of the input expression calling for whatever operation we desire upon that file, and the output expression that represents the file must change as a result of being referred to in this way. The goal is to permit the user to act as if the representation is the thing itself.

These conditions are met in many screen editors when the task is the arrangement of strings of characters. The characters appear as they are typed. They are then available for further operations. We treat them as though they are the things we are manipulating. These conditions are also met in the statistics example with which we opened this article (Figure 1), and in Steamer. The special conditions are not met in file-listing commands on most systems, the commands that allow one to display the names and attributes of file structure. The issue is that the outputs of these commands are simply "names" of the objects, and operating on the names does nothing to the objects to which the names refer. In a direct manipulation situation, we would feel that we had the files in front of us, that the program that "listed" the files actually placed the files before us. Any further operation on the files would take place upon the very objects delivered by the directory-listing command. This would provide the feeling of directly manipulating the objects that were returned.

The point is that when an interface presents a world of behaving objects rather than a language of description, manipulating a representation can have the same effects and the same feel as manipulating the thing being represented. The members of the audience of a well-staged play willfully suspend their beliefs that the players are actors and become directly engaged in the content of the drama. In a similar way, the user of a well-designed model-world interface can willfully suspend belief that the objects depicted are artifacts of some program and can thereby directly engage the world of the objects. This is the essence of the "first-personness" feeling of direct engagement. Let us now return to the issue of distance and explore the ways that an interface can be direct or indirect with respect to a particular task.

3. TWO FORMS OF DISTANCE: SEMANTIC AND ARTICULATORY

Whenever we interact with a device, we are using an interface language. That is, we must use a language to describe to the device the nature of the actions we wish to have performed. This is true regardless ofwhether we are dealing with an interface based on the conversation metaphor or on the model-world metaphor, although the properties of the language in the two cases are different. A description of desired actions is an expression in the interface language.

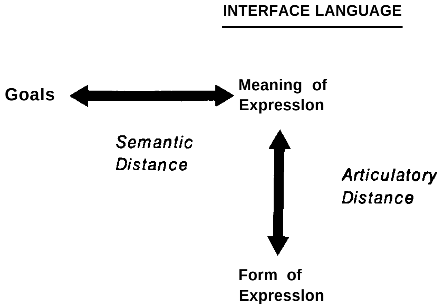
The notion of an interface language is not confined to the everyday meaning of language. Setting a switch or turning a steering wheel can be expressions in an interface language if switch setting or wheel turning are how one specifies the operations that are to be done. After an action has been performed, evaluation of the outcome requires that the device make available some indication of what has happened: that output is an expression in the output interface language. Output interface languages are often impoverished. Frequently the output interface language does not share vocabulary with the input interface language. Two forms of interface language—two dialects, if you will—must exist to span the gulfs between user and device: the input interface language and the output interface language.

Both the languages people speak and computer programming languages are almost entirely symbolic in the sense that there is an arbitrary relationship between the form of a vocabulary item and its meaning. The reference relationship is established by convention and must be learned. There is no way to infer meaning from form for most vocabulary items. Because of the relative independence of meaning and form we describe separately two properties of interface languages: semantic distance and articulatory distance. Figure 4 summarizes the relationship between semantic and articulatory distance. In the following sections we treat each of these distances separately and discuss them in relation to the gulfs of execution and evaluation.

3.1. Semantic Distance

Semantic distance concerns the relation of the meaning of an expression in the interface language to what the user wants to say. Two important questions about semantic distance are (1) *Is it possible to say what one wants to say in this language?* That is, does the language support the user's conception of the task domain? Does it encode the concepts and distinctions in the domain in the same way that the user thinks about them? (2) *Can the things of interest be said concisely?* Can the user say what is wanted in a straightforward fashion, or must the user

Figure 4. Every expression in the interface language has a meaning and a form. Semantic distance reflects the relationship between the user intentions and the meaning of expressions in the interface languages both for input and output. Articulatory distance reflects the relationship between the physical form of an expression in the interaction language and its meaning, again, both for input and output. The easier it is to go from the form **or** appearance of the input **or** output to meaning, the smaller the articulatory distance.



construct a complicated expression to do what appears in the user's thoughts as a conceptually simple piece of work?

Semantic distance is an issue with all languages. Natural languages generally evolve such that they have rich vocabularies for domains that are of importance to their speakers. When a person learns a new language—especially when the language is from a different culture—the new language may seem indirect, requiring complicated constructs to describe things the learner thinks should be easy to say. But the differences in apparent directness reflect differences in what things are thought important in the two cultures. Natural languages can and do change as the need arises. This occurs through the introduction of new vocabulary or by changing the meaning of existing terms. The result is to make the language semantically more direct with respect to the topic of interest.

3.2. Semantic Distance in the Gulfs of Execution and Evaluation

> Beware the Turing tar-pit in which everything is possible but nothing of interest is easy (Perlis, 1982, p. 10).

The Gulf of Execution

At the highest level of description, a task may be described by the user's intention: "compose this piece" or "format this paper." At the lowest level of description, the performance of the task consists of the shuffling of bits inside the machine. Between the interface and the low-level operations of the machine is the system-provided task-support structure that implements the expressions in the interface language. The situation that Perlis (1982) called the "Turing tar-pit" is one in which the interface language lies near or at the level of bit shuffling of a very simple abstract machine. In this case, the entire burden of spanning the gulf from user intention to bit manipulation is carried by the user. The relationship between the user's intention and the organization of the instructions given to the machine is distant, complicated, and hard to follow. Where the machine is of minimal complexity, as is the case with the Turing machine example, the wide gulf between user intention and machine instructions must be filled by the user's extensive planning and translation activities. These activities are difficult and rife with opportunities for error.

Semantic directness requires matching the level of description required by the interface language to the level at which the person thinks of the task. It is always the case that the user must generate some information-processing structure to span the gulf. Semantic distance in the gulf of execution reflects how much of the required structure is provided by the system and how much by the user. The more that the user must provide, the greater the distance to be bridged.

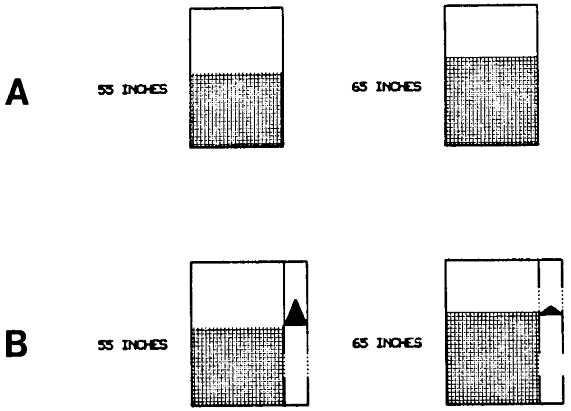
The Gulf of Evaluation

On the evaluation side, semantic distance refers to the amount of processing structure that is required for the user to determine whether the goal has been achieved. If the terms of the output are not those of the user's intention, the user will be required to translate the output into terms that are compatible with the intention in order to make the evaluation. For example, suppose a user's intent is to control how fast the water level in a tank rises. The user does some controlling action and observes the output. But if the output only shows the current value, the user has to observe the value over time and mentally compare the values at different times to see what the rate of change is (see Figure 5). The information needed for the evaluation is in the output, but it is not there in a form that directly fits the terms of the evaluation. The burden is on the user to perform the required transformations, and that requires effort. Suppose the rate of change were directly displayed, as in Figure 5B. This indication reduces the mental workload, making the semantic distance between intentions and output language much shorter.

3.3. Reducing the Semantic Distance That Must Be Spanned

Figure 5 provides one illustration of how semantic distance can be changed. In general, there are only two basic ways to reduce the distance, one from the system side (requiring effort on the part of the system designer), the other from the user side (requiring effort on the part of the user). Each direction of bridge building has several components. Here let us consider the following possibili-

Figure **5.** Matching user's intentions by appropriate output language. The user attempts to control the rate at which the water level in the tank is rising. In **(A),** the only indication is a meter that shows the current level. This requires the user to observe the meter over time and to do a mental computation on the observations. **(B)** shows a display that is more semantically direct: The rate of change is graphically indicated. (These illustrations are from the working Steamer system of Hollan, Hutchins, & Weitzman, **1984.)**



ties: (1)The designer can construct higher-level and specialized languages that move toward the user, making the semantics of the input and output languages match that of the user. (2) The user can develop competence by building new mental structures to bridge the gulfs. In particular, this requires the user to automate the response sequence and to learn to think in the same language as that required by the system.

Higher-Level Languages

One way to bridge the gulf between the intentions of the user and the specifications required by the computer is well known: Provide the user with a higher-level language, one that directly expresses frequently encountered structures of problem decomposition. Instead of requiring the complete decomposition of the task to low-level operations, let the task be described in the same language used within the task domain itself. Although the computer still requires low-level specification, the job of translating from the domain language to the programming language can be taken over by the machine itself.

This implies that designers ofhigher-level languages should consider how to develop interface languages for which it will be easy for the user to create the mediating structure between intentions and expressions in the language. One way to facilitate this process is to provide consistency across the interface surface. That is, if the user builds a structure to make contact with some part of the interface surface, a savings in effort can be realized if it is possible to use all or part of that same structure to make contact with other areas.

The result of matching a language to the task domain brings both good news and bad news. The good news is that tasks are easier to specify. Even if considerable planning is still required to express a task in a high-level language, the amount of planning and translation that can be avoided by the user and passed off to the machine can be enormous. The bad news is that the language has lost generality. Tasks that do not easily decompose into the terms of the language may be difficult or impossible to represent. In the extreme case, what can be done is easy to do, but outside that specialized domain, nothing can be done.

The power of a specialized language system derives from carefully specified primitive operations, selected to match the predicted needs of the user, thus capturing frequently occurring structures of problem decomposition. The trouble is that there is a conflict between generality and matching to any specific problem domain. Some high-level languages and operating systems have attempted to close the gap between user intention and the interaction language while preserving freedom and ease of general expression by allowing for extensibility of the language or operating system. Such systems allow the users to move the interface closer to their conception of the task.

The Lisp language and the UNIX operating system serve as examples of this phenomenon. Lisp is a general-purpose language, but one that has extended itself to match a number of special high-level domains. **As** a result, Lisp can be thought of as having numerous levels on top of the underlying language kernel. There is a cost to this method. As more and more specialized domain levels get added, the language system gets larger and larger, becoming more clumsy to use, more expensive to support, and more difficult to learn. Just look at any of the manuals for the large Lisp systems (Interlisp, Zetalisp) to get a feel for the complexity involved. The same is true for the UNIX operating system, which started out with a number of low-level, general primitive operations. Users were allowed (and encouraged) to add their own, more specialized operations, or to package the primitives into higher-level operations. The results in all these cases are massive systems that are hard to learn and that require a large amount of support facilities. The documentation becomes huge, and not even system experts know all that is present. Moreover, the difficulty of maintaining such a large system increases the burden on everyone, and the possibility of having standard interfaces to each specialized function has long been given up.

The point is that as the interface approaches the user's intention end of the gulf, functions become more complicated and more specialized in purpose. Because of the incredible variety of human intentions, the lexicon of a language that aspires to both generality of coverage and domain-specific functions can grow very large. In any of the modern dialects of Lisp one sees a microcosm of the argument about high-level languages in general. The fundamentals of the language are simple, but a great deal of effort is required to do anything useful at the low level of the language itself. Higher-level functions written in terms of lower-level ones make the system easier to use when the functions match intentions, but in doing so they may restrict possibilities, proliferate vocabulary, and require that a user know an increasing amount about the language of interaction rather than the domain of action.

Make the Output Show Semantic Concepts Directly

An example of reducing semantic distance on the output side is provided by the scenario of controlling the rate of filling a water tank, described in Figure 5. In that situation, the output display was modified to show rate of flow directly, something normally not displayed but instead left *to* the user to compute mentally.

In similar fashion, the change from line-oriented text editors to screen-oriented text editors, where the effects of editing commands can be seen instantly, is another example of matching the display to the user's semantics. In general, the development of WYSIWYG ("What You See Is What You Get") systems provides other examples. And finally, spreadsheet programs have been valuable, in part because their output format continually shows the state of the system as values are changed.

The attempt to develop good semantic matches with the system output confronts the same conflict between generality and power faced in the design of input languages. If the system is too specific and specialized, the output displays lack generality. If the system is too rich, the user has trouble learning and selecting among the possibilities. One solution for both the output and input problem is to abandon hope of maintaining general computing and output ability and to develop special-purpose systems for particular domains or tasks. In such a world, the location of the interface in semantic space is pushed closer to the domain language description. Here, things of interest are made simple because the lexicon of the interface language maps well into the lexicon of domain description. Considerable planning may still go on in the conception of the domain itself, but little or no planning or translation is required to get from the language of domain description to the language of the interface. The price paid for these advantages is a loss of generality: Many things are unnatural or even impossible.

Automated Behavior Does Not Reduce Semantic Distance

Cognitive effort is required to plan a sequence of actions to satisfy some intent. Generally, the more structure required of the user, the more effort use of the system will entail. However, this gap can be overcome if the users become familiar enough with the system. Structures that are used frequently need not be rebuilt every time they are needed if they have been remembered. Thus, a user may remember how to do something rather than having to rederive how to do it. It is well known that when tasks are practiced sufficiently often, they become automated, requiring little or no conscious attention. As a result, over time the use of an interface to solve a particular set of problems will feel less difficult and more direct. Experienced users will sometimes argue that the interface they use directly satisfies their intentions, even when less skilled users complain of the complexity of the structures. To skilled users, the interface feels direct because the invocation of mediating structure has been automated. They have learned how to transform frequently arising intentions into action specifications. The result is a feeling of directness as compelling as that which results from semantic directness. As far as such users are concerned, the intention comes to mind and the action gets executed. There are no conscious intervening stages. (For example, a user of the vi text editor expressed this as follows: "I am an expert user of vi, and when I wish to delete a word, all I do is think 'delete that word,' my fingers automatically type 'dw,' and the word disappears from the screen. How could anything be more direct?")

The frequent use of even a poorly designed interface can sometimes result in a feeling of directness like that produced by a semantically direct interface. A user can compensate for the deficiencies of the interface through continual use and practice so that the ability to use it becomes automatic, requiring little conscious activity. While automatism is one factor which can contribute to a feeling of directness, it is essential for an interface designer to distinguish it from semantic distance. Automatization does not reduce the semantic distance that must be spanned; the gulfs between a user's intentions and the interface must still be bridged by the user. Although practice and the resulting expertise can make the crossing less difficult, it does not reduce the magnitude of the gulfs. Planning activity may be replaced by a single memory retrieval so that instead of figuring out what to do, the user remembers what to do. Automatization may feel like direct control, but it comes about for completely different reasons than semantic directness. Automatization is useful, for it improves the interaction of the user with the system, but the feeling of directness it produces depends only on how much practice a particular user has with the system and thus gives the system credit for the work the user has done. Although we need to remember that this happens, that users may adjust themselves to the interface and, with sufficient practice, may view it as directly supporting their intentions, we need to distinguish between the cases in which the feeling of directness originates from a close semantic coupling between intentions and the interface language and that which originates from practice. The resultant feeling of directness might be the same in the two cases, but there are crucial differences between how the feeling is acquired and what one needs to do as an interface designer to generate it.

The User Can Adapt to the System Representation

Another way to span the gulf is for the users to change their own conceptualization of the problem so that they come to think of it in the same terms as the system. In some sense, this means that the gulf is bridged by moving the user closer to the system. Because of their experience with the system, the users change both their understanding of the task and the language with which they think about issues. This is related to the notion of linguistic determinism. If it is true that the way we think about something is shaped by the vocabulary we have for talking about it, then it is important for the designer of a system to provide the user with a good representation ofthe task domain in question. The interface language should provide a powerful, productive way of thinking about the domain.

This form of the users adapting to the system representation takes place at a more fundamental level than the other ways of reducing semantic distance. While moving the interface closer to the users' intentions may make it difficult to realize some intentions, changing the users' conception of the domain may prevent some intentions from arising at all. So while a well-designed special-purpose language may give the users a powerful way of thinking about the domain, it may also restrict the users' flexibility to think about the domain in different ways.

The assumption that a user may change conceptual structure to match the interface language follows from the notion that every interface language implies a representation of the tasks it is applied to. The representation implied by an interface is not always a coherent one. Some interfaces provide a collection of partially overlapping views of a task domain. If a user is to move toward the model implied by the interface, and thus reduce the semantic distance, that model should be coherent and consistent over some conception of the domain. There is, of course, a trade-off here between the costs to the user of learning a new way to think about a domain and the potential added power of thinking about it in the new way.

Virtuosity and Semantic Distance

Sometimes users have a conception of a task and of a system that is broader and more powerful than that provided by an interface. The structures they build to make contact with the interface go beyond it. This is how we characterize virtuoso performances in which the user may "misuse" limited interface tools to satisfy intentions that even the system designer never anticipated. In such cases of virtuosity the notion of semantic distance becomes more complicated and we need to look very carefully at the task that is being accomplished. Semantic directness always involves the relationship between the task one wishes to accomplish and the ways the interface provides for accomplishing it.

If the task changes, then the semantic directness of the interface may also change.

Consider a musical example: Take the task of producing a middle-C note on two musical instruments, a piano and a violin. For this simple task, the piano provides the more direct interface because all one need do is find the key for middle-C and depress it, whereas on the violin, one must place the bow on the G string, place a choice of fingers in precisely the right location on that string, and draw the bow. A piano's keyboard is more semantically direct than the violin's strings and bow for the simple task of producing notes. The piano has a single well-defined vocabulary item for each of the notes within its range, while the violin has an infinity of vocabulary items, many of which do not produce proper notes at all. However, when the task is playing a musical piece well rather than simply producing notes, the directness of the interfaces can change. In this case, one might complain that a piano has a very indirect interface because it is a machine with which the performer "throws hammers at strings." The performer has no direct contact with the components that actually produce the sound, and so the production of desired nuances in sound is more difficult. Here, as musical virtuosity develops, the task that is to be accomplished also changes fromjust the production of notes to concern for how to control more subtle characteristics of the sounds like vibrato, the slight changes in pitch used to add expressiveness. For this task the violin provides a semantically more direct interface than the piano. Thus, as we have argued earlier, an analysis of the nature of the task being performed is essential in determining the semantic directness of an interface.

3.4. Articulatory Distance

In addition to its meaning, every vocabulary item in every language has a physical form and that form has an internal structure. Words in natural languages, for example, have phonetic structure when spoken and typographic structure when printed. Similarly, the vocabulary items that constitute an interface language have a physical structure. Where *semantic distance* has to do with the relationship between user's intentions and meanings of expressions, *articulatory distance* has to do with the relationship between the meanings of expressions and their physical form. On the input side, the form may be a sequence of character-selecting key presses for a command language interface, the movement of a mouse and the associated "mouseclicks" in a pointingdevice interface, or a phonetic string in a speech interface. On the output side, the form might be a string of characters, a change in an iconic representation, or variation in an auditory signal.

There are ways to design languages such that the relationships between the forms of the vocabulary items and their meanings are not arbitrary. One technique is to make the physical form of the vocabulary items structurally similar to their meanings. In spoken language this relationship is called onomatopoeia. Onomatopoetic words in spoken language refer to their meanings by imitating the sound they refer to. Thus we talk about the "boom" of explosions or the "cock-a-doodle-doo" of roosters. There is an economy here in that the user's knowledge of the structure of the surface acoustical form has a nonarbitrary relation to meaning. There is a directness of reference in this imitation; an intervening level of arbitrary symbolic relations is eliminated. Other uses of language exploit this effect partially. Thus, although the word "long" is arbitrarily associated with its meaning, sentences like "She stayed a looooooooooong time" exploit a structural similarity between the surface form of "long" (whether written or spoken) and the intended meaning. The same sorts of things can be done in the design of interface languages.

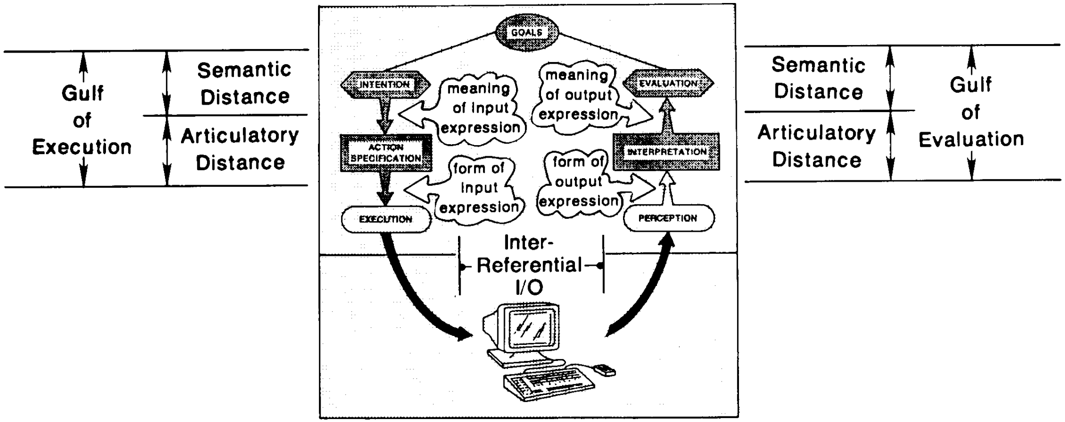
In many ways, the interface languages should have an easier time of exploiting articulatory similarity than do natural languages because of the rich technological base available to them. Thus, if the intent is to draw a diagram, the interface might accept as input drawing motions. In turn, it could present as output diagrams, graphs, and images. If one is talking about sound patterns in the input interface language, the output could be the sounds themselves. The computer has the potential to exploit articulatory similarities through technological innovation in the varieties of dimensions upon which it can operate. This potential has not been exploited, in part because of economic constraints. The restriction to simple keyboard input limits the form and structure of the input languages and the restriction to simple, alphanumeric terminals with small, low-resolution screens, limits the form and structure of the output languages.

3.5. Articulatory Distance in the Gulfs of Execution and Evaluation

The relationships among semantic distance, articulatory distance, and the gulfs of execution and evaluation are illustrated in Figure 6.

Take the simple, commonplace activity of moving a cursor on the screen. If we do this by moving a mouse, pointing with a finger or a light pen at the screen, or otherwise mimicking the desired motion, then at the level of action execution, these interactions all exhibit articulatory directness. The meaning of the intention is cursor movement and the action is specified by means of a similar movement. One way to achieve articulatory directness at the input side is to provide an interface that permits specification of an action by mimicking it, thus supporting an articulatory similarity between the vocabulary item and its meaning. Any nonarbitrary relationship between the form of an item and its meaning can be a basis for articulatory directness. While structural relationships of form to meaning may be desirable, it is sometimes necessary to re-

Figure 6. Forming an intention is the activity that spans semantic distance in the gulf of execution. The intention specifies the meaning of the input expression that is to satisfy the user's goal. Forming an action specification is the activity that spans articulatory distance in the gulf of execution. The action specification prescribes the form of an input expression having the desired meaning. The form of the input expression is executed by the user on the machine interface and the form of the output expression appears on the machine interface, to be perceived by the user. When some part of the form of a previous output expression is incorporated in the form of a new input expression, the input and output are said to be inter-referential. Interpretation is the activity that spans articulatory distance in the gulf of evaluation. Interpretation determines the meaning of the output expression from the form of the output expression. Evaluation is the activity that spans semantic distance in the gulf of evaluation. Evaluation assesses the relationship between the meaning of the output expression and the user's goal.



sort to an arbitrary relationship of form to meaning. Still, some arbitrary relationships are easier to learn than others. It may be possible to exploit previous user knowledge in creating this relationship. Much of the work on command names in command language interfaces is an instance of trying to develop memorable and discriminable relationships between the forms and the meanings ofcommand names (Black & Moran, 1982; Black & Sebrechts, 1981; Carrol, 1985).

Articulatory directness on the output side is similar. If the user is following the changes in some variable, a moving graphical display can provide articulatory directness. A table of numbers, although containing the same semantic information, does not provide articulatory directness. Thus, the graphical display and the table of numbers might be equal in semantic directness, but unequal in articulatory directness. The goal of designing for articulatory directness is to couple the perceived form of action and meaning so naturally that the relationships between intentions and actions and between actions and output seem straightforward and obvious.

In general, articulatory directness is highly dependent upon I/O technology. Increasing the articulatory directness of actions and displays requires a much richer set of input/output devices than most systems currently have. In addition to keyboards and bit-mapped screens, we see the need for various forms of pointing devices. Such pointing devices have important *spatio-mimetic* properties and thus support the articulatory directness of input for tasks that can be represented spatially. The mouse is useful for a wide variety of tasks not because of any properties inherent in itself, but because we map so many kinds of relationships (even ones that are not intrinsically spatial) on to spatial metaphors. In addition, there are often needs for sound and speech, certainly as outputs, and possibly as inputs. Precise control of timing will be necessary for those applications where the domain of interest is time sensitive. Perhaps it is stretching the imagination beyond its willing limits, but Galton (1894) suggested and carried out a set of experiments on doing arithmetic by sense of smell. Less fancifully conceived, input might be sensitive not only to touch, place, and timing, but also to pressure or to torque (see Buxton, 1986; Minsky, 1984).

4. DIRECT ENGAGEMENT

Direct engagement occurs when a user experiences direct interaction with the objects in a domain. Here there is a feeling of involvement directly with a world of objects rather than of communication with an intermediary. The interactions are much like interacting with objects in the physical world. Actions apply to the objects, observations are made directly upon those objects, and the interface and the computer become invisible. Although we believe this feeling of direct engagement to be of critical importance, in fact, we know little about the actual requirements for producing it. Laurel (1986) discusses some of the requirements. At a minimum, to allow a feeling of direct engagement the system requires the following:

> Execution and evaluation should exhibit both semantic and articulatory directness.
>
> Input and output languages of the interface should be inter-referential, allowing an input expression to incorporate or make use of a previous output expression. This is crucial for creating the illusion that one is directly manipulating the objects of concern.
>
> The system should be responsive, with no delays between execution and the results, except where those delays are appropriate for the knowledge domain itself.
>
> The interface should be unobtrusive, not interfering or intruding. If the interface itself is noticed, then it stands in a third-person relationship to the objects of interest, and detracts from the directness of the engagement.

In order to have a feeling of direct engagement, the interface must provide the user with a world in which to interact. The objects of that world must feel like they are the objects of interest, that one is doing things with them and watching how they react. In order for this to be the case, the output language must present representations of objects in forms that behave in the way that the user thinks of the objects behaving. Whatever changes are caused in the objects by the set of operations must be depicted in the representation of the objects. This use of the same object as both an input and output entity is essential to providing objects that behave as if they are the real thing. It is because an input expression can contain a previous output expression that the user feels the output expression is the thing itself and that the operation is applied directly to the thing itself.

In addition, all of the discussions of semantic and articulatory directness apply here too, because the designer of the interface must be concerned with what is to be done and how one articulates that in the languages of interaction. But the designer must also be concerned with creating and supporting an illusion. The specification of what needs to be done and evidence that it has been done must not violate the illusion, else the feeling of direct engagement will be lost.

One factor that seems especially relevant to maintaining this illusion is the form and speed of feedback. Rapid feedback in terms of changes in the behavior of objects not only allows for the modification of actions even as they are being executed, but also supports the feeling of acting directly on the objects themselves. It removes the perception of the computer as an intermediary by providing continual representation of system state. In addition, rapidity of feedback and continual representation of state allows one to make use of perceptual faculties in evaluating the outcome of actions. We can watch the actions take place, monitoring them much like we monitor our interactions with the physical world. The reduction in the cognitive load of mentally maintaining relevant information and the form of the interaction contribute to the feeling of engagement.

5. A SPACE OF INTERFACES

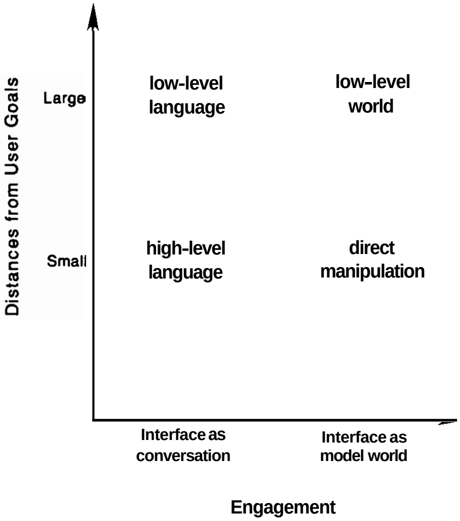
Distance and engagement are depicted in Figure 7 as two major dimensions in a space of interface designs. The dimension of engagement has two landmark values: One is the metaphor of interface as conversation; the other is the metaphor of interface as model world. The dimension of distance actually contains two distances to be spanned: semantic and articulatory distances, the two kinds of gulfs that lie between the user's conception of the task and the interface language.

The least direct interface is often one that provides a low-level language interface, for this is apt to provide the weakest semantic match between intentions and the language of the interface. In this case, the interface is an intermediary between the user and the task. Even worse, it is an intermediary that does not understand actions at the level of description in which the user likes to think of them. Here the user must translate intentions into complex or lengthy expressions in the language that the interface intermediary can understand.

A more direct situation arises when the central metaphor of the interface is a world. Then the user can be directly engaged with the objects in a world; but still, if the actions in that world do not match those that the user wishes to perform within the task domain, getting the task done may be a difficult process. The user may believe that things are getting done and may even experience a sense of engagement with the world, yet still be doing things at too low a level. This is the state of some of the recently introduced direct manipulation systems: They produce an immediate sense of engagement, but as the user develops experience with the system, the interface appears clumsy, to interfere too much, and to demand too many actions and decisions at the wrong level of specification. These interfaces appear on the surface to be direct manipulation interfaces, but they fail to produce the proper feelings of direct engagement with the task world.

Closing the distance between the user's intentions and the level of specification of the interface language allows the user to make efficient specifications of intentions. Where this is done with a high-level language, quite efficient interfaces can be designed. This is the situation in most modern integrated pro-

Figure 7. A space of interfaces. The dimensions of distance from user goals and degree of engagement form a space of interfaces within which we can locate some familiar types of interfaces. Direct manipulation interfaces are those that minimize the distances and maximize engagement. As always, the distance between user intentions and the interface language depends on the nature of the task the user is performing.



gramming environments. For some classes of tasks, such interfaces may be superior to direct manipulation interfaces.

Finally, the most direct of the interfaces will lie where engagement is maximized, where just the right semantic and articulatory matches are provided, and where all distances are minimized.

6. PROBLEMS WITH DIRECT MANIPULATION

Direct manipulation systems have both virtues and vices. For instance, the immediacy of feedback and the natural translation of intentions to actions make some tasks easy. The matching of levels of thought to the interface language — semantic directness — increases the ease and power of performing some activities at a potential cost of generality and flexibility. But not all things should be done directly. For example, a repetitive operation is probably best done via a script, that is, through a symbolic description of the tasks that are to be accomplished. Direct manipulation interfaces have difficulty handling variables, or distinguishing the depiction of an individual element from a representation of a set or class of elements. Direct manipulation interfaces have problems with accuracy, for the notion of mimetic action puts the responsibility on the user to control actions with precision, a responsibility that is sometimes best handled through the intelligence of the system and sometimes best communicated symbolically.

A more fundamental problem with direct manipulation interfaces arises from the fact that much of the appeal and power of this form of interface comes from its ability to directly support the way we normally think about a domain. A direct manipulation interface amplifies our knowledge of the domain and allows us to think in the familiar terms of the application domain rather than in those of the medium of computation. But if we restrict ourselves to only building an interface that allows us to do things we can already do and to think in ways we already think, we will miss the most exciting potential of new technol*ogy:* to provide new ways to think of and to interact with a domain. Providing these new ways and creating conditions that will make them feel direct and natural is an important challenge to the interface designer.

Direct manipulation interfaces are not a panacea. Although with sufficient practice by the user many interfaces can come to feel direct, a properly designed interface, one which exploits semantic and articulatory directness, should decrease the amount of learning required and provide a natural mapping to the task. But interface design is subject to many tradeoffs. There are surely instances when one might wisely trade off directness for generality, or for more facile ways of saying abstract things. The articulatory directness involved in pointing at objects might need to be traded off against the difficulties of moving the hands between input devices or of problems in pointing with great precision.

It is important not to equate directness with ease of use. Indeed, if the interface is really invisible, then the difficulties within the task domain get transferred directly into difficulties for the user. Suppose the user struggles to formulate an intention because of lack of knowledge of the task domain. The user may complain that the system is difficult to use. But the difficulty is in the task domain, not in the interface language. Direct manipulation interfaces do not pretend to assist in overcoming problems that result from poor understanding of the task domain.

What about the claims for direct manipulation? We believe that direct manipulation systems carry gains in ease of learning and ease of use. If the mapping is done correctly, then both the form and the meaning of commands should be easier to acquire and retain. Interpretation of the output should be immediate and straightforward. If the interface is a model of the task domain, then one could have the feeling of directly engaging the problem of interest itself. It is sometimes said that in such situations the interface disappears. It is probably more revealing to say that the interface is no longer recognized as an interface.

But are these desirable features? Are the trade-offs too costly? As always, we are sure that the answer will depend on the tasks to be accomplished. Certain kinds of abstraction that are easy to deal with in language seem difficult in a concrete model of a task domain. When we give up the conversation metaphor, we also give up dealing in descriptions, and in some contexts, there is great power in descriptions. As an interface to a programming task, direct manipulation interfaces are problematic. We know of no really useful direct manipulation programming environments. Issues such as controlling the scope of variable bindings promise to be quite tricky in the direct manipulation environments. Will direct manipulation systems live up to their promise? Yes and no. Basically, the systems will be good and powerful for some purposes, poor and weak for others. In the end, many things done today will be replaced by direct manipulation systems. But we will still have conventional programming languages.

On the surface, the fundamental idea of a direct manipulation interface to a task flies in the face of two thousand years of development of abstract formalisms as a means of understanding and controlling the world. Until very recently, the use of computers has been an activity squarely in that tradition. So the exterior ofdirect manipulation, providing as it does for the direct control of a specific task world, seems somehow atavistic, a return to concrete thinking. On the inside, of course, the implementation ofdirect manipulation systems is yet another step in that long, formal tradition. The illusion of the absolutely manipulable concrete world is made possible by the technology of abstraction.

Acknowledgments. We thank Ben Shneiderman for his helpful comments on an earlier draft of the chapter, Eileen Conway for her aid with the illustrations, and Julie Norman and Sondra Buffett for extensive editorial comments.

Support. The research reported here was conducted under Contract N00014-85-C-0133, NR 667-541 with the Personnel and Training Research Programs of the Office of Naval Research and with the support of the Navy Personnel Research and Development Center. The views and conclusions contained in this document are those ofthe authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the sponsoring agency.

REFERENCES

Black, J. B., & Moran, T. P. (1982). Learning and remembering command names. *Proceedings ofthe Human Factors in Computer Systems Conference,* 8–11. New York: ACM.

Black, J. B., & Sebrechts, M. M. (1981). Facilitating human-computer communication. *Applied Psycholinguistics,* 2, 149–177.

Borning, A. (1979). *ThingLab: A constraint-oriented simulation laboratory* (Tech. Rep. No. SSL-79-3). Palo Alto, CA: Xerox Palo Alto Research Center.

Budge, B. (1983). *Pinball construction set* [Computer program]. San Mateo, CA: Electronic Arts.

Buxton, W. (1986). There'smore to interaction than meets the eye: Some issues in manual input. In D. A. Norman & S. W. Draper (Eds.), *Usercenteredsystemdesign; New perspectives on human-computer interaction.* Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Carrol, J. M. (1985). *What's in a name? A n essay in the psychology of reference.* New **York:** Freeman.

diSessa, A. A. (1985). A principles design for an integrated computational environment. *Human-Computer Interaction, I,* 1–47.

Draper, **S.** W. (1986). Display managers as the basis for user-machine communication. In D. A. Norman & **S.** W. Draper (Eds.), ***User** centeredsystem design: New perspectives on human-computer interaction.* Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Galton, F. (1894). Arithmetic by smell. *Psychological Review, I,* 61–62.

Hollan, J. D., Hutchins, E., & Weitzman, L. (1984). Steamer: An interactive inspectable simulation-based training system. *AI Magazine,* **5,** 15–27.

Hollan, J. D., Stevens, A., & Williams, M. D. (1980). Steamer: An advanced computer-assisted instruction system for propulsion engineering. *Proceedings @Summer Computer Simulation Conference,* 400–404. Arlington, VA: AFIPS Press.

Kay, A. (1984, September). Computer software. *Scientific American,* 52–59.

Laurel, B. K. (1986). Interface as mimesis. In D. A. Norman & **S.** W. Draper (Eds.), ***User** centered system design: **New** perspectives on human-computer interaction.* Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Minksy, M. R. (1984,July). Manipulating simulated objects with real-world gestures using a force and position sensitive screen. *Computer Graphics,* 195-203.

Norman, D. A., &Draper, S. W. (Eds.). (1986). *User centered system design: New perspectives on human-computer interaction.* Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Perlis, A. J. (1982). Epigrams on programming. *SIGPLAN Notices, 17*(9), 7-13.

Shneiderman, B. (1974). A computer graphics system for polynomials. *The Mathematics Teacher, 67*(2), 111-113.

Shneiderman, B. (1982). The future of interactive systems and the emergence ofdirect manipulation. *Behavior and Information Technology, I,* 237-256.

Shneiderman, B. (1983). Direct manipulation: A step beyond programming languages. *IEEE Computer, 16*(8), 57–69.

Sutherland, I. E. (1963). Sketchpad: A man-machine graphical communication system. *Proceedings of the SpringJoint Computer Conference,* 329-346. Baltimore, MD: Spartan Books.

HCI Editorial Record. This is an invited paper based on a draft of April 1, 1985. Final manuscript received October 3, 1985. – *Editor*

APPENDIX F

Usability Engineering

Jakob Nielsen

This book is printed on acid-free paper. ♾

ACADEMIC PRESS
An Imprint of Elsevier
525 B Street, Suite 1900, San Diego, CA 92101-4495 USA
http://www.academicpress.com

Academic Press
Harcourt Place, 32 Jamestown Road, London, NW1 7BY UK

Morgan Kaufmann
340 Pine Street, Sixth Floor, San Francisco, CA 94104-3205
http://www.mkp.com

Library of Congress Catalog Number: 93000488
ISBN-13: 978-0-12-518406-9
ISBN-10: 0-12-518406-9

Printed in the United States of America
08 / 17 16

based on similarity ratings for the various functions from experienced users.

In recent years, many telephone-operated interfaces have been designed to allow users to access various forms of information and services, such as their bank account balance, over regular push-button telephones [Halstead-Nussloch 1989]. These systems are very often menu-based, but are otherwise closer to the line-oriented generation of user interfaces since the dialogue is completely linear. Telephone-operated interfaces are thus an example of the hybrid nature of menu interfaces and also indicate that the concept of "generations" of interfaces presented in this chapter should be seen more as a way to conceptualize the history of user interface design than as a sequential progression of interfaces replacing each other.

3.4 *Graphical User Interfaces*

Even though graphical user interfaces have a history going back to Ivan Sutherland's Sketchpad system from 1962 [Sutherland 1963], Douglas Engelbart's mouse from 1964 [Engelbart 1988], and several research systems from the 1970s [Goldberg 1988], they did not see widespread commercial use until the 1980s [Perry and Voelcker 1989]. Most current user interfaces belong to the category of graphical user interfaces sometimes referred to as WIMP systems (windows, icons, menus, and a pointing device) after their basic components. As can be seen in Figure 4, window interfaces almost add a third dimension to the two dimensions inherent in each window because of the possibility for overlapping windows. Of course, overlapping windows are not truly three-dimensional since it is not possible to see the content of obscured windows without moving them to the top, so it would be more accurate to refer to these interfaces as having two-and-a-half dimensions.

The primary interaction style used in many graphical user interfaces is direct manipulation [Shneiderman 1983], which is based on visual representation of the dialogue objects of interest to the user. Such a continuously updated representation allows the user to control the dialogue by moving objects around on the screen and otherwise manipulating them with the mouse. As an example, the traditional way of specifying a margin indentation in a word processor would be to issue a command to indent by a certain number of spaces. Such a command is an indirect manipulation of the margin, however, and the user may have to try several times before the desired layout is achieved. In contrast, direct manipulation of a margin would involve dragging the margin itself or a margin marker to the desired position. Since the user is getting continuous feedback about the positioning of the margin as it is being moved, the result should be less of a surprise. Of course, this example does show that direct manipulation may not be optimal for all tasks, in that it would be easier to achieve a very precise margin setting by typing in the number.

Let's move from the interaction techniques to the structure of the interface. Many graphical user interfaces can be said to be *object-oriented*.[1] Object-oriented interfaces are in contrast to the function-oriented interfaces that were the traditional structure for character-based interfaces. In a function-oriented interface, the interaction is structured around a set of commands issued by the user in various combinations to achieve the desired result. The main interface issue is how to provide easy access to these commands and their parameters, and typical solutions include command-line interfaces with various abbreviation options as well as full-screen menus.

Object-oriented interfaces are sometimes described as turning the application inside–out as compared to function-oriented interfaces. The main focus of the interaction changes to become the users' data and other information objects that are typically represented graphically on the screen as icons or in windows. Users achieve their goals by gradually massaging these objects (using various modification features that are of course similar to the concept of

[1] Note that I am talking about object-oriented *interfaces*. These interfaces may or may not be implemented using object-oriented *programming* which is a completely different issue. Once the interface has been structured around objects, it may feel natural to implemented these objects using object-oriented programming, but one can also implement object-oriented interfaces using traditional programming methods.

commands) un
desired result.
who are used t
serious difficul
interfaces [Niel

An example m
object-oriented
user interface i
certain informa
printing the resu
designed by par
specify the quer
had to select f
menu and, finall
Only after the la
from the databa
tions to be perfo
be manipulated
would start by
from the databas
for the user to r
would simplify
the user modifie
the content of th
fying the query
window layout,
typical records v
print command v
not be a surprise
centered modifi
already been obs

Most interface s
have better usabi
interfaces, especi
In spite of this co
evidence to prove
few experiments

APPENDIX G

For the system designer the choices are bewildering: How does one put together, from a multitude of techniques and devices, the combination best suited to meet the needs of a human being who must perform real work?

The Human Factors of Computer Graphics Interaction Techniques

James D. Foley
The George Washington University
Victor L. Wallace
The University of Kansas
Peggy Chan
The George Washington University

The promise of interactive graphics is to provide a user-computer communications medium that is at once benign, responsive, and graphic. We expect it to be benign and responsive in the same sense a trusted servant is expected to be. We expect it to be graphic with the clarity and richness of communication that only graphic communication presents. When a person uses an interactive graphics system to do real work, he wants the system to virtually disappear from his consciousness so that only his work and its ramifications have a claim on his energy.

This promise is frequently not fulfilled. Designers of graphics systems—software and hardware—often lack the intuition, knowledge, and experience necessary to "engineer" the forms of dialogue between user and computer to best advantage. The fault lies partly in a lack of current literature, partly in a lack of educational opportunities, and partly in our lack of knowledge of methods and information structures by which such designers can be successful. Well-designed systems are usually the result of a diligent, essentially creative, enterprise.

We believe that one of the most important elements in the design of interactive user-computer interfaces is the selection of the devices and techniques by which the user performs elementary tasks. The purpose of our work is to provide a systematic structure to aid the designer in making this selection.

Looking for guidance

There is a multitude of interaction techniques. Each has a specific purpose, such as to specify a command,

0272-1716/84/1100-0013$01.00 © 1984 IEEE

designate a position, or select a displayed object, and each is implemented with some device, such as a tablet, joystick, keyboard, light pen, trackball, or potentiometer. Typical techniques include selecting a command from a menu with a pointing device, specifying a position with a tablet or a joystick and receiving cursor feedback on the screen, typing a numeric value on a keyboard, or designating a displayed object with a pointing device.

We all recognize, from our own experiences with interactive computing (which need not have been with interactive *graphics*), the costs of poorly designed interfaces. Coming in many forms, the costs can include degraded user productivity, user frustration, increased training costs, and the need to redesign and reimplement the user interface. Specific experiments confirm that the costs are real. How can we avoid these costs? Where can we turn for guidance? There are three basic sources of information:

(1) experience-based guidelines,
(2) experiments with interaction techniques, and
(3) the human factors literature, especially that dealing with equipment design.

Over the past 10 years we have heard much lore about what makes interactive graphics systems easy (or hard) to use, and about the pros and cons of various interaction devices and techniques. Much of this lore has not found its way into the literature. Furthermore, any lore that can be found in the literature is typically scattered amidst application descriptions. Only a few writers (Bennett,[1] Britton,[2] Cheriton,[3] Engel,[4] Foley,[5] Hansen,[6] Smith,[7] and Wallace[8]) have attempted to summarize in a structured way either their design philosophy or their accumulated knowledge and experience. These papers represent one source of guidance. More recently, the annual SIGCHI (ACM's Special Interest Group for Computer-Human Interaction) and Human Factors Society conference proceedings have begun to serve as focal points.

Starting in the late 1960's, researchers have carried out a modest but rapidly expanding collection of experiments, such as comparisons of different interaction techniques. Some of the experiments were performed by computer scientists, others by human factors specialists, and still others collaboratively by multidisciplinary teams. The results are often useful, but generalizing beyond the specific circumstances of the experiment is difficult. Ramsey and Atwood,[9] as part of a larger effort, have published a 10-page review and discussion of the results of most of these experiments. We have summarized and critiqued the earlier experiments (Card,[10] Earl,[11] English,[12] Fields,[13] Haller,[14] Goodwin,[15] Irving,[16] Mehr,[17] and Morrill[18]) in a report[19] on which this article is based.

The most promising source of guidance is human factors literature. Yet, there is no single, coherent human factors literature as such because of the variety of problem areas and research disciplines involved in human factors work. Thus we find a diverse mixture of ad hoc experiments, evaluations, and lore, as well as important treatises on methodology, all packaged into a handful of books and professional journals and a large number of hard-to-find technical reports. In general, the human factors literature concentrates on human capabilities and limitations, clustered under the following topics: information presentation (visual and auditory), human control of systems, man as a system component (e.g., "man-in-the-loop" models), workspace design, and methods for observing, analyzing, and measuring human performance.

The field of human factors engineering (or ergonomics) is better defined in terms of its objectives than its researcher and practitioner constituencies, since it is multidisciplinary as well as interdisciplinary in makeup, with perhaps its heaviest concentrations in the behavioral and biological sciences. At one extreme it fades into environmental psychology, and at the other it merges with physiology. The common objective of all human factors work is simply stated: to achieve, through appropriate design, functional effectiveness of whatever physical equipment or facilities people use.

The most widely used textbook in this extremely diverse field, E. J. McCormick's,[20] provides an excellent general survey but does not illustrate how human factors can influence the design process. Also, aside from some useful tables on control devices, published by Chapanis,[21] little practical guidance is available to computer designers and virtually none to the designers of interactive graphics devices or systems.

Poorly designed interfaces can include degraded user productivity, user frustration, increased training costs, and the need to redesign and reimplement.

The two "classics" on equipment design—one by Woodson and Conover,[22] the other by Van Cott and Kinkade[23]—are of greater practical value, the former as a repository of accumulated designer experience and the latter as a compendium of studies, surveys, and experiments. Designers might also consult another work by Chapanis[24] and a paper by Fitts,[25] but these also predate the advent of interactive computer systems and are concerned mostly with the human interface to machines used to control other physical machinery or process variables. Thus all these works provide only a general backdrop of ideas for designers of complex, interactive, information-processing equipment. More recent work— Sheridan and Ferrell's,[26] for instance—has dealt with the operation of computer-controlled equipment, but this has been more concerned (justifiably) with the development of an adequate theory than with heuristics for designers.

Human factors researchers have always faced severe methodological challenges, both in collecting trustworthy data from observations of people actually using machines, and in conducting human experimentation that can be reliably applied to the design of man-machine systems. Chapanis[27] remains the only source of guidance on these problems, which are certainly no less important in the case of information-processing machines than, say, of cockpits or control towers. But the methods of cognitive ergonomics, so critical to interactive information systems, are not discussed at all in Chapanis' work, which, after all, predates the field of cognitive psychology.

Much attention has been devoted to the chronic problem of integrating human factors awareness into the early stages of the design process, where it can have the greatest benefit—see books by Meister,[28,29] for example. Although Meister's and other recent work has focused on the use of behavioral data as a foundation for system design, it has concentrated on very large scale, multiperson (crew) systems and is difficult to generalize to other types of systems.

Recently, interest in white-collar ergonomics, particularly in Europe, has generated a lot of work on the human factors relevant to the design of alphanumeric video display terminals and their various workplace environments. Cakir, Hart, and Stewart[30] summarize this work and offer recommendations for the elimination of such operator ailments as eyestrain and backache; much of this material can certainly be applied to graphics systems, whose operators often spend hours leaning uncomfortably forward in their seats to work at their consoles. However, this very good, but traditional, human engineering study says nothing about how to design interactive techniques that foster a more intimate coupling of man and machine. Like the bulk of the human factors literature, it is aimed at eliminating hazards and alleviating hindrances rather than at directing system design in a positive sense.

Additional difficulties with all these different sources of guidance are that they are hard to locate, are usually couched in disciplinary jargon, and use little consistent terminology. Consequently, the designer of an interactive system must rely primarily on personal experiences and on those of colleagues, despite the existence of many potentially useful materials. Instead of standing on their professional forebears' shoulders, designers seem destined to stand only on their forebears' toes.

A notable exception is the work of Stu Card, Tom Moran, and Alan Newell.[31,32] Their "Keystroke-Level Model" and "GOMS (goals, operators, methods, and selection rules) Model" integrate classical time and motion study concepts with certain aspects of cognitive psychology, providing useful engineering models of user performance. While the models do not address important areas such as error rates and learning, they represent a tremendously important step forward.

Our intent in this article is to integrate within a unified and logical structure a significant and useful body of the experiential and experimental conclusions drawn from all these sources.

Scope of the problem

The designer of an interactive graphics system must define everything about the user-computer interface, ranging from the concepts the user must understand, down to the finer details of screen formats, interaction techniques, and device characteristics. Here, a brief description of the overall design process will show how the issue of interaction technique fits into the whole.

A number of writers (Britton,[2] Wallace,[8] Foley et al.,[19] Newman and Sproull,[33] Moran,[34] and Dunn,[35]) have suggested a top-down design approach. The first step in the process is to understand the application area and prospective users. This understanding can be gained partly by studying the way the application is currently treated. As Hornbuckle says, "Observing what man does normally during his creative efforts can provide a starting point for the . . . designer. In particular, a mathematician does not manipulate equations at a typewriter, nor does a circuit designer prefer a keypunch."[36] Hansen is even more succinct; his advice is "Know the user"—watch him, study him, interact with him, learn to understand how he thinks and why he does what he does.[6]

This process is often called "requirements definition" or "task analysis." It results in a set of functional requirements, or capabilities, to be made available through the user-computer interface. The process also provides insight into how the capabilities of the system can best be presented to the user. Finally, the analysis identifies the type of user for whom the system is to be designed.

The requirements definition eventually is used as the basis for defining the languages of interaction between computer and user. We view the user-computer interface as composed of two languages: with one the user communicates to the computer; with the other the computer communicates to the user. Because our focus here is on interaction techniques for input (user-to-computer communication), we will discuss only the input language. Defining an input language is a top-down process, starting with the user's conceptual model, then the command structure, the syntax, and finally the assignment of physical devices and activities—the only stage in which interaction techniques are directly involved.

As a first step, we define the fundamental *conceptual model* with which the user must deal. Sometimes called the "high-level semantics" or the user's "mental model," the conceptual model embodies the key developmental components of the detailed semantics—i.e., commands. If we were defining a text editor, for example, possible conceptual models would be the line-number-oriented editor or the screen editor.

The second step, defining the detailed *semantics* of a language, follows from the conceptual model and the functional requirements. Semantics is the set of meanings conveyed by the language, including the modifiers (adjectives, adverbs, prepositions) of the language. The commands are the semantics of the input language, while the collection of information available for display to the user represents the semantics of the output language.

The language *syntax,* formed in the third step, defines how the units (words) that convey semantics are assembled into a complete sentence that instructs the computer to perform a certain task. Even for simple operations, considerable syntactic variety is possible. Consider, for example, a "move entity" command for a drafting program. Six possible syntaxes are

```
   <move command> ::=
<move>        <entity>      <position>
<move>        <position>    <entity>
<entity>      <move>        <position>
<entity>      <position>    <move>
<position>    <move>        <entity>
<position>    <entity>      <move>
```

Each of these three tokens, <move>, <entity>, and <position>, is a primitive nonterminal symbol in the syntax of the input language. By "primitive nonterminal," we mean a symbol which would be replaced by one or more terminal symbols were an additional production rule applied. In addition, primitive nonterminals have semantic meanings just as individual words in any language have meaning.

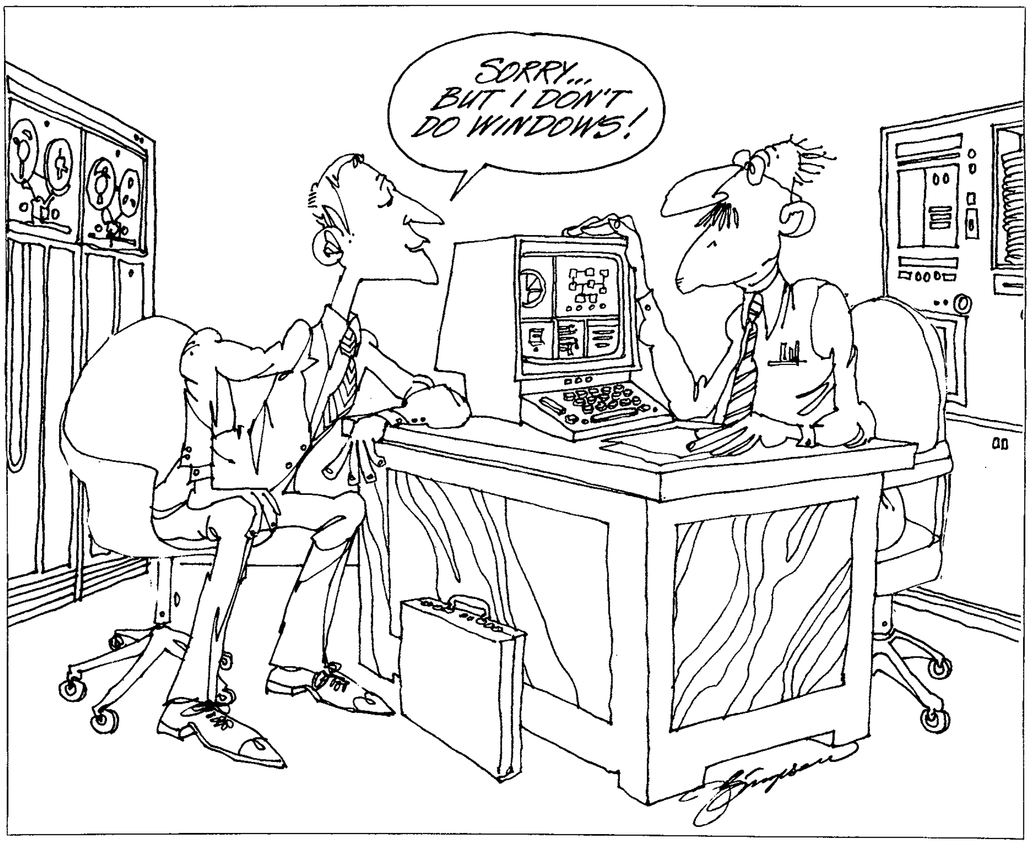
At the fourth step, the *lexical* design, these primitive nonterminals are bound to hardware devices. This is exactly where the interaction techniques come in! A technique is a binding of one or more hardware devices to primitive nonterminal symbols in the command language syntax. We call this lexical-level design because of its correspondence to the binding of syntactic tokens to letters in the input alphabet. Thus the lexical-level design requires selection of hardware devices and of the interaction techniques by which the devices will be used. The distinction between the syntactic and lexical designs is that the syntactic design stops with the primitive nonterminals, at the point where further decomposition would result in binding to devices or would cause the nonterminals to lose their meanings. That is, the syntactic design is device independent, while the lexical design is device dependent. In the example of the command sequence given above, the token "move" might be bound to hardware devices such as a keyboard, a light pen (with menu selection), or a speech recognizer.

In summary, we have a four-step, top-down design process for the user interface; the steps are conceptual design, semantic design, syntactic design, and lexical design. Our focus is on the lexical design, in which specific interaction techniques and devices are used.

Because primitive techniques will be strung together syntactically into sentences, and sentences will be combined into larger structures governed by the underlying semantics of the application, it is impossible to ignore the effect of context upon the selection of a technique. Surely, when deciding whether to carry out a positioning task with a light pen or a mouse, it makes a difference whether the task immediately preceding or the task immediately following involves a light pen. The user might well be more productive if there is a continuity of devices across the sequence of tasks—i.e., across the sentence.

The scope of our work does not extend to the physical design of interaction devices. Issues such as key shape, keyboard slant, and light pen diameter are beyond our scope and are treated extensively in the literature of traditional human factors. The basic guideline of our work is that we consider only those device characteristics normally under computer control, but not those normally built into the device hardware. We take the necessary liberty of assuming that whatever devices we might select are optimally designed for their intended uses.

Interaction tasks. Most commands to an interactive system have several primitive nonterminal symbols. The "move entity" command mentioned above has three such symbols: a position, an entity, and the imperative,



"move." The entry of each symbol by the user is an *interaction task,* performed by means of an *interaction technique.* Each task can be implemented by many different techniques. The designers of the system must select interaction techniques that best match both the user's characteristics and the specific requirements of the task, and they must also select the appropriate device. In some cases the devices are predetermined, having been selected by the hardware procurers rather than by the user interface designers. This unfortunate situation reduces the number of options for the designers and may result in a suboptimal design.

Each task has certain *requirements* dictated by the application and/or user, and each technique has certain *properties.* For example, a requirement of a positioning task may be that positions be indicated in three dimensions, while a property of a positioning technique may be that it works only in two dimensions. The 2-D techniques therefore would not be considered.

Psychological and physiological foundations. Interacting with a computer, like all human behavior, involves three types of basic human processes: perception, cognition, and motor activity. The system designer's job is to design interaction techniques which minimize the work required by these processes, both individually and in combination. To achieve this, the designer needs some key concepts from psychology, physiology, and human factors studies. Human factors bears most on the perceptual and motor processes, concerned as it is with the application of psychological and physiological knowledge to human performance, much as engineering is concerned with the application of physics and mathematics to mechanical and electrical performance. The field of cognitive psychology provides insight into our memory and learning processes.

It is beyond our scope to describe the relevant theories and experimental results from each of these areas. The work of Card, Moran, and Newell[32] integrates many of the theories, making them more usable. We refer the interested reader to the previously mentioned human factors texts; to cognitive psychology studies by Lindsay and Norman,[37] Lachman et al.,[38] Reynolds and Flagg,[39] Neisser,[40] Underwood,[41] and Nilsson;[42] and to Kolers et al.[43] on visual perception. A work by Michon et al.[44] may also be of interest. In the following subsections we point out some pertinent considerations from these fields.

The perceptual process. Perception is the process whereby unintelligible physical stimuli (generated in this case by the computer) are received by the receptor organs, transmitted to the brain, and are there recognized by a process theorized to be akin to pattern recognition. The dominant stimuli in computer use are visual, although audio stimuli are usually present to some degree (keyboard clicks, disk access arms moving, etc.) and are now commonly used (e.g., tones to catch the user's attention, speech output, etc.) as an adjunct to or replacement of visual stimuli. Tactile stimuli are also present as the user grasps for interaction devices.

Most interaction techniques start with visual perception: the user locates a menu selection, an entity to be deleted, or the cursor and recognizes a form or shape. Thus an important consideration is how to display information so that the user can quickly locate the items he needs. This means using methods such as color coding, spatial coding, blinking, brightening, movement, and reverse video to call attention to specific parts of the display.

The system designer must design interaction techniques that minimize the work required by three types of basic human processes: perception, cognition, and motor activity.

Of course, if a task might involve any of the entities being displayed, there is no point in highlighting th.m all. Often, however, particular subsets of displayed information are most germane to the application at specific points in time.

Issues of display brightness (discussed by Gould[45]), flicker (Grimes[46]), line thickness, and character fonts and sizes (McCormick[20]) are also relevant here. For instance, some fonts are more easily read than others. Spacing is important too, since menu items crammed together are much more difficult to perceive and to select than items separated by space.

The cognitive process. Cognitive psychology deals with how we acquire, organize, and retrieve information. This is what new users of an interactive system must do, not only in learning to use the system, but in regular, ongoing, productive use as well. Cognitive psychology provides a framework for studying and simplifying the information structures that new users must develop.

The theories of cognitive psychology are often expressed in terminology familiar to computer scientists. A brief, very readable article by Tracz pursues the analogy.[47] Lindsay and Norman[37] and Loftus and Loftus[48] give overviews of the area. Of course, cognitive psychology applies to user-computer interfaces in areas other than interaction devices and techniques.

Human factors engineering traditionally is not concerned with cognition but concentrates instead on designing equipment for efficiency of manipulation approaching physiological limits. Tasks involving interactive computers, however, almost by definition, have nontrivial cognitive components even at this lowest, the lexical, level. Designers of interaction techniques, therefore, must understand the process of cognition because it is sometimes the rate-determining step of a technique.

The study of cognition provides insights into ways to structure hierarchical menus, the number of choices to present, the types of words to use, and ways to name or abbreviate commands. When we learn how to use an interaction technique, we acquire and organize information concerning its use. If the information fits into categories or concepts we already understand, then the learning can proceed rapidly; if not, learning is slower. Menu symbols or names already known to the user are easier to deal with than unknown ones; a menu of 20 choices is easier to comprehend if the choices are grouped in several logically related subsets.

The motor process. The motor process comes into play when the user, having received, recognized, and decided how to respond to the stimuli, performs a response in physical actions. This may involve picking up a stylus, moving it to the tablet, and then causing the cursor to move to a particular point on the screen.

The process almost always depends on continuous perception and cognition to close the feedback loop. Perception informs the user of the locations of the tablet and stylus, and of the cursor and target, respectively, and cognition continually decides whether or not these locations have converged. In another technique—typing a command in response to a prompt—a touch typist would not depend on visual perception of the keys for feedback but instead would rely on kinesthetic, tactile, and auditory perception. He would require negligible cognition to know whether he had hit the right key. (A hunt-and-peck typist, on the other hand, would behave quite differently.)

The design and selection of interaction techniques for a task must take into account the perceptual, cognitive, and motor processes involved, even when these seem to be trivial. In general, the design goal is to minimize the time taken by each process, although this cannot always be measured in the case of cognition. The designer must also consider learning time, especially for infrequent users of a system. Identification of the processes is the important first step in analyzing and designing a technique.

Measures of ergonomic quality

An effective interaction design is one in which a user carries out his work with minimal conscious attention to his tools (the paraphernalia of the interactive terminal and the command language) and maximal effectiveness. It is free of distractions and reasonably "friendly." In our 1974 paper[5] we characterized the ideal design as one that places and structures the communication between man and computer according to the model of cooperative human-to-human conversation. We also said that it minimizes certain psychological blocks—particularly boredom, panic, frustration, confusion, and discomfort.

Many factors are involved in achieving this goal. Here, we are concentrating on approaches at the lexical level of the interaction design process: selection of appropriate interaction techniques for each elementary task a user must accomplish. The technique selected may involve the unadorned use of a physical input device, but more often a technique requires modification, through software, of the device characteristics, to make the process more natural, more interactive, easier, and more satisfying. This section examines the criteria by which techniques can be compared and a selection rationalized.

Of course, satisfaction of these primary criteria is not accomplished at the lexical level alone. The context of the task (and hence of the technique by which it is accomplished) is also significant. And of particular significance are the techniques that have normally been used for that task.

Primary criteria. In our view, the quality of an interaction design is determined by some combination of the following primary criteria:

(1) the *time* any user must spend accomplishing a particular project which the system is intended to support,

(2) the *accuracy* with which the user can accomplish the project, and

(3) the *pleasure* the user derives from the process.

In a pure-production, low-creativity environment, the third criterion may not be judged important, while in a creative, voluntary environment, maximization of that factor may be the dominant goal. Normally a design decision will be based on a combination of the above criteria, with the greatest weight applied to project time.

The relationship between a task and its most advantageous technique is also influenced strongly by user experience and knowledge. A knowledgeable user requires a wider range of facilities and finer, more precise tools than a less knowledgeable user, before he will regard the system as efficient, accurate, or pleasurable. The experienced user can tolerate a much higher apparent "memory load" with many fewer prompting features. Indeed, evidence indicates that a system design that works well for the inexperienced user can be unproductively slow, crude, and displeasing to the experienced user.

Finally, the relationship between a task and its most appropriate technique is influenced by the characteristics of the physical devices that implement the task. A three-dimensional positioning device that achieves its full range by large linear motions will be more satisfactory for implementing some techniques than one whose motion is in a small physical range and involves some rotary action in its natural motion, and vice versa. This is true even if both devices have the same resolving power, measured as a fraction of full range. The physical devices available must be considered before selecting the techniques which depend on them. This relates closely to one of Buxton's "pragmatics."[49]

In summary, then, the primary performance criteria will be met in different degrees by the same technique, depending on

(1) the context of the task among temporally adjacent tasks and the existence of global patterns of task sequencing,

(2) the experience and knowledge of the intended user, and

(3) the physical characteristics of the devices available for implementing the technique.

There is no one ideal technique for all instances of a given task.

Secondary criteria. The three primary criteria (speed, accuracy, and pleasurability) are influenced by a number of secondary criteria, which are more easily measured and used to predict performance than the primary criteria. The secondary measures of ergonomic quality that appear to be most influential are

- learning time,
- recall time,

- short-term memory load,
- long-term memory load,
- error susceptibility,
- fatigue susceptibility,
- naturalness, and
- boundedness.

Learning and recall. Perceptual learning time is the time it takes for a user to learn the patterns to be used as signatures for the elementary figures and sounds that make up the technique. This learning has already taken place in childhood for many of the common visual signatures, such as alphabetic characters from pen strokes and depth from a perspective drawing. Cognitive learning time is the time it takes to learn to use the technique to achieve the desired effect, while motor learning time is the time it takes to achieve the necessary physical skill to carry out the action. Similarly, recall time measures the ease with which a user regains competence after a period of disuse of the technique.

Techniques differ in the amount of perceptual, cognitive, and motor work they demand of the user, the rate at which they can be learned, and the rate at which they can be recalled. Here, work is measured in units of time, whereas learning and recall are measured either by the amount of time it takes an inexperienced person to reach a desired skill level or by the extent of improvement possible.

These measures are somewhat vaguely defined. Clearly, there are degrees to the quality of learning. Has a hunt-and-peck typist learned the necessary motor skills to use a keyboard-based technique, or do we give that honor only to the touch typist, who is enormously faster? For our purposes, learning time can be measured as the time it takes to reach a skill level that allows the technique to be used in a practical sense.

Together with the experience of the user, learning and recall time determine the user's skill level, and thereby affect.the primary measure, task time. Task time can be estimated for low skills and for high skills, and the likely skill level can be estimated from the values assigned to learning, recall, and experience.

Memory load. The load on the user's memory comes in both short-term and long-term forms. A technique has a high short-term memory load when the user is obliged to retain unprompted knowledge of task elements over the duration of the task. For example, if a technique requires the user to return an item to a place on the screen after he has manipulated it in some way, he is using short-term memory to remember the place until needed. If the technique provided a prompting cursor, then he would need to remember only the need for the action, and not the place.

Short-term motor memory is required by a user if, for example, he must move his hand back to an object he once held but has since released. Techniques that fail to preserve tactile continuity during use of a physical device such as a mouse, which is normally out of the field of view, put additional load on short-term memory. The hand must be able to grasp the mouse with a minimum of groping.

Long-term memory is required to recall the details for using the technique. In other words, a technique involves a series of steps, which the user must remember before he can be said to "know how to use it." Long-term memory is used in learning the key symbols of a technique, such as an appearing menu, and remembering the shape and identity of objects being manipulated over a span of at least several sequences of tasks. The long-term memory load can be minimized by techniques consisting of a small number of steps and a small amount of key information. Applying regular patterns to all techniques (for example: subject, then verb—always) and prompting the actions as well as the data are also useful ways to reduce long-term memory load. Treu[50] has demonstrated an approach to design based primarily on analysis of the required mental effort.

A technique involves a series of steps, which the user must remember before he can be said to "know how to use it."

The memory load associated with a technique influences the learning time and the ultimate skill of a user. When short-term memory load exceeds the capacity of the user, the effect is poor performance and frustration. When long-term learning requirements are high, learning and recall may be slow.

Fatigue and error. The cognitive form of fatigue has many causes. Most often it is the result of insufficient variety in a regular task, displeasing stimuli, uncertainty, and unrealistic memory loads. Perception is also susceptible to fatigue, primarily from visual and auditory clutter. Motor fatigue is usually the result of poor mechanical design of physical devices: devices that require excessive muscular strength or that cause cramping by requiring action too small for the muscles being used. Fatigue can also be caused by techniques that place limbs in an unsupported position too often or too long. Excessive use of a light pen with a vertically mounted screen is the most common example.

Fatigue affects error rates and user satisfaction (pleasurability) and indirectly affects task times by lowering attention and slowing the reflexes. The literature describing the causes and remedies of fatigue, in all three forms, is extensive. Evaluation of individual techniques for their contribution to an overall fatigue level is difficult because the penalties of fatigue are not observable until fatigue-inducing factors have exceeded a certain threshold for a substantial period.

Convenience. We have grouped the two factors naturalness and boundedness under the heading "convenience" because of their mutual strong effect on pleasurability. Naturalness captures the idea of transfer of activity from other everyday activities. Pressing the foot to slow down some operation is an example of such a technique, taking advantage of analogy to activities most people do regularly, such as using the brake of an automobile. Naturalness is also a consequence of input devices that control displays in ways analogous to action-reaction in the real environment.

Thus, meaningful, familiar icons for light buttons enhance naturalness, as do faces and facial expressions for display of multiparameter data. An orienting technique that always uses a forward roll of a handle to produce a forward roll of the picture and a leftward tilt to produce a leftward tilt of the picture also enhances naturalness.[51]

We caution that our everyday environment often presents conflicting models of behavior. For example, turning a steering wheel left causes the visual image (through the windshield) to move right even though the front of the car moves left. The numerals on hand calculators increase upwards from 0, with 1, 2, 3 in a row immediately above 0. A telephone touch pad, on the other hand, increases downward to 0, with the numerals 7, 8, 9 in a row immediately above 0. Nevertheless, observation and a sensitivity to environmental cues can often suggest that one alternative is more natural than another.

In perception the concept of naturalness tells us what visual forms to use. In cognition it lets us put facts or data in the appropriate order for analytical thinking. Naturalness in motor activity coordinates devices with surroundings and context and gives the user a proper sense of kinesthesia (force and stroke).

Boundedness is a measure of the size of the space in which one must work—perceptually, cognitively, or mechanically. A perceptually bounded technique is one with a limited physical space over which the eyes must move to perceive relevant information and a limited range of sounds to which the ear must be attuned. A cognitively bounded technique limits the range of intellectual space—ideas, concepts, facts—over which the mind must roam to use the technique. Mechanical boundedness measures the distance the user's limbs must move to use the technique.

The effect of context. As we have said, techniques should not be selected in isolation from knowledge of the other techniques in use at approximately the same time. The performance of a complete action generally involves a series of tasks to be carried out almost as a single unit. Indeed, we have suggested that, analogously to human conversations, sequential actions should be naturally grouped into action *sentences.*[5] For example, a user should be able to select an object, position it at a desired location, and affix appropriate labels to it all at once.

Designers must be careful, in selecting the technique by which all of the tasks are to be implemented, to take account of the devices likely to be in the user's hand and of the likely point of his visual focus. Certain time- and memory-consuming suboperations can be avoided, and an otherwise inefficient choice of technique may suddenly become attractive. When a device must be exchanged for another, the selection of a location for an object is slowed down by this significant extra motor activity. If the selection of the object requires the eyes to be focused off screen, then another perceptual activity is involved in identifying a place on the screen to which to move it. An alternative (such as light buttons) in which the eye is already focused on the screen is often preferable. For these reasons, knowledgeable designers often insist on a one-device design and a visually coordinated set of techniques.

The temptation is strong not to analyze the techniques individually at all. It is shortsighted to yield to that temptation, however. The proper approach is to consider each candidate technique in combination with the others likely to be used in sequence before and after it. Such a process is difficult but certainly rational, since more alternatives are considered and they are considered in light of the other performance criteria.

Work is needed to systematically incorporate these considerations into a more global approach to the design of interactions. The recent work of Card, Moran, and Newell[32] shows that, through more detailed analysis of the substeps of a technique (for example, using the interaction technique diagrams we described in our 1981 paper[19]), the advantage of combined interaction techniques to the perceptual, cognitive, and motor load can indeed be quantified.

The effect of user experience. It is clear that experienced users require different choices. The effect of user experience and knowledge can be hypothesized to result in a time compression of perceptual and motor functions to a much greater degree than of cognitive functions. If this is true, then an analysis of the perceptual, cognitive, and motor components of performance of a technique can be used to estimate the technique's relative merit for use by simply comparing the total activity or cognitive activity alone of a new user or a skilled user. The hypothesis needs to be tested, but preliminary comparisons appear to show consistency with intuition.

Interaction tasks and techniques

We suggested earlier that interaction sequences can be decomposed into a series of basic interaction tasks. Each interaction task instance has a set of *application requirements*, defined by the context of the application in which the task is embedded. For instance, a particular positioning task may require dynamic, continuous feedback by means of a screen cursor. A *property* of interaction techniques for positioning is the type of feedback they can provide. In this example only interaction techniques providing dynamic feedback would be considered candidates.

Interaction techniques have not only requirements but also *hardware prerequisites,* which must be met; otherwise, the technique cannot be considered. A positioning technique that provides dynamic, continuous feedback and allows movement in arbitrary directions must be supported by a continuous-motion input device such as a tablet, light pen, or touch-sensitive panel. Furthermore, the display device itself must be able to update a cursor position 20 to 30 times per second. In design situations where interaction devices have already been selected, these prerequisites limit the set of interaction techniques that can be considered. When device selection is part of the design process, the prerequisites link a technique being considered with required hardware characteristics.

Task types and requirements. An examination of interactive graphics leads us to conclude that there are six fun-

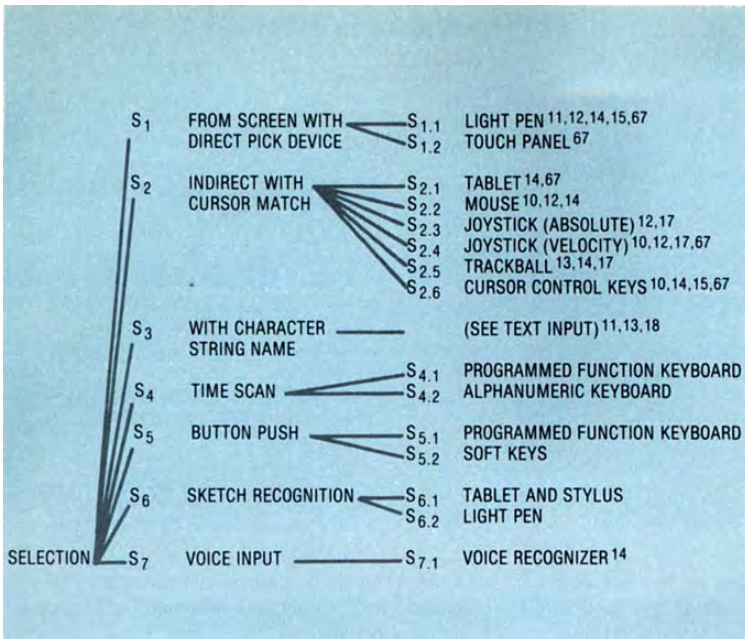

S1 FROM SCREEN WITH DIRECT PICK DEVICE
S1.1 LIGHT PEN 11,12,14,15,67
S1.2 TOUCH PANEL 67
S2 INDIRECT WITH CURSOR MATCH
S2.1 TABLET 14,67
S2.2 MOUSE 10,12,14
S2.3 JOYSTICK (ABSOLUTE) 12,17
S2.4 JOYSTICK (VELOCITY) 10,12,17,67
S2.5 TRACKBALL 13,14,17
S2.6 CURSOR CONTROL KEYS 10,14,15,67
S3 WITH CHARACTER STRING NAME
(SEE TEXT INPUT) 11,13,18
S4 TIME SCAN
S4.1 PROGRAMMED FUNCTION KEYBOARD
S4.2 ALPHANUMERIC KEYBOARD
S5 BUTTON PUSH
S5.1 PROGRAMMED FUNCTION KEYBOARD
S5.2 SOFT KEYS
S6 SKETCH RECOGNITION
S6.1 TABLET AND STYLUS
S6.2 LIGHT PEN
SELECTION
S7 VOICE INPUT
S7.1 VOICE RECOGNIZER 14

Figure 1. Selection techniques.

damental types of interaction tasks. The tasks, which are independent of application and hardware, form the building blocks from which more complex interaction tasks, and in turn complete interaction dialogues, are assembled. The tasks are user-oriented, in that they are the primitive action units performed by a user. They relate to, but differ from, the logical input devices found in device-independent graphics packages, such as those based on the Siggraph Core system,[52] GKS, and GPGS,[53] and the logical input devices discussed by Foley and Wallace,[5] Wallace,[54] and Newman,[55] because the logical input devices are hardware- and software-oriented rather than user-oriented.

The six types of interaction tasks are

- Select
- Position
- Orient
- Path
- Quantify
- Text

These are similar to the tasks described by Ramsey and Atwood[9] and Ohlson.[56] The set of tasks is based not on fundamental research into users' underlying cognitive processes but rather on our experience with dozens of interactive graphics systems and a subsequent categorization of observed interaction activities into these six types. Further study and refinement of the tasks is a key step for future research.

Further refinement of the task types is crucial for many reasons, including the development of user interface management systems and interaction technique libraries, described by Thomas and Hamlin.[57] The toolkits found in the development systems of Lisa, Macintosh, and Visi On represent interaction task categorizations.

Select. The user makes a selection from a set of alternatives. The set might be a group of commands, in which case typical interaction techniques are

(1) menu selection with a light pen,

(2) menu selection with a cursor controlled by a tablet or mouse,

(3) type-in of name, abbreviation, or number on an alphanumeric keyboard,

(4) programmed function keyboard, and

(5) voice input.

Rather than commands, the set of alternatives might be a collection of displayed entities that form part of the application information presentation. In a command-and-control application the entities might be symbols representing troop and equipment positions. Interaction techniques that might be used in this case are similar to those for command selection:

(1) pointing with a light pen,
(2) using a cursor controlled by a tablet or mouse,
(3) type-in of the entity name,
(4) pointing on a touch-sensitive panel, and
(5) voice input of the entity name.

Figure 1 shows a set of selection techniques, which we detail later. As with all six types of interaction tasks, we do not discuss every conceivable technique, as their number is limited only by one's imagination. Rather, we limit the discussion to some techniques that have been proved in use.

The application requirements for a selection task are

(1) size of the set from which the selection is made, if size is fixed, and

(2) range of set size, if variable.

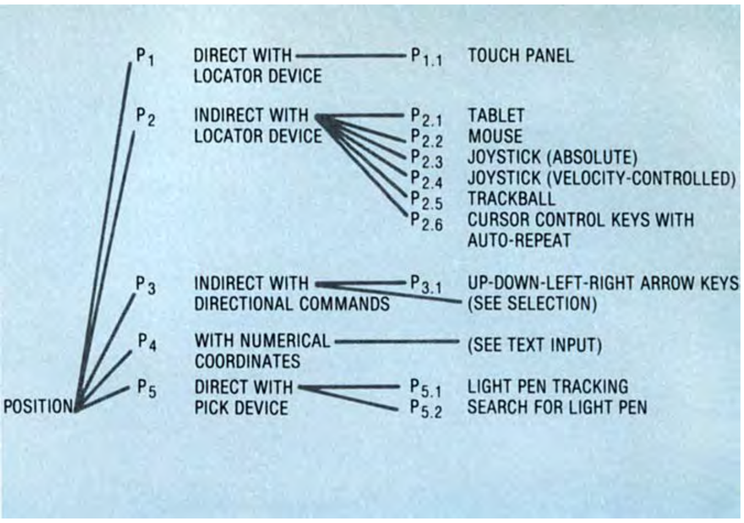

POSITION
P1
DIRECT WITH LOCATOR DEVICE
P1.1
TOUCH PANEL
P2
INDIRECT WITH LOCATOR DEVICE
P2.1
TABLET
P2.2
MOUSE
P2.3
JOYSTICK (ABSOLUTE)
P2.4
JOYSTICK (VELOCITY-CONTROLLED)
P2.5
TRACKBALL
P2.6
CURSOR CONTROL KEYS WITH AUTO-REPEAT
P3
INDIRECT WITH DIRECTIONAL COMMANDS
P3.1
UP-DOWN-LEFT-RIGHT ARROW KEYS
(SEE SELECTION)
P4
WITH NUMERICAL COORDINATES
(SEE TEXT INPUT)
P5
DIRECT WITH PICK DEVICE
P5.1
LIGHT PEN TRACKING
P5.2
SEARCH FOR LIGHT PEN

Figure 2. Positioning techniques.

Rather different techniques might be best for selection from a fixed set of two choices (such as "yes" and "no") and for selection from a very large, variable-size set of displayed entities.

Position. In carrying out a positioning task, the user indicates a position on the interactive display, often as part of a command to place an entity at a particular position. Customary interaction techniques for positioning are

(1) use of a cursor controlled by a tablet, mouse, or joystick,

(2) type-in of the numeric coordinates of the position, and

(3) use of a light pen and tracking cross.

Figure 2 shows the positioning techniques we discuss.

The application requirements of the positioning task are

(1) Dimensionality: 1-D, 2-D, or 3-D. Positioning in 1-D simply means that the position specified is constrained to be along some line.

(2) Open loop or closed loop. In the former case, the user knows in advance the exact coordinates of the position, so visual feedback of the position on the display is not an essential part of the process of specifying the position. In the latter case, visual feedback is important because the user adjusts the position, on the basis of the feedback, until the desired visual end result has been achieved. (This is the distinction between the "discrete positional" and "continuous positional" tasks proposed by Ramsey and Atwood.[9])

(3) Resolution expressed as parts of accuracy over the maximum range of coordinate value. An accuracy of 0.01 inch over a range of 10 inches is one part in 1000.

Orient. The user orients an entity in 2-D or 3-D space. For 2-D, this might mean rotating a symbol to be heading north-northeast. In 3-D, it could mean controlling the pitch, roll, and yaw of the view of a terrain model.

Interaction techniques useful for the orientation task include

(1) control of orientation angle(s) (one angle for 2-D, up to three angles for 3-D) using dial(s) or joystick, and

(2) type-in of angle(s) using alphanumeric keyboard.

Figure 3 shows the different interaction techniques used to implement an orient task.

The application requirements of the orientation task are analogous to those for the positioning task. Dimensionality is replaced by the more general term "degrees of freedom," values of which can be one, two, or three. Of course, it is only in 3-D space that two and three degrees of freedom make sense: in 2-D, only a single degree of rotational freedom is available. On the other hand, one degree of freedom in 3-D makes perfectly good sense: it is a rotation about an arbitrary axis.

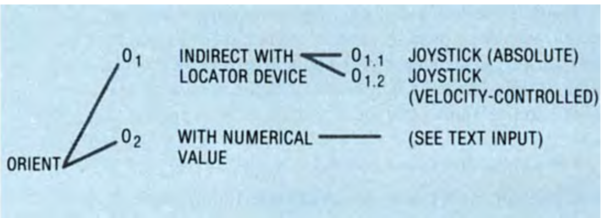


Figure 3. Orienting techniques.

Path. The user generates a path, which is a series of positions or orientations created over time. We consider generating a path a fundamental interaction task, even though it consists of other primitive tasks (position or orient), because another fundamental dimension—time—is involved and because we believe this changes the user's perception of the task. With a single position or orientation, the user's attention is focused on attaining a single end result. But in path generation the series of positions or orientations and their order are the focus of attention.

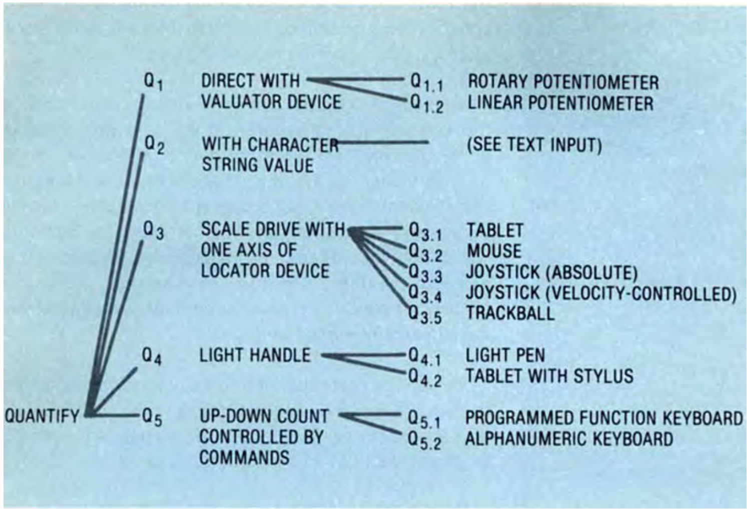


Figure 4. Quantifying techniques.

A path of positions might be generated by a user in the process of digitizing a sketch, indicating the routing of a run on a printed circuit board, or showing a desired route on a map. A path of orientations (and of positions) would also be necessary in a simulated flight over a terrain model.

The techniques for generating a path are usually position and orient task techniques that allow closed-loop feedback, typically involving use of a tablet, mouse, joystick, and/or dials. In some cases open-loop techniques might be suitable.

The application requirements of a path task are

(1) Maximum number of positions or orientations along the path, if they are to be saved. For instance, positions would be saved when digitizing a shape, but might not be saved in a flight simulation.

(2) The interval between each element on the path, and its basis. Some paths are time-based, with a new element entered at each periodic time interval (typically 33 ms for a real-time simulation). Other paths are distance-based, with the next element entered each time it differs from the preceding element by a predefined amount.

(3) Dimensionality: 2-D or 3-D.

(4) Open loop or closed loop.

(5) Resolution.

(6) Type: position, orientation, or both.

Quantify. The user specifies a value (i. e., a number) to quantify a measure, such as the height of an entity, or the value, in ohms, of a resistor. Typical techniques are

(1) value type-in on a keyboard, and

(2) rotary or slide potentiometer.

Figure 4 shows the set of quantifying techniques we will discuss.

The application requirements of a quantification task are

(1) Resolution, expressed as number of resolvable units to be specified. For instance, age in years would require about 120 units of resolution, while angle in degrees requires 360 units.

(2) Open loop or closed loop.

Text. The user inputs a text string, used for example as an annotation on a drawing or as part of a page of text. The key factor is that the text string itself becomes part of the information stored in the computer, rather than serving as a command or being converted to a value, a position, or an orientation. In other words, the text input is a new interaction task, not an intermediary step in one of the other interaction tasks. Typical interaction techniques for text input are

(1) type-in from an alphanumeric keyboard, and

(2) character selection from a menu.

Figure 5 shows the text-entry techniques to be discussed.

The text task has two application requirements:

(1) size of character set and

(2) maximum length of string to be entered.

There are other issues surrounding the text input task, such as the specific character set (as opposed to its size). Such issues, however, do not affect the choice of technique or device. The details of the character set would affect only the labels on key caps, for instance.

In summary, the task requirements, specified under the categories just outlined, limit the choice of techniques to those whose properties match the requirements. The set of requirements for each task is derived from an analysis of the needs of the application being implemented. Table 1 summarizes the requirements for each task.

Controlling tasks. None of the six interaction tasks directly modifies the objects being displayed. If such a modification is needed, the user can achieve it by performing a selection (in particular, a command selection) to invoke a picture-modifying program, using as operands data developed from other tasks.

A number of tasks, nevertheless, have as their basic purpose the control of objects already visible on the display. These tasks are elementary in the sense that the user cannot divide them into a sequence of other elementary tasks. They are, on the other hand, closely related to the tasks we

Table 1.
Summary of interactive task requirements.

Task	Requirements
Select	Size of set, if fixed Range of set size, if variable
Position	Dimensionality: 1-D, 2-D, or 3-D Open loop or closed loop Resolution
Orient	Degrees of freedom: 1, 2, or 3 Open loop or closed loop Resolution
Path	Maximum number of path elements to be retained Type of interval between each element on path Size of interval between each element on path Dimensionality: 2-D or 3-D Open loop or closed loop Resolution Type: position or orientation or both
Quantify	Resolution Open loop or closed loop
Text	Size of character set Maximum length of string

have already described. We refer to them as controlling tasks, because they characteristically control something, rather than specify something (as do the elementary tasks). There are four controlling tasks, named for the type of modification they effect on an object:

- Stretch
- Sketch
- Manipulate
- Shape

Techniques for implementing these tasks are called controlling techniques.

Stretch. The user grasps a particular feature and moves it to a new position, leaving remaining features of the object in place. The result is a distortion of the shape of the object, much like stretching a rubber mask or a collection of rubber bands. Typical stretching techniques are

(1) stretched lines,
(2) stretched horizontal or vertical lines,
(3) stretched vertices (lines possessing a common endpoint),
(4) horizontal-vertical connections (called a zig-zag—see page 41), and
(5) stretched polygons, prisms, and pyramidal forms.

These techniques are all based on positioning techniques and carry the same prerequisites and requirements. They are most useful when the feedback is continuous but can exist in both continuous and discrete feedback forms.

Sketch. The user, manipulating a locating device as if it were a brush or pen, creates an object by "freehand sketching." Line structure (thickness, dot-dash character, color, etc.) may be specified as part of the brush form. Sketching is the controlling version of the path task. It shares all forms and requirements of pathing, and in addition has a line-style or brush-form requirement, which specifies the attributes of the sketch lines left on the screen after the device motion has taken place.

Manipulate. The user causes an object to move about in the viewable space, by either translation or orientation under the control of an input device (a locator). We arbitrarily include scaling as a variant of this task. Manipulation techniques are described as either dragging, twisting, or scaling, depending on whether they are based on translation, orientation, or valuation techniques. Dragging and twisting differ from the elementary orientation technique because the cursor or gnomon is replaced by an already existing object on the screen.

Shape. The user causes a smooth, curved line or surface to change its general shape according to the placement of a positioning device. Techniques for shaping are described in greater detail in a later section.

Organization of interaction techniques. We now turn our attention to the interaction techniques used to implement the interaction tasks. Figures 1 through 5 show how we have organized the techniques. The lists are by no means exhaustive, but we believe the organization will easily cover other techniques as well.

Techniques and variations. At the first level in these treelike diagrams, we have the fundamentally different techniques, such as menus and command type-in for the selection task in Figure 1. At the second level are variations on a basic technique, such as the specific physical device used to drive the cursor for selection from a menu. In cases where the technique draws on techniques normally associated with other interaction tasks, the diagrams simply refer to another diagram.

Technique parameters. Another aspect to interaction techniques is not shown in these diagrams but does affect the characteristics of individual techniques. We call this aspect the technique "parameters." For example:

(1) the form of the cursor used in connection with some positioning and selection techniques,
(2) the ratio of hand movement to cursor movement when a tablet, joystick, mouse, or other physical positioning device is used, and
(3) the layout of a menu as either a row, column, or grid of choices.

One might include hardware device characteristics, such as the length or diameter of a joystick, as technique parameters. However, following our basic tenet of taking hardware as a fixed given, we do not do so. Instead, we limit technique parameters to those aspects normally controllable by software.

In our later discussion of specific techniques, we describe some technique parameters. Like basic techniques themselves, the types of parameters associated with them are limited only by our imagination and creativity. Accordingly, we cannot be exhaustive, but rather we address the most substantial parameters, especially those for which human factors literature offers guidance.

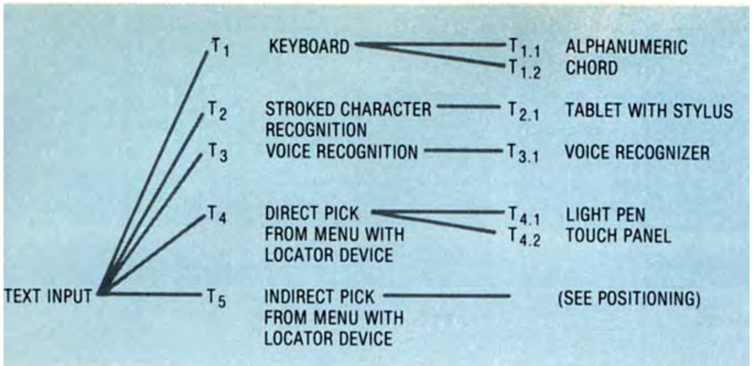


Figure 5. Text-entry techniques.

Hardware prerequisites. Each of the techniques (as opposed to technique variations) has a set of hardware prerequisites, with respect to both the display technology and the types of devices used with the technique. A typical prerequisite, say for a closed-loop positioning technique, would be a continuous-movement physical device and a display on which the feedback to the user can be dynamically repositioned 15 to 30 times per second.

Interaction techniques

The choice of an interaction technique is not an "either/or" proposition. In fact, it is usually desirable to make available several techniques for the same task, to suit different user characteristics. For example, command selection is performed in many systems by either menu selection or function key. The former technique is generally preferred by users who are not familiar with a certain part of a system; the latter, by experienced, frequent users who have memorized the function key meanings.

If multiple techniques are available, provision of a smooth *transition path* for moving from one technique to another is essential. In the preceding example, the transition path is simply for the user to begin using the function keys.

Selection techniques. A selection technique typically involves picking an item from a list of alternatives. Typical applications are command selection, operand selection, and attribute selection. An inherent pick device is the light pen, but any positioning device can simulate a pick by placing a cursor over a displayed representation of the selection desired.

Command selection. A menu of commands is displayed, typically in list form. The desired command is selected from the presented set of alternatives. Direct selection devices such as the light pen or touch panel, indirect selection devices such as tne mouse or tablet, or devices that can simulate selection, such as an alphanumeric keyboard or a physical locator, can be used.

Menu selection is most commonly used for command entry. There are also operand menus—e.g., a menu of capacitors, resistors, etc.—for a circuit design application. Also common are property or attribute menus.

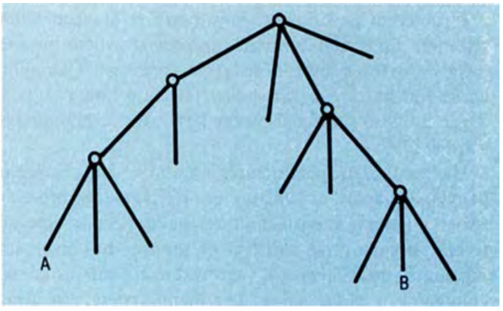


Figure 6. A four-level hierarchy menu tree.

Before choosing a selection technique, menu designers must consider the following parameters of the menu itself:

(1) Organization: single-level vs. hierarchical. If the set of alternatives is small enough to be contained in the space the screen provides, a single-level menu can be used. Otherwise, there are two possibilities: a hierarchical menu or a single-level menu requiring several sequentially displayed screens to view. In either case, "navigational aids" will be needed to move through the selections.

With a hierarchy, the user chooses a phase from the main displayed menu and then selects subphases from subsequently displayed menus until he finds the desired command. Sometimes the user has to go through several menu phases and subphases before he finds the desired entry. Applications may also require the user to refer back to the main menu to make another command selection.

Suppose an application has a four-level hierarchy tree defined by the phases shown abstractly in Figure 6. The leaf nodes of the tree represent the commands, and the internal nodes represent groups of commands. When a selection is made, we travel down the tree from its root toward its leaves. Each selection moves us one level further down, until we reach a leaf (i.e., a command). Suppose command A has been selected, and command B is desired next: the user must be provided with the controls to climb back up the hierarchy to the first node above both A and B, and then to descend toward B. This can be done by control commands such as "move to the top of the hierarchy," and "move up one level in the hierarchy." If

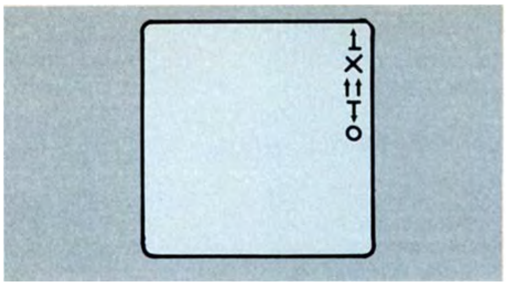


Figure 7. An iconic command menu.

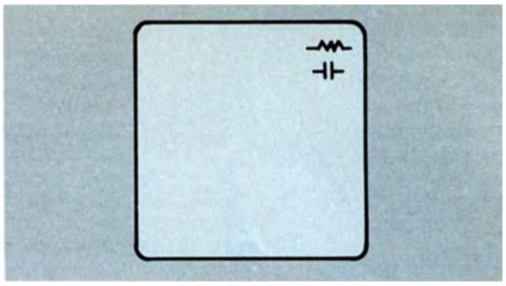


Figure 8. An iconic operand menu.

the tree is too big, the capability of going directly to a few frequently used nodes of the tree is particularly useful.

In a recent experiment, Snowberry et al. found that selection time and accuracy improved when broader menus with fewer levels of selection were used.[58] Doughty and Kelso have reported similar selection time results,[59] which are also predicted by the keystroke-level model of Card et al.[31]

Hierarchical menu structures almost demand an accompanying keyboard or function key technique for more experienced users. These techniques make selection especially easy if each node and leaf of the tree has an unambiguous name, allowing a user to directly enter a known command or phase name. The menu system provides a backup if the user's memory fails. An alternative is to require unambiguous names for each entry within an individual menu. Then the experienced user, seeking to avoid direct interaction with menus, can enter the complete path name to a leaf node.

In the case of a single-level menu, commands are needed to "flip pages" forward and backward. Each of these commands, like those for traversing a hierarchy, represents another selection task. That is, a single complex selection task has been converted into a larger number of simpler (but potentially still complex) selection tasks, simply because of space limitations on the display.

(2) Item order. There are several reasonable ways of sequencing command entries in a menu:

- alphabetical;
- frequency of use, the most frequent appearing at the top of the menu; and
- logical, placing entries of the same category together. (This is normally the way that commands would be grouped in a hierarchy.)

Uber et al. recognized many years ago that all three ways can be used together or separately to construct menus.[60]

Card has experimented with order for an 18-entry menu of text-editing commands. He found that, for new users, selection from an alphabetical order was about 35 percent faster than from an order based on functional (logical) grouping. Random order was four times slower than alphabetical order. As users became experienced, considerable improvement in selection speed occurred, with alphabetical and functional order becoming essentially equivalent, but with random order still being slower.[61]

(3) Representation: iconic vs. textual. While menus are often in text form, they can also be in graphic form. A set of graphical symbols, known as icons, can be used to represent commands (Figure 7) or operands (Figure 8). Iconic menus can be designed to occupy less screen space than text menus and thus are more compact. Icons can also decrease the cognitive load of menu selection, if the icons are more immediately evocative of their meanings than the equivalent text strings.

Hemenway discusses some of the issues involved in icon design and gives examples of icons used for objects, operands, and object/operand pairs.[62] Icons representing objects are probably the most readily recognized, while recognition of icons representing actions depends on the obviousness of the relation between the icon and the action (scissors imply cutting, a glue pot implies pasting). The menus for the Lisa, Macintosh, and Star computers contain many examples of icons used to represent objects and attribute values. Lisa and Macintosh icons are shown in Figures 9 and 10.

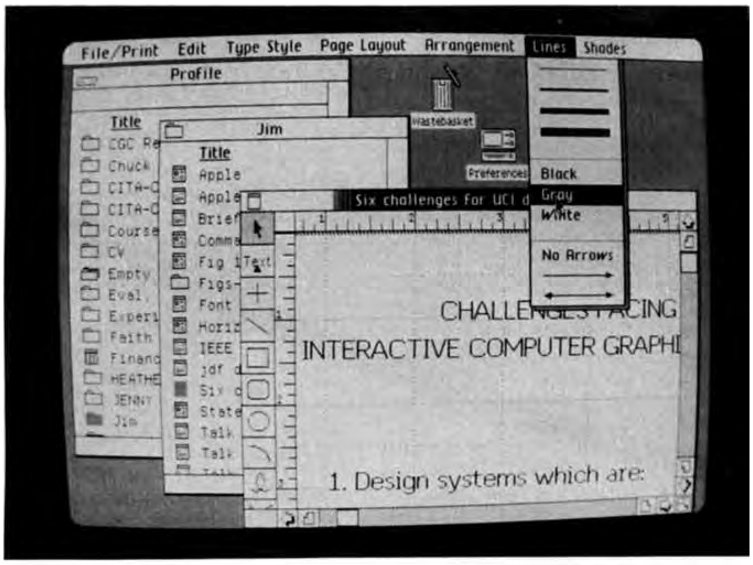
Bewley and his coworkers at Xerox performed extensive testing of icons being considered for use in Star.[63] A series of timed recognition and naming tests were used to evaluate four preliminary designs as the basis for a minor redesign of the selected preliminary design.

(4) Position: static vs. moving (i.e., pop-up). A menu that is always in the same position on the display screen is called static. A static menu can be

- part of the same screen as the main display;
- on an auxiliary screen next to the main display screen;
- on the same screen as the main display, but replacing the main display when the user wants to make a selection—i.e., the user has to switch back and forth;
- printed on a tablet.

A menu printed on a tablet, as shown in Figure 11, can be used for fixed-application systems. One general use of this technique is to imprint a keyboard image on the tablet to simulate type-in for users who don't type. The use of a tablet or an auxiliary screen means that the user has to look away from the application display, hence destroying visual continuity. The advantages are the saving of display space, which is often at a premium, and the accommodation of a large set of commands in one menu, often not possible by using only the application display.

A pop-up menu, which appears when a selection is to be made, is feasible if a positioning device can be considered for implementing the selection. The menu always appears near or at the screen cursor, which is usually in the



CHALLENGES FACING
INTERACTIVE COMPUTER GRAPHI

1. Design systems which are:

Figure 9. A Lisa screen showing icons representing objects (wastebasket, preferences), attributes (shades), and object/command pairs (the geometric figure icons mean "create one of these," i.e., "create a square").

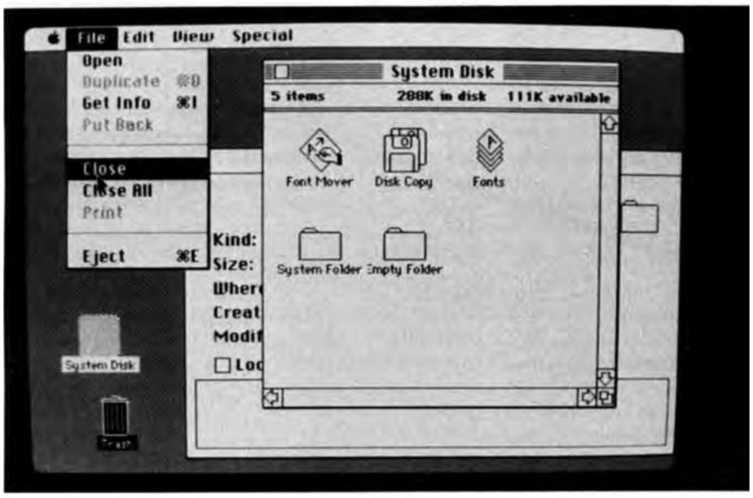


Figure 10. A Macintosh screen showing icons representing objects, and a pull-down menu. Command abbreviations on the right side of the menu can be entered from the keyboard, allowing experienced users to work more rapidly by avoiding the menu.

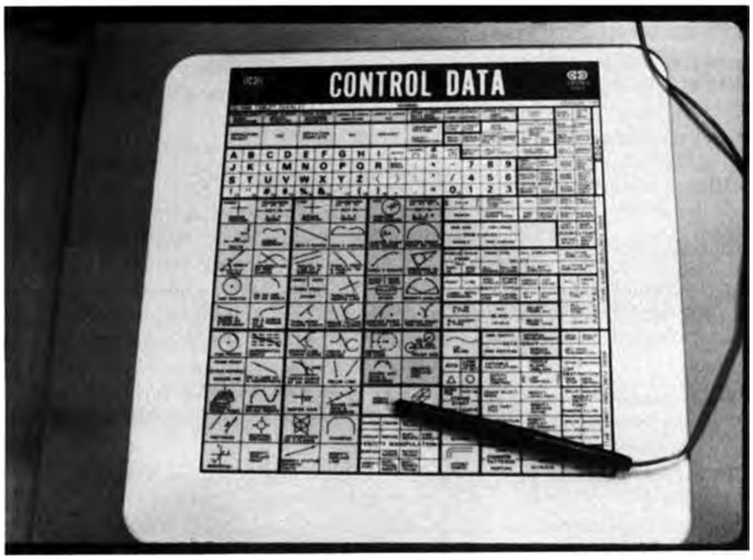
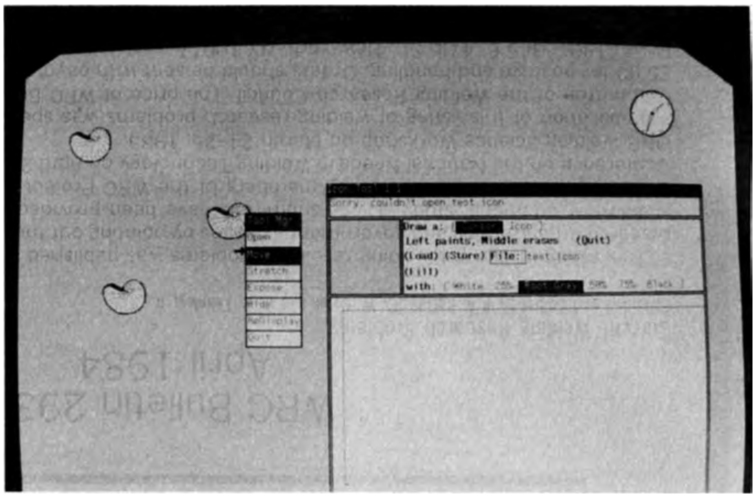


Figure 11. A data tablet overlay with command and keyboard entry areas. Notice the double coding of many commands with both a name and an icon.



sorry, couldn't open test.icon

Draw a: (Cursor Icon)
Left paints, Middle erases (Quit)
(Load) (Store) File: test.icon
(Fill)
with: (White 25% Root Grey 50% 75% Black)

Figure 12. The Sun Microsystems window manager. The pop-up menu is called up by holding down a mouse button. As the cursor is moved up and down the menu, the currently selected command is shown in reverse video. Releasing the mouse button invokes the command and causes the menu to disappear.

vicinity of the user's visual attention. Figure 12 shows an example of a pop-up menu. Although a pop-up menu preserves visual continuity, it cannot take advantage of "muscle memory" as well as a static menu. However, it has the advantage of requiring only minimum hand movement when a user makes a selection. Pop-up menus were used in the late 1960's by N. Wiseman at Cambridge and were first adapted to raster displays by W. Newman in about 1975.

(5) Visibility: always visible vs. sometimes visible. Pop-up menus are in the "sometimes visible" category, while static menus can be in either. The "pull-down" menus of Lisa and Macintosh are static and sometimes visible, while their menu bars across the top of the screen are static and always visible (see Figures 9 and 10).

(6) Organization: horizontal, vertical, or blocked. Menu items can be listed vertically or horizontally. Studies by both Coffee[64] and Earl and Goff[11] show no difference in search times between horizontal and vertical lists. Items also can be grouped into clusters, with extra space between the groups. Cropper and Evans showed such grouping to be effective in improving search time in tabular data.[65]

(7) Target size: Fitts's Law. Display targets, whether alphanumeric or iconic, should be as large as possible in order to reduce positioning time and error rate. This recommendation is based on the experimental evidence of Fitts's Law, which predicts that the hand movement time to position a target from one location to another increases with the distance moved and decreases with the size of the target.[66] (See also Card et al.[10]).

To make menu selections, many interaction techniques and variations of techniques are possible. We limit our discussion to some commonly used command selection techniques:

Character string name type-in. A command menu is displayed. The desired command on the menu is typed in on an alphanumeric keyboard. In this case the menu is solely a memory aid, used as a prompt rather than as an integral part of the command entry process. Of course, the menu need not be shown, in which case we have a traditiońal command-language-driven system.

Label type-in. The label associated with a menu item is typed in. This technique is a variation of the name type-in technique described above. A menu of system-defined commands and their corresponding labels is displayed. To make a selection, the user types in the label representing the command instead of the command itself. The labels can be numeric codes or mnemonics. This shortens input time and reduces typing errors. The menu is still partly a memory aid; the experienced user remembers the label to enter without looking at the menu, while the inexperienced user consults the menu.

Direct pick by light pen. A light pen is used to pick the desired command entry directly from the displayed command menu. Pointing with a light pen has a certain naturalness not found in most other devices.

Touch panel pick. The user touches a finger to the desired menu choice on the display screen. The prerequisite device is a touch-sensitive panel. This technique is very attractive because no intermediary device is needed. The user's finger becomes the picking device. The user's motor load is lower than for pen picking, because the user does not have to physically acquire a device prior to making a selection.

Simulated pick with cursor match. A cursor is positioned over the desired menu choice, and a button is depressed. The system automatically matches the position of the cursor to the nearest command, taking it as the desired command. Precise positioning of the cursor is not required. A pick can be effectively simulated with a continuous locator such as a tablet, a mouse, or a joystick. A discrete locator (an up-down, left-right cursor) can also be used, though it takes longer and can be very awkward.

Function key. A unique function key associated with the command is depressed. A bank of buttons, each but-

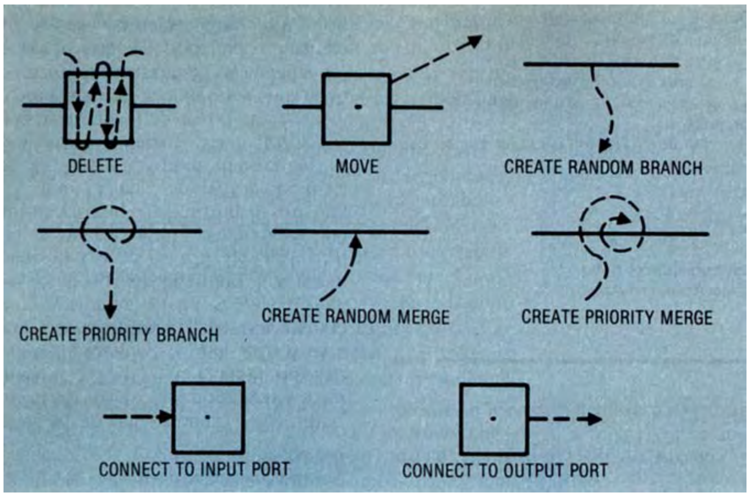


Figure 13. A series of sketch patterns.

ton corresponding to a command, is the prerequisite device.

Button push—soft keys. A button associated with a command is depressed to enter the command. Commands are presented as a series of labeled buttons. The buttons are so-called "soft keys" located on the edge of the display area with their labels displayed on the screen, and thus their meanings can be changed readily.

Voice recognition. The user speaks the name of the selected command, and a word recognizer determines which of a set of known words was spoken. Voice input is a simple means to distinguish commands from data; commands are entered by voice, the data by keyboard or other means. In a keyboard environment, this removes the need for special characters or modes to distinguish data and commands. This method might be preferable for nontypists as an input means for alphanumeric data, provided that voice recognition technology is able to accurately recognize a large set of words, letters, and numbers.

Sketch recognition. The user makes sequences of movements with a continuous positioning device such as a tablet, a mouse, or a light pen and tracking cross. The sketch recognition system recognizes the sequence to determine what command is being entered (see also character recognition). More specifically, the user sketches a simple pattern. The sketch recognizer automatically matches the pattern with a set of defined patterns, each of which has an associated command. Figure 13 shows one set of sketch patterns and their related commands. Some of the commands are unique to the design of queuing networks, the application from which this example is taken.

The technique requires no typing skill and preserves tactile continuity. Furthermore, if the command requires that a position be specified, the place where the sketch is done can serve as the position. Similarly, if an operand is required, the sketch can be done "on top of" the operand if it is part of the displayed image. Skilled operators can work very fast with this technique.

Operand selection. Operands are normally objects created interactively by a user—elements of an IC mask or of an architectural drawing, for example. Very often such objects are organized hierarchically. Controlling the hierarchical level at which selection occurs is a concern that transcends specific techniques.

A pick device has no inherent notion of hierarchy, yet the user's intent must somehow be made known to the application program. One way to achieve this goal is to design the interaction commands so that the level of object to be operated on can be inferred by the program without explicit operator action. For example, in a phase having to do with placing houses in a subdivision, the entire house would be designated; in a phase for designing individual houses, the house components would be selected.

If the intent is not implicit, then the command language must provide the user with explicit means for conveying his purpose. Commands like "move door" and "move house" illustrate such means. On the other hand, if the hierarchical level at which the user operates changes relatively infrequently, it is more convenient to have a separate command that sets the level at which all ensuing picks will be made. The user respecifies the level whenever necessary.

Another approach is required if the number of hierarchical levels of objects is unknown to the system designer and is potentially large (as in a drafting system where templates can be defined to contain both other templates and basic objects like lines, points, and arcs). Two user commands are required: "travel up the hierarchy" and "come back down." When the user picks something, the system highlights the lowest level object seen. If this is what the user wants to pick, he can proceed. If not, he issues the first of the two commands: "travel up the hierarchy." The entire first-level object (the house, say) of which the detected object is a part is highlighted. If this is not what the user wants, he travels up again and still more of the picture is highlighted. If he should accidentally travel too far up the hierarchy, he reverses direction with the "come back down" command.

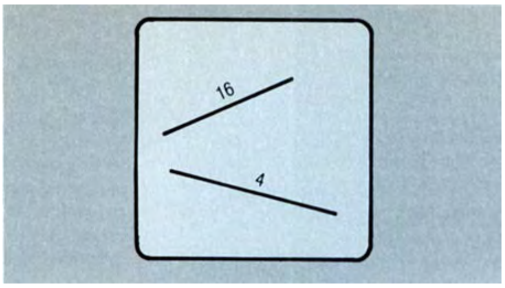


Figure 14. Labeled lines.

In some text editors, there is a character-word-sentence-paragraph hierarchy. Using the Xerox Star text editor, for instance, the user selects a character by positioning the screen cursor on the character and pushing the "select" button (one of two) on the mouse. To choose the word, the user quickly pushes "select" again. Further moves up the hierarchy are accomplished by further pushes.

Again, we limit our discussion to the most common operand selection techniques:

Cursor match. A positioning device such as a locator is used to move the displayed cursor close to the desired operand. The system automatically matches the position of the cursor to the nearest item, taking it as the desired selection. Precise positioning of the cursor is not necessary; it need only be placed nearer the item to be selected than to any other item.

Picking. The operand is selected with a pick device.

Label type-in. The user selects the desired operand by typing in the label of the operand on an alphanumeric keyboard. The technique is very similar to command type-in except that the label of the operand is being entered rather than the command. The label is typically displayed along with each potential operand. For example, we might have the display of Figure 14, which shows two lines and their labels—in this case the integers 4 and 16 (alphanumeric labels can be used, of course). To delete line 4, the delete command would be entered, followed by the number 4.

Time scan. The system displays a screen of possible choices. Each displayed entity is then successively increased in intensity for a short period of time. When the desired entity brightens, the user activates a button to indicate the selection.

A group of buttons and a refresh display are needed for the technique. Usually several entities are likely to have brightened during the brief moment between the desired entity's brightening and the button activation. Thus another button is used to reverse the brightening sequence, one item at a time. When the correct entity is again brightened, the user activates a third button to actually make the selection.

This technique is especially useful if items are close together on the screen, making it difficult for the user to pick with the light pen or the cursor. An example of an application of the technique is to pick a single atom from a large molecule.

Studies and evaluation of selection techniques. Table 2 shows estimated rankings of selection techniques, based on our readings, the experiments, and our experiences. The ratings are relative and are only our best estimate. They are far from sacrosanct.

The techniques involving selection from the screen with a pointing device have been relatively extensively studied, as indicated by the studies cited in Figure 1. The annotated references at the end of this article describe the techniques studied in most of these experiments.[10-15,18,67] One sees from Figure 1 that many of the techniques have not been studied at all.

The experiment by Card et al.[10] found the mouse superior to a velocity joystick and cursor control keys. Similarly, English et al.[12] found that for experienced users the mouse is superior to a light pen or an absolute joystick, and that for inexperienced users the light pen is marginally better than the mouse and superior to both the absolute and velocity-controlled joysticks. These two experiments suggest that neither the joystick nor the cursor control keys are as satisfactory as the mouse or light pen. The tablet used in the latter experiment was mechanically coupled and not typical of contemporary tablets; we thus offer no conclusions about tablets. Goodwin[15] confirmed the superiority of the light pen to cursor control keys.

The study by Albert[67] used only inexperienced subjects for a small number of experiments in which users selected 20 targets in rapid succession with size in the half-inch to one-inch range. He found the selection speed from fastest to slowest to be touch panel, light pen, data tablet, trackball, force (static) velocity-control joystick, normal velocity-control joystick, and cursor control keys. These are consistent with and hence reinforce the earlier results. We point out, however, that the touch panel result will not hold up for selection of very small and closely spaced targets.

More recently, Haller et al.[14] studied various devices and found the light pen faster (2.1 seconds) than the tablet (3.7 seconds) and the mouse (3.6 seconds) for selecting highlighted text. These times are not consistent with other investigators' measurements. However, the conditions of Haller's experiment do differ from the others. Unfortunately, differing conditions often create difficulties in integrating the results of various experiments.

Mehr and Mehr[17] found that the trackball is superior to several different types of joysticks in moving a cursor to a target (which is to be selected). On the basis of all these results, we are inclined to dismiss the joystick as a selection device.

Comparing name type-in on a keyboard versus selection from a menu on the screen, we have contradictory results. Fields et al.[13] found the keyboard and menu selection with a trackball equal in speed (although menu selection was more accurate, as typing errors were precluded). In contrast, Earl and Goff[11] found light pen picking faster (as well as more accurate) than keyboard type-in. A crucial additional factor to consider is the size of the menu. In Earl and Goff's work it was small (up to 18 items); in Fields et al.'s it was large (up to 40 items), so search time would naturally work to the disadvantage of the menu. Also, in both experiments the subjects already knew which menu selection they were seeking; thus the menu was not serving one of its useful roles—a memory aid to indicate the set of available choices.

Both experiments compared menus to the keyboard type-in technique in environments in which the *name* of the desired selection is given to the operator, and he must locate *where* the selection is on the menu. If the user were making the selection from a displayed drawing with which he was interacting, we would expect different results. The user would already know where an item of interest was located and could thus quickly point at it. Remembering (or creating) a name to be typed would take longer.

Thus, between selection techniques, we see a difference based upon the form of the knowledge the user brings to the task. We will see the same thing in the positioning task, where the distinction is between knowing the position as a place on the display or knowing the coordinate values. In the selection task, the distinction is between knowing the selection as a place on the display or as a name. The experiments deal only with the latter case, in which the name is known and the location in the menu is not. This spatial-versus-linguistic distinction in the form of knowledge is fundamental to the selection of interaction techniques.

There are very persuasive anecdotal stories about menu-with-light-pen systems in the hands of experienced users who, through practice, know the position of a desired menu item. Users are reported (by reliable observers) to have their light pens poised to make a selection, even before the menu actually appears on the screen. In such cases computer delays in presenting the menu actually slow down the interaction. This is a problem if the computer supporting the application is time-shared.

Positioning techniques. The positioning task involves specifying a position in application coordinates. The requirements of the task, determined by the application, are dimensionality, resolution, and closed-loop or open-loop feedback. Before discussing specific interaction techniques for positioning, we must consider several general issues that transcend specific techniques but are relevant to some or all of them. These are some of the parameters of interactive positioning techniques:

(1) Coordinate systems. The user of an interactive graphics system is typically aware of up to three coordinate systems: the *application* coordinate system, in which the computer maintains coordinates; the *screen* coordinate system, in which the user views an image; and the *positioning-device* coordinate system, in which the user moves a tablet, joystick, mouse, or other device. At issue is the relationship (i.e., the geometric transformation) between these three coordinate systems. It is important because it determines the relation between user hand movements and graphic object movements on the screen. Empirical observations by Britton et al.,[51] suggest there should be no rotation in the positioning-device-to-screen transformation. This means a movement of the hand to the right should cause the screen cursor or other graphic object also to move to the right. This should be true even if the viewing transformation from world to screen coordinates does include a rotation.

(2) Cursor form and visual aids. For positioning techniques involving movement of a displayed screen cursor to a desired location, the user must first find the screen cursor. Not surprisingly, studies have shown that the more alike the background items and the target are, the longer it takes to visually acquire the target. Thus we should consider several things when choosing the form of the screen cursor. On an alphanumeric display we should choose a cursor form that is distinctively different from any of the alphabets, numerals, or special characters. Hence a box- or diamond-shaped form is a much better choice than an underline or a cross form. For other graphics applications, it is important to choose a cursor form that is different from the commonly used grapics forms.

Table 2.
Comparison of selection techniques.

	Ergonomic Measures								
Techniques	Cognitive Load	Percep-tual Load	Motor Load	Visual Acqui-sition	Motor Acqui-sition	Ease of Learning	Fatigue	Error Proneness	Feedback Type
Light pen	L	L	M	L	M	L	M	L	NA
Touch panel	L	L	L	L	L	L	L	L	NA
Indirect, cursor match				[See Quantifying]					NA
Character string name				[See Text Entry]					NA
Time scan, programmed function keyboard	H	H	M	H	M	M	H	H	NA
Time scan, alpha-numeric keyboard	H	H	M	H	H	M	H	H	NA
Uniquely labeled button, programmed function key	L	L	L	M	M	L	L	L	NA
Uniquely labeled button, soft keys	L	L	L	M	M	L	L	L	NA
Unique movement, tablet & stylus	H	M	M	H	M	H	H	M	NA

H=high, M=medium, L=low

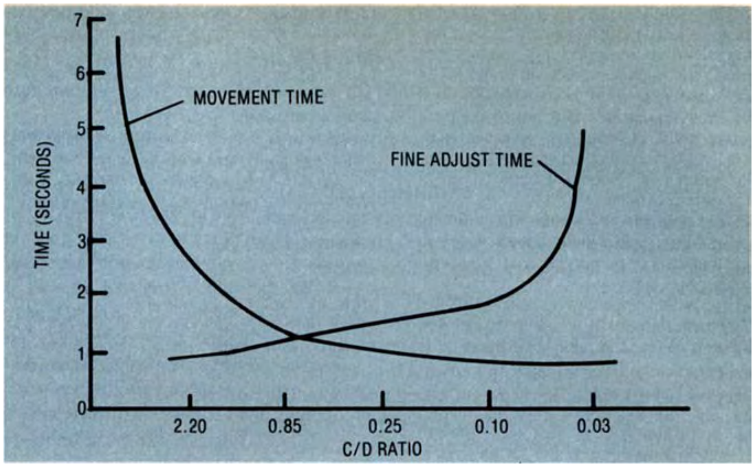


Figure 15. The effect of C/D ratio on movement time and adjustment time (adapted from Jenkins and Connor[69]).

The cursor can also be differentiated from the rest of the displayed information by visual aids such as intensity, color, or blink. Vartabedian[68] found that a box-shaped cursor blinking at 3 Hz was optimal for effective searching and moving on an alphanumeric display. Other cursor forms or blink rates took longer to acquire visually.

An important visual aid for many positioning tasks is a grid superimposed (perhaps at low intensity) on the drawing, to help in aligning positions or objects. A grid (or coordinate axes along the edges of the display) is also helpful if the user must convert a position on the screen into numeric coordinates for keyboard entry.

(3) Control/display ratio. The C/D ratio is the ratio of distance traveled by the control (the hand movement of the operator, the movement of the joystick, etc.) to that of the moving element on the display (target, screen cursor, etc.). Studies by Jenkins et al.[69-71] have shown that the C/D ratio of a control device is critical to the operator's performance. For linear controls, say a tablet with a stylus, the C/D ratio is defined by the formula

$$C/D = \text{movement of hand/movement of cursor}$$

For rotary controls, such as the trackball and the joystick, the C/D ratio is defined as

$$C/D = \frac{\text{(fraction of circle movement)} * \text{(diameter of circle)}}{\text{(movement of cursor)}}$$
$$= (A/360) * (2 * \pi * L) \,/\, \text{(movement of cursor)}$$

where A = the travel of the control device (in degrees), and L = the length of the control device.

Generally speaking, a low ratio is good for fast movement and a high ratio is good for fine adjustment accuracy. Figure 15 illustrates the effect of C/D ratio on movement time and adjustment time. The optimum C/D ratio is that which produces the least total time to use a control. Experience (not experiments) suggests that for tablets the workable C/D ratio ranges from 1 to 0.3. For knobs and dials the optimum C/D ratio usually falls between 0.2 and 0.8. For example, a 5-inch joystick with 90 degrees of movement has about 8 inches of travel. Used with a typical 12 × 12-inch screen, the C/D ratio is about 0.67. However, fine positioning with a joystick requires grosser motor movements than with a mouse or tablet, making the device difficult to use for fine adjustments. Note that the C/D ratio is just the scale factor that relates the positioning-device coordinates to screen coordinates, as described in item 1 above. A fixed C/D ratio is not necessary. For example, the Macintosh mouse has a lower C/D ratio when it is moving rapidly than when it is moving slowly.

(4) Feedback. Inherent in graphics applications is the type of feedback provided. A continuous translation technique, whether it is direct (using a light pen or touch panel to show a position), or indirect (using a locator to show a position by moving a screen cursor), implies closed-loop dynamic feedback as either the cursor or the object of interest moves across the display. This allows the user to move the screen cursor through a succession of trial positions until the results are satisfactory. Therefore, continuous feedback is appropriate when the user knows where on the screen the position of interest is but does not know its coordinates.

Conversely, discrete feedback is appropriate when the user knows the coordinates but not the desired positions. The user types in the coordinates, and the object or screen cursor is repositioned appropriately.

These forms of discrete and continuous feedback can be combined, continually moving the cursor and simultaneously displaying its coordinates in numeric form. When a user has a spatial position task, he watches the cursor feedback; given a desired coordinate position, he watches the numeric readout instead.

(5) Absolute and relative locators. An absolute locator indicates position with respect to the absolute origin of its control movement. Hence, its range is limited by its physical size. The tablet is a typical absolute locator.

A relative locator, in contrast, indicates position relative to its control movement. The mouse, the trackball, and the velocity-control joystick are some of the relative locators. For example, using a mouse to move a display cursor, the operator rolls the device over a surface, picks it up, and rolls it again. This action causes the cursor to move rapidly across the screen, independent of the surface area

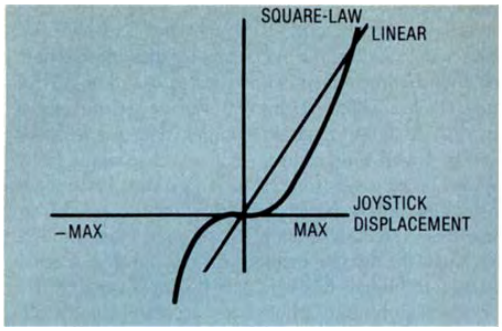


Figure 16. Relationship between joystick displacement and velocities.

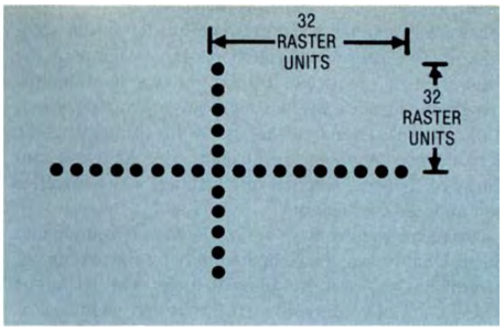


Figure 17. A tracking cross.

available for the mouse to roll on. Thus, the relative locator is not limited by physical space. The absolute locator, on the other hand, has a more permanent hand-eye relationship. A relative locator can be simulated with an absolute locator; the converse is not true.

We begin our description of specific positioning techniques with the *continuous translation* category:

Continuous indirect translation with locator devices. A locator is a device whose position is used to indicate a location on the screen. The movement of a locator device is directly mapped into the movement of a cursor or a displayed object. (We use the terms "displayed object" or "graphic object" to subsume the case of a cursor.) Examples of prerequisite devices are the tablet, the mouse, the trackball, the joystick, and cursor control keys with auto-repeat. Also required is a refresh display device.

Locators have dimensionality. Depending on their design, locators give positions in one, two, or three dimensions. Some can even locate positions in up to six or seven dimensions.

Continuous indirect translation with velocity-controlled joystick. A return-to-zero joystick controls the velocity of the graphic object, which is continuously repositioned. The position at the time t is given by a starting position plus the integral, up to time t, of the velocity values entered from the joystick. Zero displacement of the joystick corresponds to zero velocity. The prerequisite device is a positioning device with automatic (force) return to zero. The spring-return joystick and isometric joystick are members of this class. Only joysticks with rotatable center shafts can be used for 3-D positioning tasks; otherwise, joysticks are limited to 2-D.

Any initial position can be used. Joystick movement causes the cursor to move. When the user releases the joystick, the cursor stops moving. The relationship between the joystick displacement and the velocities along the x and y axes is usually linear, readily allowing full-screen movement at maximum velocity in a few seconds. Figure 16 shows such a relationship, as well as the relationship when the control has a signed quadratic form. The lower sensitivity in the low displacement range helps fine motor control.

Continuous indirect translation with up-down-left-right keys. The location of a graphic object is controlled by using step keys to move the object up, down, left, and right for 2-D applications, plus in or out for 3-D. The user indicates the desired location of the display cursor by depressing a set of keys assigned to control the cursor movements. A continual key depression causes the cursor to move in a rapid continuous motion, while a quick key depression causes the cursor to move a distance of one unit of display resolution. The user can achieve rapid positioning by keeping the key down, thus allowing the cursor's speed to accelerate. When the key is released, the cursor stops.

The prerequisite device can be a set of four or six keys on an alphanumeric keyboard, a programmed function keyboard, or other special key-input devices.

Continuous direct translation with light pen tracking. Tracking is performed with a small cross (called the tracking cross), which is pointed at with a light pen. As the pen moves to a new position, the tracking cross follows. The prerequistite devices are a light pen and a vector refresh display. Figure 17 shows a 64×64 display unit tracking cross. To start tracking, the user points at the cross with the light pen. As the pen moves, the cross follows the motions of the pen. If tracking is "lost" because the pen is moved too fast and the tracking algorithm of the system cannot follow the pen's motion, the user can resume tracking by moving the pen back to the cross. (The use of a light pen for positioning on a *raster* display is described as "direct, with locator device.")

Continuous direct translation with continuous search for light pen. A light pen is pointed at the screen. A raster scan of the screen is used to find the pen's position. Continuous search for the light pen is a variation of the tracking technique. To find the position of the light pen, a raster scan is displayed for each refresh cycle. When a displayed entity in the raster scan is "seen" by the pen, the position of the pen is known. In an important variation, the raster scan is limited to a nondistracting single frame and is displayed only when the user depresses a switch on the pen. The prerequisite devices are a light pen and a vector or raster refresh display.

The other category of positioning techniques is discrete translation. We discuss the major technique:

Discrete translation by position type-in. To indicate a position on the display screen, the user types in the coordinates of the location via an alphanumeric keyboard. A cursor appears at the user-specified location to provide visual feedback. Unlike a continuous translation technique, which continuously updates the position of the display cursor, a discrete translation type-in technique changes the location of the cursor only when a new position is entered. One disadvantage of this technique is the lack of continuous visual feedback to give the user a continuity of changes. Another disadvantage is the cognitive load imposed on the user.

This technique can have arbitrarily high resolution and dimensionality, which are limited only by the amount of information the user is willing to type. The technique preserves tactile continuity if the action language is keyboard-oriented.

Studies and evaluation of positioning techniques. The characteristics of these techniques are summarized in Table 3. Notice that by scaling up the displayed image, any of the techniques can be used to achieve any desired resolution. On the other hand, creating the scaled-up image is slow on some hardware configurations (but essentially instantaneous on others), so resolution can be a concern.

The choice of a continuous or a discrete feedback technique, based on whether the user knows where the position is on the display or knows the coordinates of the position, is critical. The wrong choice means a heavy cognitive load in converting from a spatial representation to a linguistic text-string representation, or vice versa.

The direct-versus-indirect positioning-technique choice hinges on questions of arm fatigue and hand-eye coordination. Direct techniques minimize all but motor effort, which is high because the arm must normally be raised to the screen. Indirect techniques require more learning of hand-eye coordination.

Fatigue in direct methods has been much discussed and is commonly assumed to be a problem. However, there are numerous anecdotal reports of drafters and designers using light pens for hours without problems. We have found no germane experiments.

Learning hand-eye coordination for indirect methods, the other common concern, is really not a major issue. Card et al.[10] studied the mouse and joystick for selection, and found improvement with repeated use in both error rate and selection time. However, even the performance of novices was quite good. Positioning time for a mouse decreased with practice from 2.2 to 1.3 seconds. For a joystick it decreased from 2.2 to 1.7 seconds. For discrete positioning (arrow key control) the time decreased from more than 3 seconds to about 2.2 seconds. Note, though, that when the discrete method was compared to a direct selection technique (the light pen) by English et al.,[12] the pen was slightly faster (though less accurate) for novices. Again, the difference was small.

There are no "pure" positioning experiments; the above-mentioned experiments all concern selection. We believe the results, which are discussed in detail in the selection subsection, can be generalized to the positioning task.

Orienting techniques. The orienting task involves specifying an orientation (not a position) in a coordinate system. The requirements of the task, again determined by the application, specify the degrees of freedom, the type of feedback (closed-loop or open-loop), and the resolution desired.

Interaction devices needed, depending on task requirements, are the quantifiers, the locators used for positioning, and the alphanumeric keyboard.

As with selection and positioning, a number of parameters have an impact on the detailed design of an orienting technique:

(1) *Center of rotation.* The center of rotation of an object might be the origin of world coordinates, the center of the object, or any arbitrary, user-specified position. In any case, the user must know where the center of rotation is located. Intuitively, the most convenient center of rotation is the center of the object being oriented. If the object

Table 3.
Comparison of positioning techniques.

	Ergonomic Measures								
Techniques	Cognitive Load	Percep-tual Load	Motor Load	Visual Acqui-sition	Motor Acqui-sition	Ease of Learning	Fatigue	Error Proneness	Feedback Type
Direct, touch panel	L	L	L	L	L	L	L	L	D
Indirect, tablet	M	M	M	M	M	M	L	L	C
Indirect, mouse	M	M	M	M	M	M	L	M	C
Indirect, absolute joystick	M	M	M	M	M	M	L	M	C
Indirect, velocity-controlled joystick	M	M	M	M	M	M	L	H	C
Indirect, track-ball	M	M	M	L	M	M	L	M	C
Indirect, cursor control keys & auto-repeat	H	H	M	H	M	M	L	M	C
Indirect, up-down-left-right arrow keys	H	H	M	H	M	M	L	M	C

C = continuous, D = discrete

spans the entire screen, then the center of the screen becomes the center of rotation.

(2) *Visual aids.* Especially in 3-D orientation tasks, the user often has difficulty knowing exactly how the displayed object is currently oriented. Display of a gnomon can help—a commonly used gnomon is just a set of axes on which the positive and negative *x, y,* and *z* axes are labeled. The axes are displayed with the same orientation as the object of interest.

(3) *Coordinate systems.* The previous discussion of this issue with respect to positioning devices applies also to orientation.

Like positioning techniques, orientation techniques are either continuous or discrete:

Indirect continuous orientation with locator devices. To rotate a 2-D object, the user specifies the angle of orientation by using a continuous quantifier or one axis of a physical locator. Typical devices used are the tablet, the absolute-controlled 3-D joystick, and dials. The device is read continuously and its value mapped to the new orientation of the displayed object. There is a one-to-one correspondence between the movement of the hand-held device and the movement of the object on the screen. The definite advantage of the technique is the preservation of hand-eye coordination. Of course, use of a locator means that linear movement of the hand is converted to rotational movement.

For 3-D applications, quantifiers or locators can be used to rotate a displayed object by specifying roll-pitch-yaw or direction cosines.

Indirect continuous orientation with velocity-controlled joystick. The velocity of orientation change is controlled by the use of a return-to-zero joystick. Zero displacement of the joystick corresponds to zero velocity and hence to no change in orientation. The displayed object is continuously rotated. The technique is very similar to the direct-controlled techniques for continuous translation, except that in this case the angle of orientation, rather than the distance of translation, is velocity controlled on the basis of the values from the device. The relation between angular displacement(s) and the input value(s) is analogous to that for velocity-controlled positioning. Moving the device rotates the object. When the device is released, the object stops rotating.

Unlike the direct-controlled technique, this technique does not preserve hand-eye coordination. Its real advantage is that a large change of orientation can be obtained by small hand movements.

The prerequisite device is any positioning device with automatic return to zero, such as the spring-return and the isometric joysticks.

Discrete orientation by angle type-in. The user types in values to define an orientation at an alphanumeric keyboard, either as angles, direction cosines, roll-pitch-yaw, or some other form. The new orientation is shown. If it is wrong, the type-in must be repeated. However, if the action language is keyboard-oriented, the technique preserves tactile continuity.

The technique involves implicit or explicit specification of the center of rotation. It can be the center of the object or of any arbitrary, user-specified position. In the latter case, the center of rotation would be specified by means of a positioning technique.

Evaluation of orienting techniques. Characteristics of the techniques are given in Table 4. All of these techniques are indirect—there is no practical analog to the touch panel for orientation. (Herot and Weinzapfel[72] describe an experimental direct orientation device.) Thus the fatigue question does not arise. The hand-eye coordination issue is germane, but no data are available. Naturally, clockwise hand movement should result in clockwise rotation of an object on the screen. Conversely, in a simulation environment, a clockwise hand movement would cause the screen display to rotate counterclockwise. The continuous-versus-discrete-feedback issue, discussed in regard to positioning, is equally relevant to orienting.

Pathing techniques. The pathing task involves specification of a series of locations or orientations, evolving in time and space. Pathing is always continuous and closed loop. Requirements that must be specified for pathing are the maximum number of positions or orientations along the path, the interval of sampling and whether it is time based or distance based, the dimensionality, the resolution of individual samples, and whether the path is a positioning or an orienting path, or both.

Any positioning or orienting device can be used for pathing, provided only that the system is capable of supporting closed-loop operation. Furthermore, pathing techniques involve all of the same issues as the corresponding positioning and orienting techniques.

In addition, pathing involves interval selection. When the sampling interval is distance based, the smoothness of the path echo (see next paragraph) is controlled. Each sample position can be a vertex on an echo, and when ver-

Table 4.
Comparison of orienting techniques.

	Ergonomic Measures								
Techniques	Cognitive Load	Perceptual Load	Motor Load	Visual Acquisition	Motor Acquisition	Ease of Learning	Fatigue	Error Proneness	Feedback Type
Indirect, joystick (absolute)	L	L	L	L	L	L	L	L	C
Indirect, joystick (velocity-controlled)	M	L	L	L	L	L	L	M	C
Numeric character strings				[See Text Entry]					

tices are separated by a uniform distance, the visual effect is a uniform smoothness over the duration of the historical path. When sampling is time based, samples will be separated by longer distances when the path is changing rapidly and shorter distances when it is changing slowly. The effect on a line-drawn echo is greater smoothness and faithfulness to shape when the device is moved slowly and, presumably, with more deliberation. When the echo is a scene change, time-based sampling produces results close to the effect of a motion picture. Thus, the choice between distance sampling and time sampling depends on whether smoothness and uniform faithfulness of the historical path or the perception of smooth motion of the echo image is more important.

Another pathing issue is echo form. By "echo" we mean the visual manifestation of the variation of position or orientation. The echo may be a continuously wavy line that lays down the history of positions followed by the path. The echo may be only a cursor or gnomon indicating the current endpoint or end orientation of the path, or it may be a modification of the viewing or image transformation of some displayed object. In the latter case the effect is one of looking through the window of a vehicle whose path is being controlled, or of "flying" an object on the screen under the control of a device.

A final issue is that techniques differ according to how the echo is smoothed. An echo that lays down the history of the path as an image can join the simple vertices by straight lines, not join them at all (leaving a series of dots), or join them by one or another spline technique. (A spline is a continuous curve with continuous first, and sometimes second, derivations. That is, the curve has no cusps or fold points.) The orientation of an image can be similarly smoothed to give the effect of continuous motion even when the samples are widely separated in time.

Studies and evaluation of pathing. Because pathing differs from positioning and orienting only in the nature of the data returned and not in performance characteristics, our earlier discussions of studies and evaluations of those techniques are appropriate to pathing as well.

Intuition suggests that good hand-eye coordination is more important in pathing than in positioning or orienting. Furthermore, pathing relies almost exclusively on continuous-feedback techniques, even though occasionally an application may deal with the selection of a preplotted (discrete) path.

Experimental data for pathing are similarly dependent on experiments testing positioning. The experiments by Irving et al.[16] compared devices in the special context of pathing, but they were inconclusive. In our own experience, most computer artists find a tablet with pen stylus preferable to a mouse for freehand sketching. The stylus has the shape, mass, and feel of a pen and hence is easy to use for sketching either large, sweeping shapes or finely detailed ones.

Quantifying techniques. The quantifying task involves specifying a value or number within a specified range of numbers. Several parameters are germane to all quantifying techniques:

(1) *Range specification.* It is important that the range of the number to be specified is reasonably chosen. Too large a range, for many techniques, can limit useful resolution and cause the control/display ratio to be excessively high. Unbounded techniques may not have this problem if the C/D ratio is well chosen.

(2) *Control/display ratio.* As with positioning, the amount of physical movement corresponding to any change of the selected number is critical to performance. Again, low ratios are good for fast change, while high ratios are good for fine adjustment accuracy. Several techniques listed below exhibit a variable ratio, initially favoring a low ratio and finishing the action with a high ratio.

(3) *Echo form.* Whether the user is provided a scale or a number to indicate the value being quantified can be critical to performance.

Quantifying techniques can be continuous or discrete:

Continuous quantifying with physical devices. Quantifying can be directly accomplished by use of a slide (linear potentiometer), a dial (rotational potentiometer), or a strain gauge device. (Locating mechanisms are also generally applicable, but they appear not to be commercially available—e.g., tablet, resistive surface, strain, servo, etc.) In the continuous mode, one may use a bounded dial, an unbounded dial, or a slide. A bounded dial is similar to the volume contol on a radio—turning the dial far enough in one direction, one reaches a stop, which prevents further turning. This type of device indicates an absolute quantity.

An unbounded dial resembles the bounded dial except that there are no stops. One can turn the dial an unbounded distance in either direction. An unbounded dial specifies a relative quantity. The provision of some sort of echo enables the user to determine what value is currently being specified. Turning the dial in one direction increments the quantity; turning the dial in the other direction decrements the quantity.

Slide potentiometers allow the user to tell at a glance the approximate setting. Rotary dials, even with pointers, are not as good for this. The choice between rotary and linear devices should also take account of whether the visual effect (i.e., feedback) is rotational (a turning clock hand) or linear (a rising temperature gauge).

Continuous quantifying by scale drag. With a selection device, the user points to the current-value indicator on a displayed gauge or scale and then moves along the scale to the desired value. He performs an action, such as clicking on a mouse button or depressing the tablet stylus, to indicate selection of the desired value. A highlighted line or pointer may be used to indicate the length selected on the scale, or a numeric echo may appear (preferably at a standard position on the screen).

Continuous quantifying by locator value. The screen is initialized with a locator on a scale. The user moves the locator along the *x*-axis or the *y*-axis, causing the current coordinate to position a pointer on the scale. (This movement may or may not be proportional to the movement of the pointer on the scale; however, the movements should be coordinated.) The position of the cursor indicates the value to be specified. A digital value may also be displayed. Note that this technique may be used like a scale drag. A quantity ranging from the initial indicator position to the current indicator position can be specified. The range of the quantifier can be extended by treating the tablet as a relative locator, such as Thornton's number wheel.[73]

Continuous quantifying by dial with echo. The user turns a dial to increment or decrement a digital echo. On the screen (preferably at a standard position), a digital representation of the quantity changes. This change of the represented value may correspond to either a large or a small amount of turn of the dial, depending on whether or not fine tuning of the number is desired. Some method of fine tuning may be included. Alternatively, the digital readout may be on an LED or LCD panel immediately next to the dial.

Continuous quantifying by dial with scale. The refresh display screen shows a scale and a cursor or other indicator. As the user turns a dial, the cursor or other form of pointer moves along the scale. A method of controlling the velocity of the indicator may be included.

Continuous quantifying by light handle. A tracking cross or cursor is moved by a light pen or other continuous positioning device in a work area composed or a least two adjacent rectangles. A displayed value is associated with the location of the tracking cross in the work area. Upward movements of the cross cause the value to increase; downward, decrease. Movements in the left part of the work area cause larger changes than movements in the right part. Horizontal movements have no effect on the value. All vertical movements, except those in the right-hand part, cause changes proportional to the square of the change in *y*. In the right-hand part, the number of changes are proportional to the change in *y*, so that a fine tuning mechanism exists. See Newman[55] for a detailed discussion.

Continuous quantifying by locator with ratchet value. In a scale-drag (or locator-value) setting, cursor movements that would normally cause the displayed value to decrease instead cause the scale to shift. The user points with a continuous positioning device to the least-valued end of the scale and then moves along the scale, causing the displayed value to increase. When the user moves back along the scale, instead of the value decreasing, the whole scale shifts to allow larger values to be specified.

As in the scale-drag method, some action, either explicit or implicit (for example, entering a new command), must signal the selection of a value.

Continuous quantifying by simulated stopwatch. The user pushes a button, which activates a "digital watch" effect: a number is displayed and begins to change at a constant rate. This rate optionally may be regulated by a dial; turning the dial causes the rate to either accelerate or decelerate. Before reaching the desired value, the user can cause smaller changes in the number with quick jabs of the button. When the desired value is displayed, he releases the button altogether and by another action—for example, pressing another button—signals the designation of the displayed number. A backup button can also be provided, to be used if the desired value is initially passed over.

Discrete quantifying by type-in. The user types in the desired quantity at the keyboard.

The characteristics of these techniques are summarized in Table 5. We are aware of no experiments relating to quantifying tasks or techniques.

Text entry techniques. Text entry involves expressing information in strings or blocks of characters selected from a character set predefined for the discourse. Text is entered from a keyboard as described below.

An important distinction must be noted between text entry and selection. In text each character individually causes no action, but collectively in the string the characters act as a single entity. Therefore, each key has an unchanging meaning, regardless of the situation. In selection, however, each key can cause an action, and the meaning of the key can change depending on the situation in which the key is depressed.

Notice that we do not specify what the keyboard looks like, how many keys it has, or what types of actions are initiated by strings of keystrokes. In fact, keyboards other than alphanumeric keyboards are possible. For example, one might use a steno keyboard, which enters strings of syllables, or a piano keyboard, in which meaning is attached to chords. In any keyboard, however, some special convention is needed to signal the end of the text. On an alphanumeric keyboard a "return" key usually serves this purpose.

Table 5.
Comparison of quantifying techniques.

	Ergonomic Measures								
Techniques	Cognitive Load	Perceptual Load	Motor Load	Visual Acquisition	Motor Acquisition	Ease of Learning	Fatigue	Error Proneness	Feedback Type
Direct, rotary potentiometer	M	L	M	M	M	L	L	L	C
Direct, linear potentiometer	M	L	M	M	M	L	L	L	C
Numeric character string				[See Text Entry]					
Scale drag, one axis of tablet	M	M	M	M	M	L	M	L	C
Scale drag, one axis of mouse	M	M	M	M	M	L	M	M	C

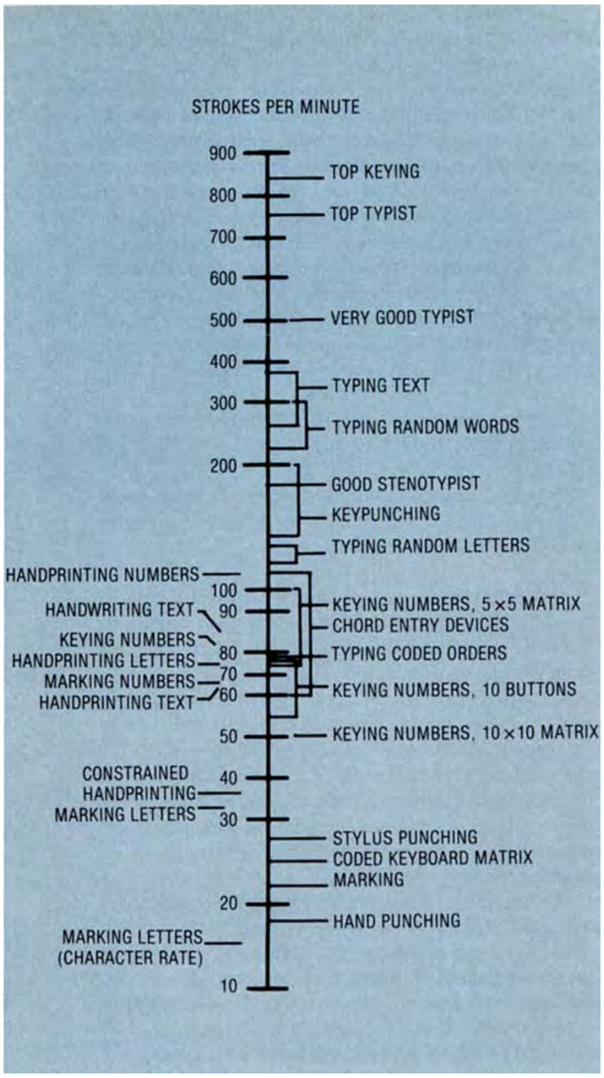


Figure 18. A comparison of entry rates (adapted from Devoe[76]).

Text entry by voice recognition. A keyboard can be implemented by means of a voice recognition device. Each letter is spoken, and separated by 0.1 to 0.2 seconds of silence. Commonly used words can also be recognized to speed up the input process. Some contemporary systems no longer require a pause between each word or letter. Systems that recognize an unlimited vocabulary, on the other hand, are still in the future.

Text entry by stroked character recognition. The user prints the text with a continuous positioning device, usually a tablet and stylus. The computer then breaks the text into strokes from which letters can be recognized. For instance, the capital letter "A" consists of three strokes—typically, two downward strokes and a horizontal stroke.

Character recognizers have been used with interactive graphics since the early 1960's (see Brown[74] and Teitelman,[75] for example). Program and data typically occupy less than four kilobytes. Several commercial microprocessor-based implementations exist. A simplified adaptation of Teitelman's recognizer, developed by Ledeen, is described by Newman.[33] The computer recognizes characters drawn with a locator by comparing characteristic features (slopes, points of inflection, quadrants crossed, etc.) against a stored dictionary of each character's features. A recognizer can be trained to identify different styles of block printing: the parameters of each feature are calculated from sample characters drawn by the user.

The rate at which we print is slow compared to typing or writing cursively. It is difficult to block print more than two or three characters per second (try it!), so recognition would not be appropriate for massive input of text. We can write cursively faster than we can print, but as yet there are no simple algorithms to recognize continuous script.

Text entry by menu selection. A series of letters, syllables, or other basic units is displayed as a menu. The user then inputs text by choosing letters from the menu with a selection device.

Evaluation of text entry techniques. For massive input of text, there is no substitute for a skilled typist working

Table 6.
Comparison of text entry techniques.

	Ergonomic Measures								
Techniques	Cognitive Load	Perceptual Load	Motor Load	Visual Acquisition	Motor Acquisition	Ease of Learning	Fatigue	Error Proneness	Feedback Type
Alphanumeric keyboard	H	H	M	M	M	M	M	M	D
Chord keyboard	H	L	M	M	M	L	M	H	D
Stroked character recognition	H	L	M	M	M	L	M	H	C
Voice recognition	L	L	L	L	L	H	L	H	D
Direct pick from menu, light pen	L	L	M	L	M	L	M	L	D
Direct pick from menu, locator device	L	L	L	L	L	L	L	L	D
Indirect pick from menu, locator device				[See Positioning]					

with a traditional keyboard, except automatic scanners. Figure 18, adapted from Devoe,[76] shows experimentally determined input rates for a variety of techniques. Speech input, not shown on the chart, is slow but especially attractive for applications where the hands need to be free for other purposes, such as handling paperwork.

The hunt-and-peck typist is limited by the perceptual task of finding a key and the ensuing motor task of moving to and striking it, while the trained typist has only the motor task of striking the key, preceded sometimes by a slight hand or finger movement to reach the key.

The characteristics of the techniques for text entry are summarized in Table 6. None of the techniques discussed pose any real limit on character set size, so long as Western alphabets are involved.

Controlling techniques

Fundamentally, each of the techniques we have described represents a task of choosing. Selection chooses from among a set of entities, positioning chooses a place in space, orientation chooses an angle in space, and quantification chooses a number. Similarly, pathing chooses a sequence of places or angles in space, while text entry chooses a sequence from among a special set of entities, the characters.

The controlling techniques carry out another set of fundamental tasks, but the purpose of these tasks is to form and transform visible objects, usually by a process of continuous modification. The controlling tasks and techniques are directed to an object that in some sense exists and is modified to satisfy a conception of what it ought to be.

Stretching techniques. A stretching technique involves taking a target object (a line, a triangle, a circle, a rectangle, or a prism) and distorting its shape by coercing one of its points to coincide with a specified position.

Because a positioning (of the movable point) is an intrinsic part of the task, all the richness of technique inherent in the positioning task (see Figure 2 again) is inherent in the variety of stretching techniques. In particular, stretching techniques can be classified according to whether they are performed with continuous or discrete feedback and whether they are direct or indirect. They can also be exercised in two or three dimensions.

Here we discuss the techniques for stretching independently of the choice of positioning technique used. In essence, stretching techniques differ from corresponding positioning techniques only in the choice of the form of the object being stretched and the manner of stretching. Generally, stretching techniques based on continuous-feedback positioning techniques are far more useful than those based on discrete feedback.

Stretched lines. The stretched, or "rubber-band," line is a stretching task that maintains a line extending from a reference point to a point specified by a positioning technique. As the latter point is moved, the line is modified to follow. The effect is like a rubber band being stretched between a fixed point and a moving cursor.

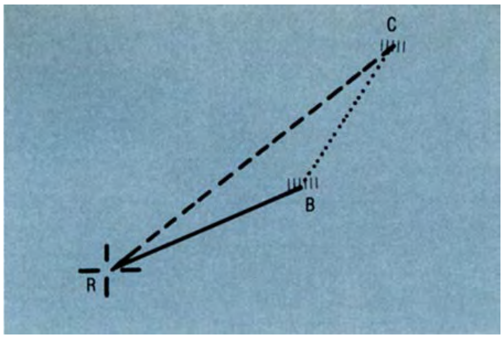


Figure 19. A rubber-band line.

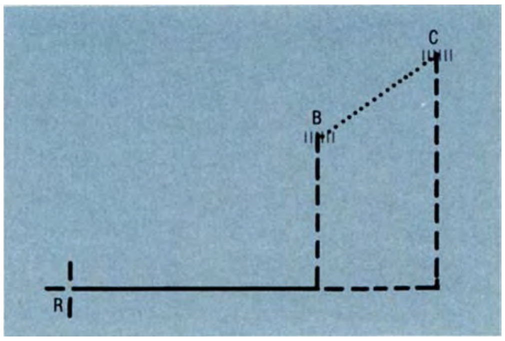


Figure 20. A stretched horizontal line.

The rubber-band technique, in its most basic form, makes use of an echo position (cursor) and a reference point. The object is to display a line from the reference point to another point on the screen. The user selects the reference point by pointing and then signaling acceptance of the cursor position (with positioning and selecting techniques, respectively). Further motion of the cursor from the reference point to the desired endpoint causes a line to be stretched from the reference point to the cursor (see Figure 19). Moving the cursor from B to C (along the dotted line) causes the displayed line to be displaced to the line from R to C.

The technique of rubber-banding is useful in building sketched forms, connecting lines in graphs, and in creating forms for further manipulation and analysis. Often, however, connecting lines made up of only horizontal or vertical segments are desired. The following techniques and variations can be used for this purpose:

Stretched horizontal or vertical lines. A line is stretched horizontally between a reference point and the x-coordinate of a point specified by a positioning technique and the y-coordinate of the reference point (Figure 20). Vertical stretching is analogous.

Variations on this technique include

- Combining constrained horizontal with constrained vertical may produce a system that acts like a constrained vertical if the angle between the vertical line

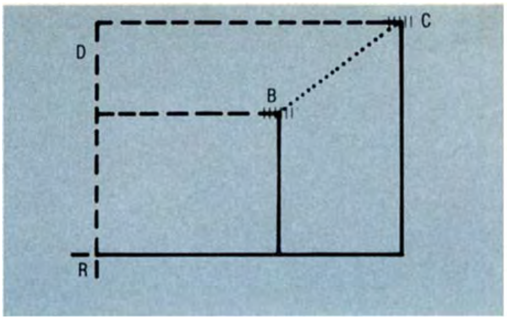


Figure 21. Displaying *x* and *y*.

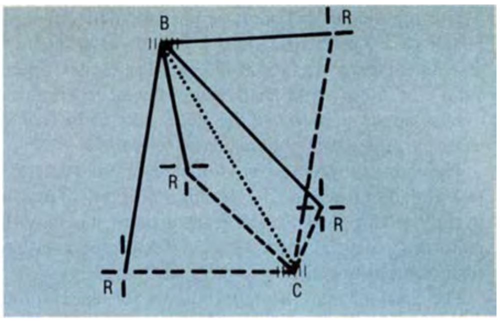


Figure 22. A rubber vertex.

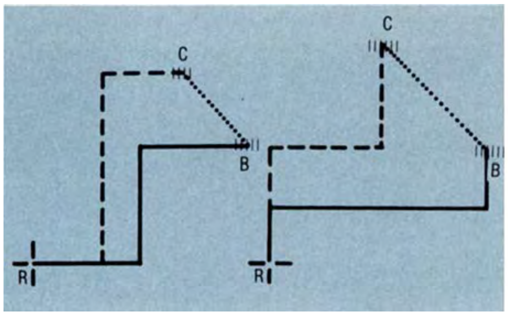


Figure 23. A zig-zag line (two alternatives).

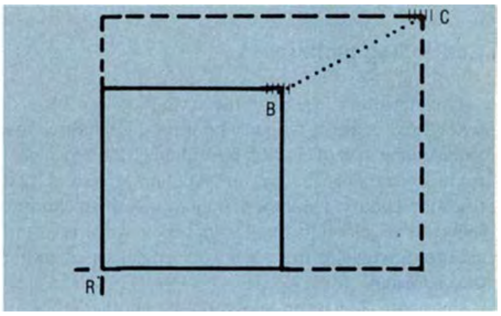


Figure 24. A rubber rectangle.

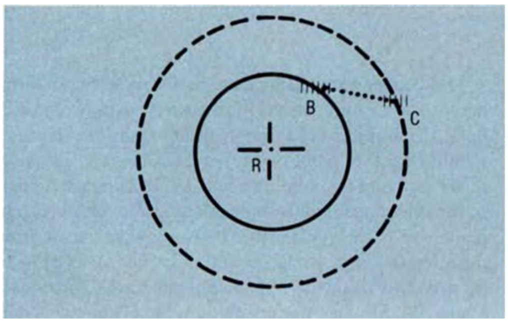


Figure 25. A rubber circle.

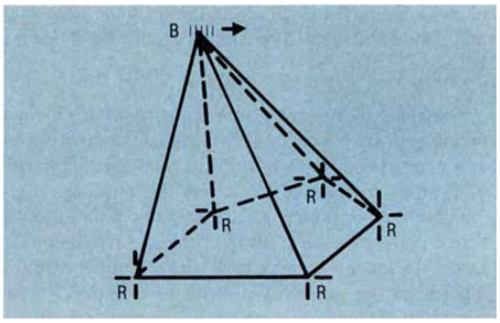


Figure 26. A rubber pyramid.

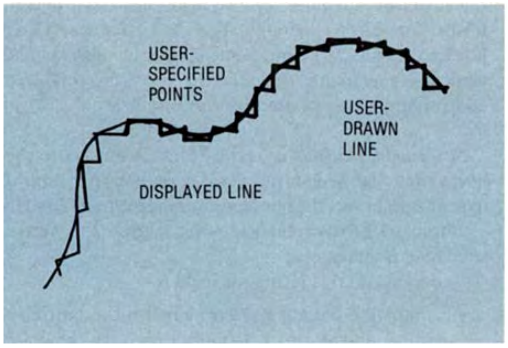


Figure 27. Sketching, with stair-step smoothing.

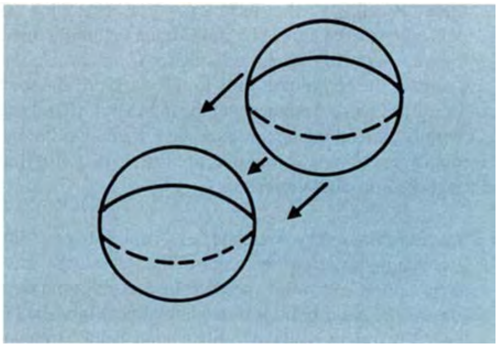


Figure 28. Dragging a sphere.

through the reference point and the line through the reference point and the cursor is less than 45 degrees.

- Displaying the x and y components of the line between the reference point and the cursor (Figure 21).

Rubber-band vertex. A set of lines is drawn from a set of reference points (not necessarily coplanar) to a single point specified by a positioning technique. In other words, a number of rubber-banded lines are drawn from the reference points to a common cursor (Figure 22).

Zig-zag. A zig-zag displays one of two possible paths: either horizontally leading or vertically leading from the reference point (Figure 23). Either or both of these paths may be available, depending on the system.

Other figures can be stretched in a manner similar to the stretching of a line. A few are enumerated below, and are simply generalizations of the idea of rubber-band lines and its variants:

Rubber rectangles. A rectangle is stretched so that one of its corners is at the reference point, and the diagonally opposite corner is at a point specified by a positioning technique (Figure 24).

Rubber circles. A circle is expanded with its center at the reference point, so that a point specified by a positioning technique is on its periphery (Figure 25).

Rubber rectangles and rubber circles can be generalized to regular figures with an arbitrary number of sides. They can also be generalized to three dimensions (rubber parallelopiped and rubber sphere).

Rubber pyramids. A three-dimensional rubber vertex is drawn, with reference points connected to form a closed base (Figure 26). Note that this closed figure need not lie in one plane. Many other variations of these techniques are possible.

Sketching techniques. The sketching task involves specification of a curved line in two or three dimensions. The user specifies a starting position, a path, and its end. The requirements of this task, as determined by the application, are dimensionality, resolution, sampling criterion, and smoothing method. Since the technique has a continuous-feedback positioning task embedded in it, all requirements associated with positioning are applicable, and, in fact, any positioning technique that satisfies the positioning requirements of the task can be used. Reference should be made to the positioning requirements and issues discussed earlier.

The sketching task is also similar to the pathing task. It differs from the pathing task in that its whole purpose is to create a curve in space, whereas the pathing task is primarily concerned with a temporal evolution of position or orientation. Nevertheless, the requirements and issues of pathing, particularly sampling criterion and smoothing method, are relevant.

In sketching, the choice of whether to apply a time sampling or a space sampling is a requirement of the task, since the user will have in mind the production of a shape either with a desired degree of granularity or with special faithfulness to those portions of the path that were drawn most carefully. In the former case, space sampling is preferable; in the latter, time. It is also possible to combine the criteria, either in a weighted manner or according to a priority order (sampling first by the criterion met first).

The manner of approximation is also a requirement of the task. When general smoothness is desired, spline and piecewise polynomial approximations of degree greater than 1 are preferable to linear methods, even at the expense of considerable heightening of the computation requirements of the machine. Piecewise linear (line-segment) approximations of the curve are faceted at the approximation points, but they are computationally simple and more readily implemented directly in hardware. Hence the linear approach is more likely to be acceptably responsive in a rapidly interactive application. The manner of approximation includes, also, a specification of whether it is an exact matching or a smoothed approximation. In a smoothed approximation, the sampling points are used as control points. One might specify use of B-spline, Bézier, or other styles of interpretation for the control points.

It should be observed that the task of sketching can also be served by a technique of placing a simple curve on the screen and then shaping the curve to one's liking, using the techniques described later, for shaping. Variants would use a 3-D joystick rather than a stylus.

Sketching with stylus or pen. The user draws a freehand curve that continuously follows the path of a pen or stylus-cursor. The position is sampled at time intervals or distances as specified. A continuous curve is updated on the screen to include each sampled position, until a signal (button depression) is given to indicate termination of the curve. (The curve remains or disappears according to the purpose of the sketching.) The curve is usually made of line segments joining the sample locations.

Variations include a "stair-step" technique, by which the system connects the appropriately spaced points by displaying the x and y components of the distances between the samples (Figure 27).

Shaping a line. The user causes a line to be stretched between specified endpoints, and then the "adjustable curve" technique is applied to shape the curve to the desired form.

Manipulating techniques. "Manipulating" refers to operations performed on a displayed object whereby the form of the object remains unchanged, but position and orientation are changed. The requirements are generally the same as for the positioning and orientation tasks, and the reader is referred to those earlier sections.

Dragging. A drag occurs when the user picks or locates an object on the screen (e.g., a circle or a cube) and moves it to a new location on the screen so that a reference point on the object coincides with a point specified by a positioning technique. For instance, suppose a sphere is located on the left side of a screen. The user drags the sphere to the right side by moving a locator (or pick) to the sphere and then moving the locator across the screen to the new location (Figure 28). The movement of the object is normally continuous during the dragging. Typical devices are a stylus or pen. Variations are derived from the use of other positioning techniques and devices.

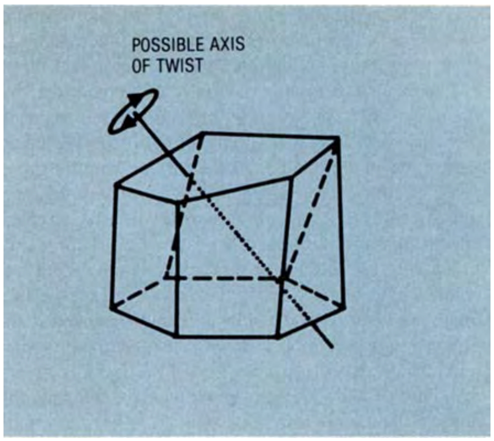


Figure 29. Twisting an object (hidden lines shown dashed).

Twisting. Twisting occurs when a displayed object is caused to rotate about an axis. This process is analogous to turning an object around in one's hand (Figure 29.) The user chooses the axis by picking a line or by performing a position and orientation. The degree of twist is specified by a quantifying technique. The movement is normally continuous. The prerequisite is a quantifying device, such as a dial.

Scaling. "Scale" refers to how large the displayed object appears on the screen. By manipulating the scale, one can cause the object to appear larger or smaller on the display (independently of other displayed objects). The scale is specified by a quantifying technique, such as the use of a dial.

Shaping techniques. The shaping task molds a curve or surface until it reaches some desired shape. In interactive graphics systems, shaping is highly dependent on how lines and surfaces are represented inside the system. Shapes can be represented by control points, which exert an influence on different parts or on the whole shape. In other words, given the control points, one can specify a particular shape. Two common shape representations using control points are the Bézier method and the spline method. In the Bézier method, the control points exert influence on a particular blending function. This blending function, coupled with the control points, defines the shape. (The control points do not necessarily lie on the shape.)

Adjustable curves. Complex curved lines in two or three dimensions can be represented and displayed by any of a number of representation techniques. One usually wants a curve that lacks corners or cusps. Therefore, splines and Bézier functions are usually used.

In splines the curve is represented by control points that lie on the curve, and the curve is constructed by means of piecewise polynomial representations (usually cubics which maintain continuous first derivatives between the pieces). With Bézier representations the control points are generally external to the curve but nevertheless control the shape.

The most common technique for shaping or reshaping a curve is called "flexing"—that is, dragging the control points with a stylus or pen. For further discussion of approximation (smoothing) methods, see the Newman and Sproull[33] and the Foley and Van Dam[77] textbooks.

Adjustable surfaces. The shaping of surfaces, as with curved lines, is highly dependent on the way surfaces are represented in the system. Basically, we can extend the two methods described for curves to surfaces by taking the Cartesian product of two curves representing the cross sections of the surface. We draw the surface itself by keeping the parameterizing variable in one cross section equation constant and varying the parameterizing variable in the other cross section equation. The process is then repeated, keeping the parameterizing variable of the second equation constant. With the Bézier method, we maintain continuity at boundaries of Bézier surfaces by ensuring that the common control points and the adjacent control points are all colinear across cross sections, and that the ratio of the distances of the common control points to the adjacent control points is constant across cross sections. Similarly, we can form B-spline surfaces by taking the Cartesian product of two B-spline curves.[33,77]

As with curved lines, the most common technique for forming or reshaping surfaces is to use a locator to select and drag displayed control points to a new position, causing the surface representation to change shape.

Conclusion

In this research work we have proposed an organization of interaction techniques based on the user tasks for which the techniques are used. We have suggested that task requirements limit the set of techniques that can be considered for a particular application. We have enumerated the characteristics of a variety of techniques and discussed many of the considerations important to their effectiveness. We have tabulated relevant experimental and experiential comparisons.

Now, how does one select from the set of feasible techniques? In some cases experiments or experience can help. In general the perceptual, cognitive, and motor loads of each technique can be considered. However, we have not quantified these loads. We have suggested a diagramming method,[19] which can assist in such quantification.

Interaction techniques cannot be selected in a vacuum. The context in which they are used is crucial and is not accounted for by experiments. Experience, however, shows the importance of perceptual, cognitive, and motor continuity from one interaction task to the next, particularly (but not exclusively) within a single sentence.[5]

Future research. Research directions are best set in the context of a long-range goal. Our goal is a model of user-computer interaction that can predict the performance of both new and skilled users of various interaction techniques. While this goal may never be completely achievable in practice, it can nevertheless act as the motivator for research.

The first step in developing such a model is the identification of basic interaction tasks. We have suggested one set, but there are certainly others. Our tasks do not very well account for the substantial differences between positioning by coordinate pair type-in and by explicit pointing at a location on the display surface, nor between operand name type-in and explicit pointing at the displayed operand. In both cases, the wrong type of technique can force the user into a cognitive process of converting from one representation to another. Perhaps this is just another example of the cognitive analog to motor and perceptual continuity from one step in the user-computer dialogue to the next. Whatever the case, more work is needed in this area.

Next, we need to characterize interaction techniques in such a way that their perceptual, cognitive, and motor components are identified. The interaction technique diagrams mentioned above are our starting attempts in this direction. We found them essential in precisely defining the various experiments. Some formalism such as these charts can be crucial.

The next step is to continue where the diagrams leave off—tying together a series of interaction tasks as would be done in a real user-computer dialogue, representing the various types of discontinuities we have discussed, and also representing the characteristics of sentences in the interaction language.

A most difficult step is to quantify the perceptual, cognitive, and motor processes in terms of the time they take. Many factors must be considered—in menu selection, for example, we must account for the size of the menu, where it is positioned, and the symbolic representation used (i.e., text vs. iconic). The starting model is the work reported by Card et al.,[31,32] in which the motor and cognitive components of a few selection, positioning, and text entry techniques have been successfully quantified for skilled, well-trained users performing routine tasks with keyboard and mouse. The next steps would be to extend their "keystroke-level model of user interaction" to include additional techniques and devices, and to account for the perceptual processes of visually acquiring a menu or cursor.

A final, but perhaps concurrent, step is to quantify the other criteria for interactive performance (learning time, recall time, memory load, error susceptiblity, and naturalness) and relate them to the work factors (perceptual, cognitive, and motor).

Perhaps, also, this catalog of techniques should be augmented to include all known useful techniques.

What is the role of experiments in all this? The types of experiments we have described, while useful, provide very limited and specialized knowledge, which is hard to generalize for use in new situations. The proper role of experiments should be to

- obtain basic performance data to be used in an overall model(s) of user-computer dialogues, and
- verify the dialogue model(s).

The model would be the tool we need for quantitative evaluation of individual interaction techniques and sequences of interaction techniques. ■

Acknowledgments

We are grateful to many people for providing advice, information, and sources. We are particularly grateful to Terry Jennings, former student assistant at the University of Kansas, and John L. Burch of Ergosyst Associates, Inc. They both worked with us throughout the project, wrote portions of the draft, engaged with us in many fruitful discussions and critiques, and contributed freely their ideas and knowledge. We are indebted to Franklin Moses of the US Army Research Institute for the Behavioral and Social Sciences for his support, encouragement, expert criticism, helpfulness, and patience. Guest Editor Jack Grimes and the *IEEE Computer Graphics and Applications* referees also provided helpful comments.

This work was supported in part by the US Army Research Institute for Behavioral and Social Sciences under contract MDA-903-79-G-01.

References

1. J. L. Bennett, "User-Oriented Graphics Systems for Decision Support in Unstructured Tasks," *Workshop on User-Oriented Design of Interactive Graphics Systems*, ACM, Pittsburgh, Pa., Oct. 1976, pp. 3-13.

2. E. G. Britton, "A Methodology for the Ergonomic Design of Interactive Computer Graphics Systems," PhD dissertation, Dept. of Computer Science, Univ. of North Carolina, Chapel Hill, N.C., 1977.

3. D. R. Cheriton, "Man-Machine Interface Design for Timesharing Systems," *Proc. ACM 1976 Conf.*, pp. 362-366.
Presents a design approach and design criteria for the man-machine interface in timesharing systems. Outlines a conceptual view of timesharing systems, focusing on the interface between the user and the capabilities of the system. Considers user needs and requirements, and suggests design guidelines and approaches to meet these needs.

4. S. E. Engel and R. Granda, *Guidelines for Manual Display Interfaces*, IBM tech. report, TR00 2720, 1975.
Documents a set of human factors guidelines relating to the interface between a user of an interactive computing system and a display terminal connected to the system. Though intended primarily for software developers of interactive systems, many of the guidelines should be of interest to hardware developers. Areas covered include display frame layout, frame content, command languages, error prevention and recovery, response times, and behavioral principles.

5. J. D. Foley and V. L. Wallace, "The Art of Graphic Man-Machine Conversation," *Proc. IEEE*, Apr. 1974, pp. 462-471.

6. W. J. Hansen, "User Engineering Principles for Interactive Systems," *AFIPS Conf. Proc.*, Vol. 39, 1971 FJCC, AFIPS Press, Montvale, N.J., pp. 523-532.

7. D. Smith, C. Irby, R. Kimball, W. Verplank, and E. Harslem, "Designing the Star User Interface," *Byte*, Vol. 7, No. 4, Apr. 1982, pp. 242-282.

8. V. L. Wallace, "Conversational Ergonomics," *Workshop on User Oriented Design of Interactive Graphics Systems*, ACM, Pittsburgh, Oct. 1976.

9. H. R. Ramsey and M. E. Atwood, "Human Factors in Computer Systems: A Review of the Literature," tech. report SAI-79-111-DEN, Science Applications, Inc., Sept. 21, 1979, 169 pp. Available from NTIS as AD-A075-679.

10. S. K. Card, W. K. English, and B. J. Burr, "Evaluation of Mouse, Rate-Controlled Isometric Joystick, Step Keys, and Text Keys for Text Selection of a CRT," *Ergonomics*, Vol. 21, No. 8, Aug. 1978, pp. 601-613.
Experiment evaluates different devices for selection in a text-editing environment. Both the step keys and the text keys are cursor-control keys. The step keys move the cursor up, down, left, right, or to the top upper left corner of the display; the text keys move the cursor in units of paragraph, line, word, and character. Results show the mouse to be the fastest and the most accurate device, followed by the joystick, the text keys, and the step keys. The closed-loop feedback devices (the mouse and the joystick) are found to be better than the open-loop devices (the step keys and the text keys) for selection. The experiment was well conducted, and the tasks the subjects performed were well monitored and described. One can generalize these device ratings to a non-text-editing environment.

11. W. K. Earl and J. D. Goff, "Comparison of Two Data Entry Methods," *Perceptual and Motor Skills*, Vol. 20, 1965, pp. 369-384.
Experiment compares the efficiency and accuracy of several selection techniques. In each case, three commonly used words are shown to the subjects, who enter the words in one of three ways:
(1) Picking with a light pen from a menu. The desired words are guaranteed to be on the menu.
(2) Picking from the menu, but with no guarantee that the words are on the menu. If the words are not found, the subject types them in.
(3) Typing in all three words.

The results suggest that menu selection with a light pen is a faster and more accurate technique than the other two techniques. The second technique, which requires the use of two devices, is significantly slower.

12. W. K. English, D. C. Engelbart, and M. L. Berman, "Display Selection Techniques for Text Manipulation," *IEEE Trans. Human Factors in Electronics,* Vol. HFE-8, No. 1, Mar. 1967, pp. 5-15.
This experiment compared the performance of five selection devices: the light pen and four locators used to move a cursor to a target—the mouse, the Grafacon (a now obsolete tablet), the knee control, and the joystick. Results indicate that for the experienced user the mouse is the fastest and the least error-prone device, while for inexperienced users the knee control and the light pen are marginally faster than the mouse. For all users the mouse is more accurate, with error rates typically half of those for other devices. At the other extreme, the joystick used as either an absolute or a velocity-controlled locator fared the poorest both in speed and accuracy. In spite of a few hardware and software problems, which might have degraded the performance of the light pen and the joystick, the experiment was well coordinated and the procedures clearly described.

13. A. F. Fields, R. E. Maisano, and C. F. Marshall, *A Comparative Analysis of Methods for Tactical Data Inputting,* Battlefield Information Systems Technical Area, Sept. 1978, US Army Research Institute for the Behavioral and Social Sciences, 5001 Eisenhower Ave., Alexandria, Va.
Evaluates four methods of entering data in terms of speed, accuracy, and ease of learning. The data, taken from written intelligence reports, is in part numeric and in part words selected from a predefined set of terms. The techniques studied are:
(1) Typing numeric codes, each corresponding to a term, into a displayed form (label type-in).
(2) Typing as in 1, but with error correction attempted by the computer (label type-in with error correction).
(3) Menu selection from a list of legal terms by means of a trackball.
(4) Typing only sufficient digits or characters to uniquely identify a term, using either the appropriate numeric code or the term itself. The computer automatically completes the entry (label type-in with autocompletion).
Menu selection with a trackball was the most accurate input technique, and ranked second in speed performance, only slightly slower than label type-in. The results suggest that menu selection with occasional typing is a viable interaction technique for text entry, if the set of data is fixed and small.

14. R. Haller, H. Mutschler, and M. Voss, "Comparison of Input Devices for Correction of Typing Errors in Office Systems," *Proc. IFIP Interact 84 Conf.* Vol. 2, pp. 218-223 (to be published by Elsevier Science, Amsterdam).

15. N. C. Goodwin, "Cursor Positioning on an Electronic Display Using Lightpen, Lightgun, or Keyboard for Three Basic Tasks," *Human Factors*, Vol. 17, No 3, 1975, pp. 289-295.
This experiment compares the performance of the light pen, the lightgun (a variant of the light pen using a pistol grip), and step keys with cursor match for selection on an alphanumeric display. The only measurement considered was the time taken to move the cursor to a target character on the display screen, and then to type a replacement char-

acter on the keyboard. Results showed no significant difference between the light pen and the light gun, but both devices were faster than the cursor positioning keys. In the extreme case, in which the subject moves the cursor large distances over the display, step keys were found to be five times slower. However, this is partly because the step keys force cursor movement to be line-by-line rather in an arbitrary up-down-left-right direction.

16. G. W. Irving, J. J. Horineek, D. H. Walsch, and P. Y. Chan, *ODA Pilot Study II: Selection of an Interactive Graphics Control Device for Continuous Subjective Functions Applications* (report no. 215-2), Integrated Sciences Corp., Santa Monica, Calif., Apr. 1976.

    Evaluates three locator techniques—light pen with tracking cross, trackball, and joystick—to move a cursor to a random position and then do freehand sketching of an equilateral triangle and a circle. The performance measures were the straightness of the sides of the triangle, the closeness of the drawn triangle to an equilateral triangle, and the standard deviation of sample points on the drawn circle from the centroid of the circle. The trackball was preferred over the others for the tasks described. Because the evaluation criteria used were not very realistic and because the numbers of subjects and replications were small, the results require further validation.

17. M. H. Mehr and E. Mehr, "Manual Digital Positioning in 2 Axes: A Comparison of Joystick and Track Ball Controls," *Proc. 16th Annual Meeting Human Factors Society,* Oct. 1972.

18. C. S. Morrill, N. C. Goodwin, and S. L. Smith, "User Input Mode and Computer-Aided Instruction," *Human Factors,* Vol. 10, No. 3, 1968, pp. 225-232.

19. J. D. Foley, V. Wallace, and P. Chan, "The Human Factors of Interaction Techniques," The George Washington Univ. Institute for Information Science and Technology, tech. report GWU-IIST-81-03, Washington, D.C., 1981.

20. E. J. McCormick, *Human Factors in Engineering and Design,* 5th ed., 1982, McGraw-Hill, New York, 491 pp.

21. A. Chapanis, "Design of Controls," Chapter 3 in ref. no. 23, pp. 345-380.

22. W. E. Woodson and D. W. Conover, *Human Engineering Guide for Equipment Designers,* 2nd ed., Univ. of Calif. Press, 1964.

23. H. P. Van Cott and R. G. Kinkade, *Human Engineering Guide to Equipment Design,* revised edition, US Government Printing Office (stock no. 008-051-00050-0, catalog no. D4.10:EN3), 752 pp.

24. A. Chapanis, *Man-Machine Engineering,* Wadsworth, Belmont, Calif., 1965.

25. P. M. Fitts, "The Information Capacity of the Human Motor System in Controlling Amplitude of Movement," *J. Experimental Psychology,* Vol. 47, 1954, pp. 381-391.

26. T. B. Sheridan and W. R. Ferrell, *Man-Machine Systems: Information, Control, and Decision Models of Human Performance,* MIT Press, Cambridge, Mass., 1974, 452 pp.

27. A. Chapanis, *Research Techniques in Human Engineering,* Johns Hopkins Press, 1959, 310 pp.

28. D. Meister, *Human Factors: Theory and Practice,* Wiley-Interscience, New York, 1971, 415 pp.

29. D. Meister, *Behavioral Foundations of System Development,* Wiley-Interscience, New York, 1976, 376 pp.

30. A. Cakir, D. J. Hart, and T. F. M. Stewart, *Visual Display Terminals: A Manual Covering Ergonomics, Workplace Design, Health and Safety, Task Organization,* Wiley-Interscience, New York, 1980, 307 pp.

31. S. Card, T. Moran, and A. Newell, "The Keystroke-Level Model for User Performance Time with Interactive Systems," *Comm. ACM,* Vol. 23, No. 7, July 1980, pp. 396-410.

32. S. Card, T. Moran, and A. Newell, *The Psychology of Human-Computer Interaction,* Lawrence Erlbaum Associates, Hillsdale, N.J., 1983.

33. W. M. Newman and R. F. Sproull, *Principles of Interactive Computer Graphics,* 2nd ed., McGraw-Hill, New York, 1979.

34. T. P. Moran, "Introduction to the Command Language Grammar: A Representation for the User Interface of Interactive Computer Systems, " Xerox Palo Alto Research Center, report SSL-78-3, 1978.

35. R. M. Dunn, "A Philosophical Prelude to Methodology of Interaction," *Methodology of Interaction,* North-Holland, Amsterdam, May 1979, pp. 183-188.

36. G. D. Hornbuckle, "The Computer Graphics/User Interface," *IEEE Trans. Human Factors in Electronics,* Vol. HFE-8, No. 1, Mar. 1967, pp. 17-22.

37. P. Lindsay and D. Norman, *Human Information Processing,* 2nd ed., Academic Press, New York, 1977.

38. R. Lachman, J. L. Lachman, and E. C. Butterfield, *Cognitive Psychology and Information Processing: An Introduction,* Lawrence Erlbaum Associates, Hillsdale, N.J., 1979, 573 pp.

39. A. G. Reynolds and P. W. Flagg, *Cognitive Psychology,* Winthrop Publishers, Cambridge, Mass., 457 pp.

40. U. Neisser, *Cognitive Psychology,* Prentice-Hall, Englewood Cliffs, N.J., 1967, 351 pp.

41. G. Underwood, ed., *Strategies of Information Processing,* Academic Press, London, 1978, 455 pp.

42. L. G. Nilsson, ed., *Perspectives on Memory Research: Essays in Honor of Uppsala University's 500th Anniversary,*" Lawrence Erlbaum Associates, Hillsdale, N.J. 1979, 400 pp.

43. P. A. Kolers, M. E. Wrolstad, and H. Bouma, eds., *Processing of Visible Language,* Vol. 1, Plenum Press, New York, 1979, 537 pp.

44. J. A. Michon, E. G. Eijkman, G. J. de Klerk, and F. W. Len, *Handbook of Psychonomics,* Vols. I and II, North-Holland, Amsterdam, 1979.

45. J. Gould, "Visual Factors in the Design of Computer-Controlled CRT Displays," *Human Factors,* Vol. 10, 1968, pp. 462-476.

46. J. Grimes, "Effects of Patterning on Flicker Frequency," *Proc. Human Factors Society Conf.,* 1983, pp. 46-50.

47. W. Tracz, "Computer Programming and the Human Thought Process," *Software—Practice and Experience,* Vol. 9, 1979, pp. 127-137.

48. G. Loftus and E. Loftus, *Human Memory—The Processing of Information,* Lawrence Erlbaum Associates, Hillsdale, N.J. 1976.

49. W. Buxton, "Lexical and Pragmatic Considerations of Input Structures," *Proc. Graphical Input Interaction Technique Workshop Summary,* published as *Computer Graphics,* Vol. 17, No. 1, Jan. 1983, pp 31-37.

50. S. Treu, "Interactive Command Language Design Based on Required Mental Work," *Int'l J. Man-Machine Studies,* Vol. 7, No. 1, 1975, pp. 135-149.

51. E. Britton, J. Lipscomb, and M. Pique, "Making Nested Rotations Convenient for the User," *Computer Grahpics* (Proc. Siggraph 77), Vol. 12, No. 3, 1978, pp. 222-227.

Subimage motion in a three-dimensional computer graphics system is much easier for the user to control if the subimage moves in the same direction as his hand while he manipulates the control device. The implementation of such coordinated motion of hand and subimage implies modification of the normal procedure for calculating transformations. The convenience of such manipulation also depends on appropriate selection or design of the input device.

Paper reviews the relevant attributes of locator devices and presents an approach to their selection. Presents the mathematics of transformation nesting and "compensation" to preserve motion synchrony. Offers a case history of an interactive graphics system whose human factors were improved by these techniques.

52. "General Methodology and the Proposed Core System" (Report of the Graphics Standards Planning Committee), *Computer Graphics* (Proc. Siggraph 79), Vol. 13, No. 3, Aug. 1979, 179 pp.

53. L. Caruthers, J. van der Bos, and A. van Dam, "A Device-Independent General Purpose Graphic System for Stand-Alone and Satellite Graphics," *Computer Graphics* (Proc. Siggraph 77), Vol. 11, No. 2, summer 1977, pp. 112-119.

54. V. L. Wallace, "The Semantics of Graphic Input Devices," *Proc. Siggraph/Sigplan Symp. Graphics Languages,* Apr. 1976, pp. 61-65.

55. W. M. Newman, "A Graphical Technique for Numerical Input," *Computing J.,* Vol. 11, May 1968, pp. 63-64.

56. M. Ohlson, "System Design Considerations for Graphics Input Device," *Computer,* Vol. 11, No. 11, Nov. 1979, pp. 9-18.

57. J. Thomas and G. Hamlin, "Graphical Input Interaction Technique Workshop Summary," *Computer Graphics* (ACM), Vol. 17, No. 1, Jan. 1983, pp. 5-30.

58. K. Snowberry et al., "Computer Display Menus," *Ergonomics,* Vol. 26, No. 7, 1983, pp. 699-712.

59. R. Doughty and J. Kelso, "Depth vs. Breadth Trade-Offs in Menu Design," unpublished manuscript, Dept. of EE&CS, The George Washington Univ., Washington, D.C.

60. G. T. Uber, P. E. Williams, B. L. Hisey, and R. G. Siekert, "The Organization and Formatting of Hierarchical Displays for the On-Line Input of Data," *AFIPS Conf. Proc.,* Vol. 33, 1968, pp. 219-226.

61. S. Card, "User Perceptual Mechanisms in the Search of Computer Command Menus," *Proc. Human Factors in Computer Systems Conf.,* Washington, D.C., ACM, Mar. 1982, pp. 190-196.

62. K. Hemenway, "Psychological Issues in the Use of Icons in Command Menus," *Proc. Human Factors in Computer Systems Conf.,* Washington, D.C., ACM, Mar. 1982, pp. 20-24.

63. W. Bewley, T. Roberts, D. Schroit, and W. Verplank, "Human Factors Testing in the Design of Xerox's 8010 'Star' Office Workstation," *Proc. CHI 83 Human Factors in Computing Systems,* ACM, New York, 1983, pp. 72-77.

64. J. L. Coffee, "A Comparison of Vertical and Horizontal Arrangements of Alpha-numerical Materials," *Human Factors,* Vol. 3, 1961, pp. 93-98.

65. A. Cropper and S. Evans, "Ergonomics and Computer Display Design," *Computer Bulletin,* Vol. 12, No. 3, July 1968, pp. 94-98.

66. P. M. Fitts and B. Radford, "Information Capacity of Discrete Motor Responses Under Different Cognitive Sets," *J. Experimental Psychology,* Vol. 71, 1966, pp. 475-482.

67. A. Albert, "The Effect of Graphic Input Devices on Performance in a Cursor Positioning Task," *Proc. Human Factors Society Conf.,* 1982.

68. A. G. Vartabedian, "Human Factors Evaluation of Several Cursor Forms for Use on Alphanumeric CRT Displays," *IEEE Trans. Human Factors in Electronics,* Vol. HFE-6, Sept. 1965, pp. 74-83.

69. W. L. Jenkins and M. B. Connor, "Some Design Factors in Making Settings on a Linear Scale," *J. Appl. Psychology,* Vol. 33, 1949, p. 395.

70. W. L. Jenkins, L. O. Maas, and D. Rigler, "Influence of Friction in Making Settings on a Linear Scale," *J. Appl. Psychology,* Vol. 34, 1950, p. 434.

71. W. L. Jenkins and A. C. Karr, "The Use of a Joystick in Making Settings on a Simulated Scope Face," *J. Appl. Psychology,* Vol. 38, 1954, p. 457.

72. C. Herot and G. Weinzapfel, "One-Point Touch Input of Vector Information for Computer Displays," *Computer Graphics* (Proc. Siggraph 78), Vol. 12, No. 3, Aug. 1978, pp. 210-216.

73. R. W. Thornton, "The Number Wheel: A Tablet Based Valuator for Three-dimensional Positioning," *Computer Graphics* (Proc. Siggraph 79), Vol. 13, No. 2, Aug. 1979, pp. 102-107.

74. R. Brown, "On-Line Computer Recognition of Hand-Printed Characters," *IEEE Trans. Computers,* Vol. EC-13, No. 12, Dec. 1964, pp. 750-752.

75. W. Teitelman, "Real-Time Recognition of Hand-drawn Characters," *AFIPS Conf. Proc.,* Vol. 24, 1964 FJCC, Spartan Books, Baltimore, Md., p. 559.

76. D. Devoe, "Alternatives to Handprinting in the Manual Entry of Data," *IEEE Trans. Human Factors in Electronics,* Vol. HFE-8, No. 1, Jan. 1967, pp. 21-32.

77. J. Foley and A. van Dam, *Fundamentals of Interactive Computer Graphics,* Addison-Wesley, Reading, Mass., 1982.

Additional readings

Barmack, J. E., and H. W. Sinaiko, *Human Factors Problems in Computer-Generated Graphics Displays,* AD 636 170, Institute for Defense Analysis, Research and Engineering Support Division, 400 Army-Navy Drive, Arlington, Va., 1966.

A review of current practices in computer-generated graphics displays from the point of view of engineering psychology. Input devices, which are integral to man-computer systems, are also considered. Theories of cognition are examined with respect to their applicability to computer graphics.

Bennett, J. L., "User Interfaces in Interactive Systems," *Annual Review of Information Sciences and Technology, Encyclopedia Britannica,* Vol. 111, No. 7, 1973, pp. 159-196.

Bloomfield, J. R., "Experiments in Visual Search," *Visual Search Symp.,* National Research Council, Div. of Behavioral

Sciences, spring 1970, Washington, D.C., National Academy of Sciences, Vol. 7, 1973, p. 150.

Boies, S. J., "User Behavior in an Interactive Computer System," *IBM Report,* 1972.

Chambers, J. B., and H. C. Stockbridge, "Comparison of Indicator Components and Push-Button Recommendation," *Ergonomics,* Vol. 3, No. 4, 1970, pp. 401-420.

Christ, R. E., "Review and Analysis of Color Coding Research for Visual Displays," *Human Factors,* Vol. 17, No. 6, 1975, pp. 542-570.

Analyzes 42 studies published between 1952 and 1973, whose results can be used to determine the effectiveness of color codes relative to various types of achromatic codes. Quantitative analyses indicated that color may be a very effective performance factor under some conditions, but detrimental under others. Derives tentative conclusions about the nature of these conditions. Provides a guide for design decisions and an indication of knowledge gaps.

Conrad, R., "Short-term Memory Factor in the Design Data-Entry Keyboard," *J. Applied Psychology,* Vol. 50, 1966, pp. 353-356.

De Greene, K. B., ed., *Systems Psychology,* McGraw-Hill, New York, 1970, ch. 10.

Drury, C. G., and M. R. Clement, "Effect of Area, Density and Number of Background Characters on Visual-Search," *Human Factors,* Vol. 20, No. 5, 1978, pp. 369-384.

In a search task, area of search field, density of background characters, and number of background characters are not independent. Many authors have found increases in search times with each of these factors but have not adequately controlled all three together. In this experiment, eight subjects searched a set of fields covering combinations of all three variables. Search time was found to depend most on number of background characters, but there were significant effects due to the other two variables. For a constant number of background characters, search time decreases as density increases. Direct visual lobe measurements confirmed these findings, which could have importance in visual inspection tasks.

Englebart, D. C., "Design Considerations for Knowledge Workshop Terminals, *AFIPS Conf. Proc.*, Vol. 42, 1973, pp. 221-227.

Flanagan, J. L., "Computers That Talk and Listen: Man-Machine Communication by Voice," *Proc. IEEE,* Vol. 64, No. 4, Apr. 1976, pp. 405-415.

Computer techniques now emerging in the laboratory promise new capabilities for voice communication between man and machine. Three modes of interaction are of special interest: computer voice readout of stored information, automatic verification of a caller's identity by means of his voice signal, and automatic recognition of spoken commands. Applications extend to voice-directed installation of telephone equipment, authentication by voice of a credit customer or of an individual requesting readout of privileged information, and voice-controlled services such as repertory dialing or automatic booking of travel reservations.

Fitts, P. M., and J. R. Peterson, "Information Capacity of Discrete Motor Responses," *J. Experimental Psychology,* Vol. 67, 1964, pp. 103-112.

Foley, J. D., "The Structure of Command Languages," in *Methodology of Interaction*, R. A. Guedj, et al., eds., North-Holland, Amsterdam, 1980, pp. 227-234.

Gaines, B., and P. Facey, "Some Experiences in Interactive System Development and Application," *Proc. IEEE,* Vol. 63, No. 6, June 1975.

Gallo, J., and J. Levine, *Human Factors in the Design of an Observer's Keyset,* US Navy Electronics Laboratory, San Diego, Calif., Oct. 1966.

On a chord keyboard, users enter a desired command by depressing the appropriate patterns of keys. A pilot study done before the experiments conducted in this research found that command entry on a chord keyboard in the manual mode is much slower and more error-prone than in the automatic mode. Subjects entered one of 26 alternatives coded in five-bit patterns by pressing the appropriate keys to automatically enter the command; in the manual mode, only by striking the entry bar, which is similar to a space bar, could the subject enter the message. Even though no experimental results are given, the authors strongly state that the use of the entry bar greatly increased the error rates and sharply decreased the input rates.

Guedj, R., et al., eds., *Methodology of Interaction,* North-Holland, Amsterdam, 1980.

Hayes, P., E. Ball, and R. Reddy, "Breaking the Man-Machine Communication Barrier," *Computer,* Vol. 14, No. 3, Mar. 1981, pp. 19-30.

Holmgren, J. E., "Effect of a Visual Indicator on Rate of Visual Search-Evidence for Processing Control," *Perception and Psychophysics,* Vol. 15, No. 3, 1974, pp. 544-550.

Hunstad, A., and B. M. Brown, "An Evaluation of Certain Interactive Input Devices Associated with Computer Driven Displays," *Agard Conf. Proc. Data Handling Devices,* 1970, p. 8.

Hutchinson, A., *Labanotation,* Theatre Arts Books, New York, 1970.

Johnson, J. K., "Touching Data," *Datamation,* Vol. 23, No. 1, Jan. 1977, pp. 70-72.

Kennedy, T. C., "Design of Interactive Procedures for Man-Machine Communication," *Int'l J. Man-Machine Studies,* Vol. 7, 1975, pp. 233-247.

Kennedy, T. C., "Some Behavior Factors Affecting the Training of Naive Users of an Interactive Computer System," *Int'l J. Man-Machine Studies,* Vol. 7, 1975, pp. 817-834.

Klapp, S. T., "Feedback Motor Programming in the Control of Aimed Movements," *J. Experimental Psychology: Human Perception & Performance,* Vol. 104, Nos. 1, 2, May 1975, pp. 147-153.

Klemmer, E. T., "Keyboard Entry," *Applied Ergonomics,* Vol. 2, No. 1, 1971, pp. 2-6.

Summarizes many recent studies of keyboard entry, with emphasis on performance data and fundamental questions about the design of keyboards. Reviews the roles of auditory and visual feedback and physiological measurements. Gives typical speed and error rates for several types of situations and operators. Discusses other methods of data entry as well as source documents, ordering of keys, keyboard interlocks, and chord keyboards.

Knowlton, K. C., "Virtual Pushbuttons as a Means of Person Machine Interaction," *Proc. IEEE Conf. Computer Graphics, Pattern Recognition, and Data Stucture,* May 1975, pp. 350-351.

Kroemer, K. H., "Human Engineering the Keyboard," *Human Factors,* Vol. 14, No. 1, 1972, pp. 51-63.

The standard typewriter keyboard serves as a model for keyboards of teletypewriters, desk calculators, consoles, computer keysets, cash registers, etc. This man-machine interface should be designed to allow high-frequency, error-free operation with the least possible strain on the operator. This paper discusses several feasible biomechanical improvements of the keyboard. Describes experimental findings supporting the following design concepts: (1) the keys should be arranged in a "hand-configured" grouping to simplify the motion patterns of the fingers; (2) the keyboard sections allotted to each hand should be physically separated to facilitate the positioning of the fingers; and (3) the keyboard sections allotted to each hand should be declined laterally to reduce muscular strain of the operator.

McMichael, E., and S. McCarthy, "Visual Search through Words and Nonwords in Horizontal and Vertical Orientations," *Perceptual and Motor Skills,* Vol. 41, No. 3, 1975, pp. 740-742.

Miller, L. A., and J. C. Thomas, "Behavioral Issues in the Use of Interactive Systems," *Int'l J. Man-Machine Studies,* Vol. 9, No. 56, Sept. 1977, pp. 509-536.

Miller, L. H., "An Investigation of the Effects of Output Variability and Output Bandwidth on User Performance in an Interactive Computer System," Report ISI/RR-76-50, Information Science Institute, University of Southern California, 1976.

Miller, R. B., "Response Time in Man-Computer Conversational Transactions," *Proc. AFIPS Conf.,* Vol. 33, AFIPS Press, Monvale, N.J., 1968, pp. 267-277.

Moran, T., ed., *Computing Surveys,* Special Issue: The Psychology of Human-Computer Interaction, Vol. 13, No. 1, Mar. 1981.

Peterson, J. L., "Petri Nets," *Computing Surveys,* Vol. 9, No. 3, 1977.

Ramsey, H. R., M. E. Atwood, and P. J. Kirshbaum, "A Critically Annotated Bibliography of the Literature of Human Factors in Computer Systems," tech. report SAI-78-070-DEN, Science Applications, Inc., May 31, 1978.

Richard, E. G., "Development of Display Guidelines for Human Use—An Experimental Approach," *Human Factors and Graphics,* IBM Corp., New York, 1975.

Richtie, G. J., and J. A. Turner, "Input Devices for Interactive Graphics," *Int'l J. Man-Machine Studies,* Vol. 7, 1975, pp. 639-660.

Root, R. T., and R. Sadacca, "Man-computer Communication Techniques: Two Experiments," *Human Factors,* Vol. 9, 1967, pp. 521-528.

Rouse, W. B., "Design of Man-Computer Interfaces for On-Line Interactive Systems," *Proc. IEEE,* Vol. 63, No. 6, June 1975, pp. 847-857.

Seibel, R., "Data Entry Devices and Procedures," in *Human Engineering Guide to Equipment Design* (revised ed.), H. P. Van Cott and R. G. Kinkade, eds., US Government Printing Office, Washington, D.C., 1972, pp. 311-344.

Shackel, B., and P. Shipley, *Man-Computer Interaction: A Review of Ergonomics Literature and Related Research* (report no. DMP-3473), EMI Electronics Ltd., Hayes, Middlesex, England, Feb. 1970.

Sheridan, T. B., ed., *IEEE Trans. Human Factors Electronics* (Special Issue on Man-Computer Input-Output Techniques), Vol. HFE-8, Mar. 1967.

Sneeringer, J., "User-Interface Design for Text Editing: A Case Study," *Software—Practice and Experience,* Vol. 8, 1978, pp. 543-577.

Stewart, T. F. M., "Displays and the Software Interface," *Applied Ergonomics,* Vol. 7, No. 3, 1976, pp. 137-146.

Teichner, W. H., and M. J. Krebs, "Visual Search for Simple Targets," *Psychological Bull.,* Vol. 81, No. 1, 1974, pp. 15-28.

Thompson, D. A., "Interface Design for an Interactive Information Retrieval System: a Literature Survey and a Research System Description," *J. Amer. Soc. for Information Science,* Vol. 22, 1971, pp. 361-373.

Tilbrook, D., "A Newspaper Page Layout System," MSc thesis, Dept. of Computer Science, Univ. of Toronto, Canada, 1976. Demonstrated in *Siggraph Video Tape Review,* Vol. 1, May 1980.

Vartabedian, A. G., "Human Factors Evaluation of Several Cursor Displays," *IEEE Trans. Man-Machine Systems,* Vol. MMS-11, No. 2, 1970, pp. 132-137.

Vaughan, J., "Scanning Strategies in Visual-Search," *Bull. Psychonomic Society,* Vol. 8, No. 4, 1976, pp. 262-263.

James D. Foley is a professor of electrical engineering and computer science with The George Washington University, Washington, DC. His research interests are computer graphics and human factors. He initiated, in 1978, one of the first computer science courses devoted exclusively to the design of user-computer interfaces. Foley is coauthor, with A. van Dam, of *Fundamentals of Interactive Computer Graphics*, an associate editor of *Transactions on Graphics*, and a new member of the *IEEE Computer Graphics and Applications* Editorial Board.

Foley received the BS in electrical engineering from Lehigh University in 1964. His PhD in computer, information, and control engineering is from the University of Michigan. He is a member of Tau Beta Pi, Phi Beta Kappa, ACM, and NCGA and is a senior member of the IEEE.

Victor L. Wallace is a professor of computer science at The University of Kansas, Lawrence, Kansas, where he was also department chairman from 1976-1983. He was a faculty member in computer science at the University of North Carolina from 1969-1976 and in electrical engineering at the University of Michigan from 1957-1969. His research interests include the theoretical foundations of computer system performance models, software tools for interactive graphics, the human interface for interactive graphics systems, analysis of numerical algorithms, operating systems, and computer networks.

Wallace received his BEE and MEE from the Polytechnic Institute of New York and his PhD from the University of Michigan. He is a member of the ACM, IMS, AAUP, Sigma Xi, and Tau Beta Pi and a senior member of the IEEE.

Peggy Chan is a supervisory consultant for Arthur Young & Company in Washington, DC, where she is currently managing financial database and management information system projects for the US Departments of Treasury and Commerce. Previously, she was a member of the technical staff at Computer Sciences Corporation. As a graduate research/teaching assistant at The George Washington University, she worked on projects to identify the human factors and user requirements aspects of different interaction techniques for computer graphics systems. Her interests include computer system methodology, computer graphics systems, and mini and mainframe application systems and DBMSs.

Chan received the BS in mathematics and the BA in English from Pacific Lutheran University, Tacoma, Washington, and the MS in computer sciences from The George Washington University in Washington, DC.

Foley can be contacted at The George Washington University, EE&CS Dept., Washington, DC 20052. Wallace's address is Department of Computer Science, The University of Kansas, Lawrence, KS 60045. Chan's address is Arthur Young & Company, 1025 Connecticut Avenue, NW, Washington, DC 20036.

APPENDIX H

DESIGNING THE USER INTERFACE

Strategies for Effective Human-Computer Interaction / 5th Edition

Ben Shneiderman & Catherine Plaisant

Editor-in-Chief: Michael Hirsch
Editorial Assistant: Stephanie Sellinger
Managing Editor: Jeff Holcomb
Senior Manufacturing Buyer: Carol Melville
Art Director: Linda Knowles
Senior Media Producer: Bethany Tidd
Director of Marketing: Margaret Waples
Marketing Manager: Erin Davis
Marketing Coordinator: Kathryn Ferranti
Director of Marketing Services: Stacey Abraham
Online Product Manager: Bethany Tidd
Manager, Rights and Permissions: Diann Korta
Permissions Project Manager: Shannon W. Barbe
Text Permission Coordinator: Dana Weightman
Visual Research Manager: Beth Brenzel
Image Permission Coordinator: Ang'john Ferreri
Full-Service Project Management: Rose Kernan, Nesbitt Graphics, Inc.
Composition and Illustrations: Nesbitt Graphics, Inc.
Text Designer: Jerilyn Bockorick, Nesbitt Graphics, Inc.
Cover Designer: Joyce Cosentino Wells
Cover Photos: (clockwise, from top left) © Jeff Savage; © Alan Becker/The Image Bank/Getty images; © Bob Daemmrich/The Image Works; © Michael DeYoung/Aurora/Getty Images; © Jon Feingersh/Iconica/Getty Images; © Mel Yates/Cultura/Getty Images. Background: © Jason Reed/Photodisc/Getty Images

Access the latest information about Addison-Wesley titles from our World Wide Web site:
http://www.pearsonhighered.com/cs

Credits and acknowledgments borrowed from other sources and reproduced, with permission, in this textbook appear in the Acknowledgments section in the endmatter of this book.

Many of the designations by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed in initial caps or all caps.

Microsoft® and Windows® are registered trademarks of the Microsoft Corporation in the U.S.A. and other countries. Screen shots and icons reprinted with permission from the Microsoft Corporation. This book is not sponsored or endorsed by or affiliated with the Microsoft Corporation.

The programs and applications presented in this book have been included for their instructional value. They have been tested with care, but are not guaranteed for any particular purpose. The publisher does not offer any warranties or representations, nor does it accept any liabilities with respect to the programs or applications.

Library of Congress Cataloging-in-Publication Data

Shneiderman, Ben.
Designing the user interface / Ben Shneiderman, Catherine Plaisant; contributors, Maxine Cohen, Steven Jacobs. -- 5th ed.
p. cm.
Includes bibliographical references and index.
ISBN 0-321-53735-1 (alk. paper)
1. Human-computer interaction. 2. User interfaces (Computer systems) I.Plaisant, Catherine. II. Title.
QA76.9.H85S54 2009
005.4'37--dc22
2009003477

Addison Wesley
is an imprint of

PEARSON

ISBN-13: 978-0-321-53735-5
ISBN-10: 0-321-53735-1
10 9 8 7 6 5 4 3 2 1—CRK—13 12 11 10 09

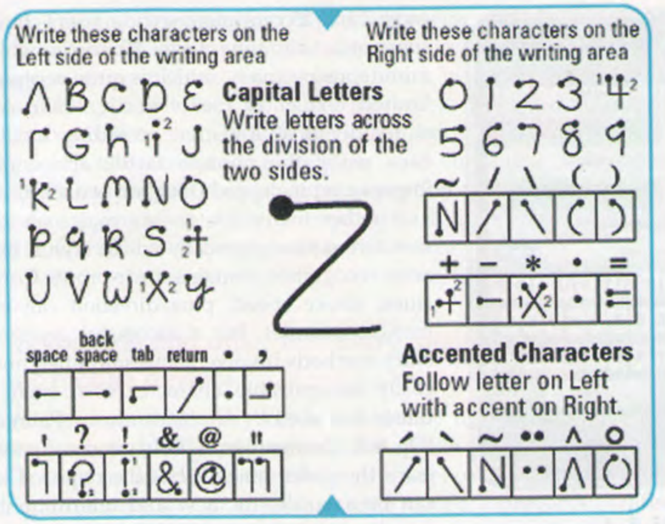

Write these characters on the Left side of the writing area
Write these characters on the Right side of the writing area
Capital Letters
Write letters across the division of the two sides.
A B C D E
G H I J
K L M N O
P Q R S T
U V W X Y
0 1 2 3 4
5 6 7 8 9
~ / \ ( )
+ − * • =
space
back space
tab
return
• ,
' ? ! & @ "
Accented Characters
Follow letter on Left with accent on Right.
´ ` ~ •• ^ o

FIGURE 8.5

The Palm Graffiti 2 characters (http://www.polatheschools.com/palm/documents/grafitti2_alphabet.pdf).

because the users can avoid learning commands, reduce the chance of typographic errors on a keyboard, and keep their attention on the display. The results are often faster performance, fewer errors, easier learning, and higher satisfaction. Pointing devices are also important for small devices and large wall displays that make keyboard interaction less practical.

The diversity of tasks, the variety of devices, and the strategies for using them create a rich design space (Hinckley, 2008). Physical device attributes (rotation or linear movement), dimensionality of movement (1, 2, 3 . . .), and positioning (relative or absolute) are useful ways of categorizing devices, but here we focus on tasks and degree of directness as organizing dimensions.

8.3.1 Pointing tasks

Pointing devices are useful for seven types of interaction tasks (expanded from the six tasks of Foley et al., 1984):

1. *Select*. Users choose from a set of items. This technique is used for traditional menu selection, the identification of objects of interest, or marking a part (for example, in an automobile design).

2. *Position*. Users choose a point in a one-, two-, three-, or higher-dimensional space. Positioning may be used to create a drawing, to place a new window, or to drag a block of text in a figure.
3. *Orient*. Users choose a direction in a two-, three-, or higher-dimensional space. The direction may rotate a symbol on the screen, indicate a direction of motion, or control the operation of a device such as a robot arm.
4. *Path*. Users rapidly perform a series of positioning and orientation operations. The path may be realized as a curving line in a drawing program, a character to be recognized, or the instructions for a cloth-cutting or other type of machine.
5. *Quantify*. Users specify a numeric value. The quantify task is usually a one-dimensional selection of integer or real values to set parameters, such as the page number in a document, the velocity of a vehicle, or the volume level of music.
6. *Gesture*. Users indicate an action to perform by executing a simple gesture, such as a swipe motion to the left (or right) to turn a page forward (or backward), or a rapid back and forth motion to erase.
7. *Text*. Users enter, move, and edit text in a two-dimensional space. The pointing device indicates the location of an insertion, deletion, or change. Beyond the simple manipulation of text are more elaborate tasks, such as centering, setting margins and font sizes, highlighting (boldface or underscore), and page layout.

It is possible to perform all these tasks with a keyboard by typing numbers or letters to select, entering integer coordinates to position, typing a number representing an angle to point or a number to quantify, making menu selections to select actions, and entering cursor-control commands to move around in the text. In the past, the keyboard was used for all of these purposes, but now most users employ pointing devices to perform the tasks more rapidly and with fewer errors; expert users can further improve performance by using keyboard shortcuts for tasks that are invoked frequently (e.g., Ctrl-C followed by Ctrl-V to copy and paste).

Pointing devices can be grouped into those that offer *direct control* on the screen surface, such as the touchscreen or stylus, and those that offer *indirect control* away from the screen surface, such as the mouse, trackball, joystick, graphics tablet, or touchpad. Within each category are many variations, and novel designs emerge frequently (Box 8.1).

8.3.2 Direct-control pointing devices

The lightpen was an early device that enabled users to point a tethered pen at a screen and then press a button on the pen to point at objects or draw on the screen. It was fragile, and users were required to pick up the device, so the

APPENDIX I

US006724403B1

(12) **United States Patent**
Santoro et al.

(10) **Patent No.: US 6,724,403 B1**
(45) **Date of Patent: Apr. 20, 2004**

(54) **SYSTEM AND METHOD FOR SIMULTANEOUS DISPLAY OF MULTIPLE INFORMATION SOURCES**

(75) Inventors: **Ovid Santoro**, London (GB); **Klaus Lagermann**, Copenhagen (DK)

(73) Assignee: **Surfcast, Inc.**, Palo Alto, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 259 days.

(21) Appl. No.: **09/702,325**

(22) Filed: **Oct. 30, 2000**

Related U.S. Application Data

(60) Provisional application No. 60/162,522, filed on Oct. 29, 1999.

(51) **Int. Cl.**^{7} **G06F 15/00**

(52) **U.S. Cl.** **345/765**; 345/790

(58) **Field of Search** 345/765, 766, 345/764, 729, 781, 788, 716, 717, 792, 790

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,555,775	A		11/1985	Pike	
4,653,020	A		3/1987	Cheselka et al.	
4,712,191	A		12/1987	Penna	
4,831,556	A	*	5/1989	Oono	345/786
5,157,384	A		10/1992	Greanias et al.	
5,394,521	A		2/1995	Henderson, Jr. et al.	
5,479,602	A		12/1995	Baecker et al.	
5,550,968	A	*	8/1996	Miller et al.	345/741
5,740,430	A		4/1998	Rosenberg et al.	
5,740,549	A		4/1998	Reilly et al.	
5,757,371	A	*	5/1998	Oran et al.	345/779
5,778,181	A		7/1998	Hidary et al.	
5,793,368	A		8/1998	Beer	
5,796,383	A		8/1998	Henshaw et al.	
5,796,401	A		8/1998	Winer	345/433
5,812,123	A	*	9/1998	Rowe et al.	725/43
5,813,007	A		9/1998	Nielsen	
5,831,664	A	*	11/1998	Wharton et al.	725/81
5,838,326	A	*	11/1998	Card et al.	345/775
5,841,418	A		11/1998	Bril et al.	345/3
5,848,352	A		12/1998	Dougherty et al.	
5,905,492	A		5/1999	Straub et al.	
5,918,237	A		6/1999	Montalbano	
5,929,854	A	*	7/1999	Ross	345/783
6,003,041	A		12/1999	Wugofski	
6,011,537	A		1/2000	Slotznick	
6,025,837	A		2/2000	Matthews, III et al.	
6,028,602	A		2/2000	Weidenfeller et al.	
6,160,553	A	*	12/2000	Robertson et al.	345/767
6,166,738	A	*	12/2000	Robertson et al.	345/839
6,188,405	B1	*	2/2001	Czerwinski et al.	345/764
6,411,275	B1	*	6/2002	Hedberg	345/156

OTHER PUBLICATIONS

Martin S Matthews and Erik B. Poulsen, FrontPage 2000: The Complete Reference, May 1, 1999, McGraw–Hil Osborne Media, Chpater 19, pp. 1–12.*

John Ross, ABCs of Internet Explore 4, Copyright 1997, Sybex, Chapter 13, pp. 1–3.*

Paul McFedries, The Complete Idiot's Guide to Window 95, Mar. 1997, 2nd Edition, pp. 3–7, 97, 101, 105–107, 379.*

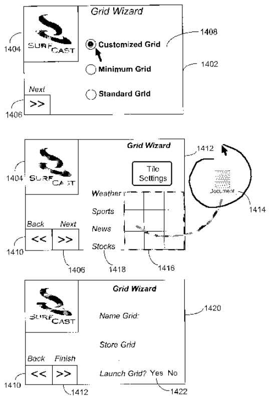
(List continued on next page.)

Primary Examiner—Cao (Kevin) Nguyen
(74) *Attorney, Agent, or Firm*—Pennie & Edmonds LLP

(57) **ABSTRACT**

A computerized method of presenting information from a variety of sources on a display device. Specifically the present invention describes a graphical user interface for organizing the simultaneous display of information from a multitude of information sources. In particular, the present invention comprises a graphical user interface which organizes content from a variety of information sources into a grid of tiles, each of which can refresh its content independently of the others. The grid functionality manages the refresh rates of the multiple information sources. The present invention is intended to operate in a platform independent manner.

52 Claims, 27 Drawing Sheets



OTHER PUBLICATIONS

PCT International Search Report, Application No. PCT/US00/29850, dated Jun. 25, 2001, 3 sheets.

Available Web Site: www.dodots.com Accessed on: May 9, 2001.

Available Web Site: www.snippets.com Accessed on: May 9, 2001.

Available Web Site: www.ububu.com Accessed on: May 9, 2001.

Available Web Site: www.chatb.com Accessed on: Nov 7, 2000.

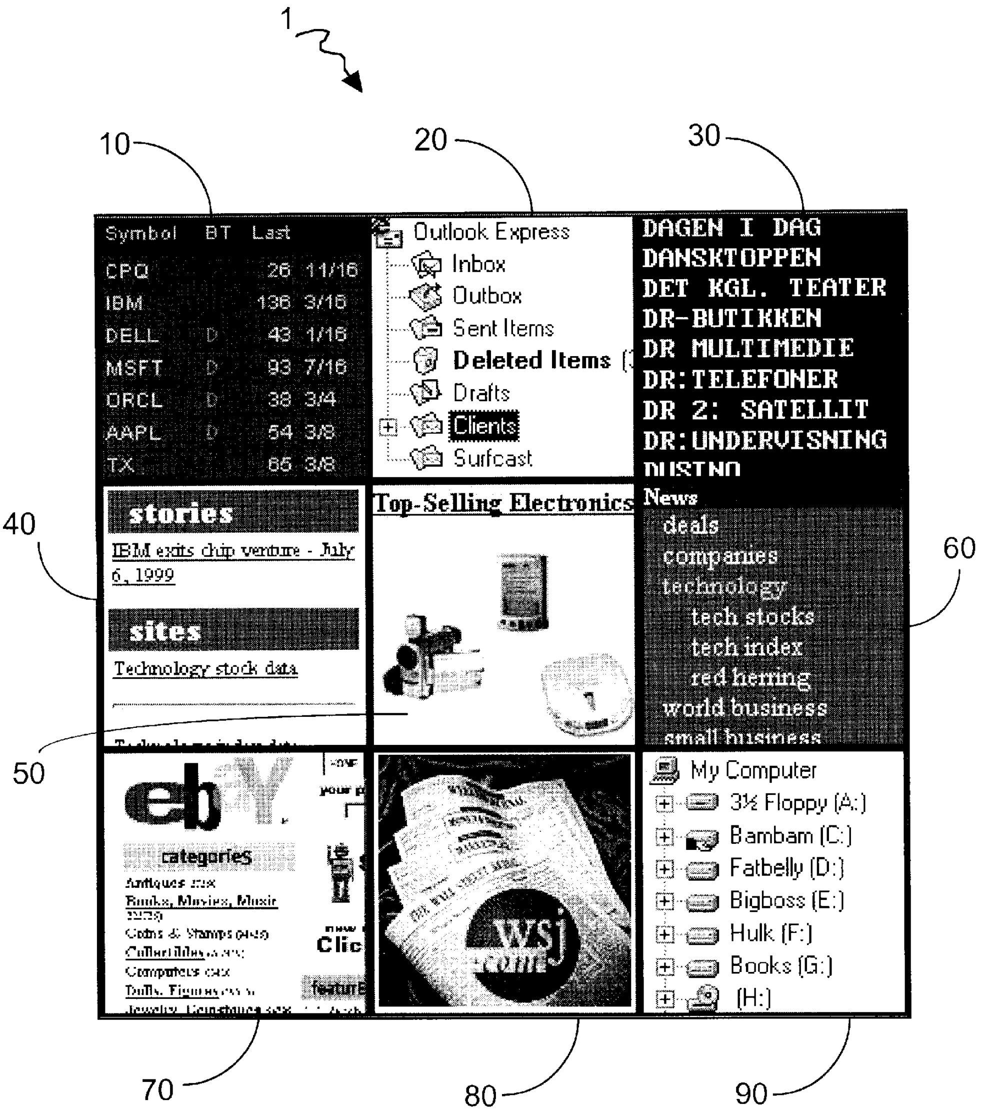
Duplex Multiplexer ,Sensormatic, Samsung, . . . ireless communications,hand helds,maxon Available Web Site: www.mindspring.com/~stancom/multi.html Accessed on: Nov. 7, 2000.

push technology. Available Web Site: www.whatis.com/WhatIs_Definition_Page/0,4152,213345,00.html Last Update: Jul. 7, 2000 Accessed on Nov. 7, 2000.

Clyman, John. Web Integration/Internet Explorer 4.0 Available Web Site: www.zdnet.com/pcmag/features/memphis/memphis1.htm Accessed on Nov. 7, 2000.

Oct. 2000, Product Spotlight: Non–browser based portal solution from Snippets Software, Inc., *Corporate Portals Letter* [Online] 1(10), 1–3. Available Web Site: www.snippets.com/download/Corporate_Portal_Article.pdf Accessed on May 9, 2001.

* cited by examiner



Symbol	BT	Last
CPQ		26 11/16
IBM		136 3/16
DELL	D	43 1/16
MSFT	D	93 7/16
ORCL	D	38 3/4
AAPL	D	54 3/8
TX		65 3/8

Fig. 1

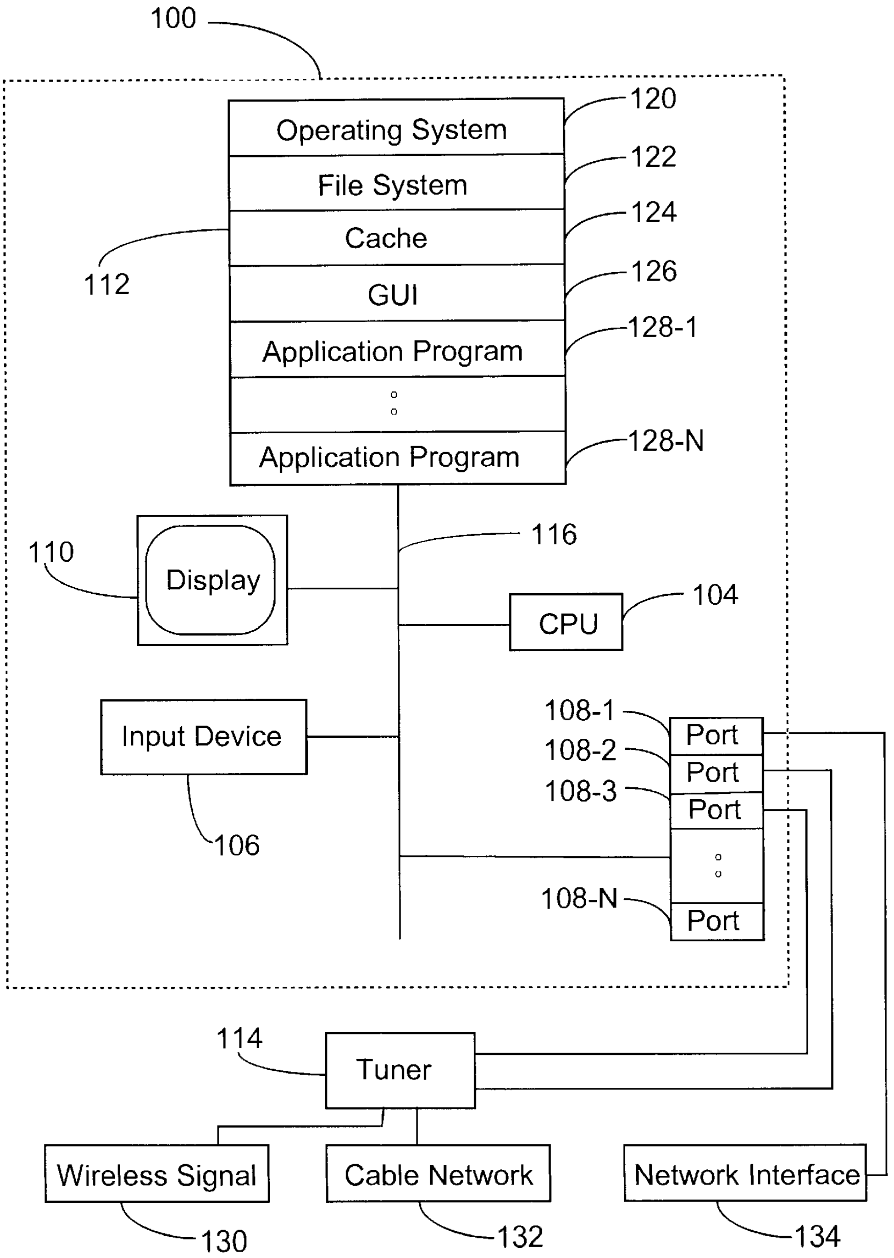

100
Operating System
120
File System
122
Cache
124
112
GUI
126
Application Program
128-1
Application Program
128-N
116
110
Display
CPU
104
Input Device
106
108-1
108-2
108-3
108-N
Port
Port
Port
Port
114
Tuner
Wireless Signal
Cable Network
Network Interface
130
132
134

Fig. 2

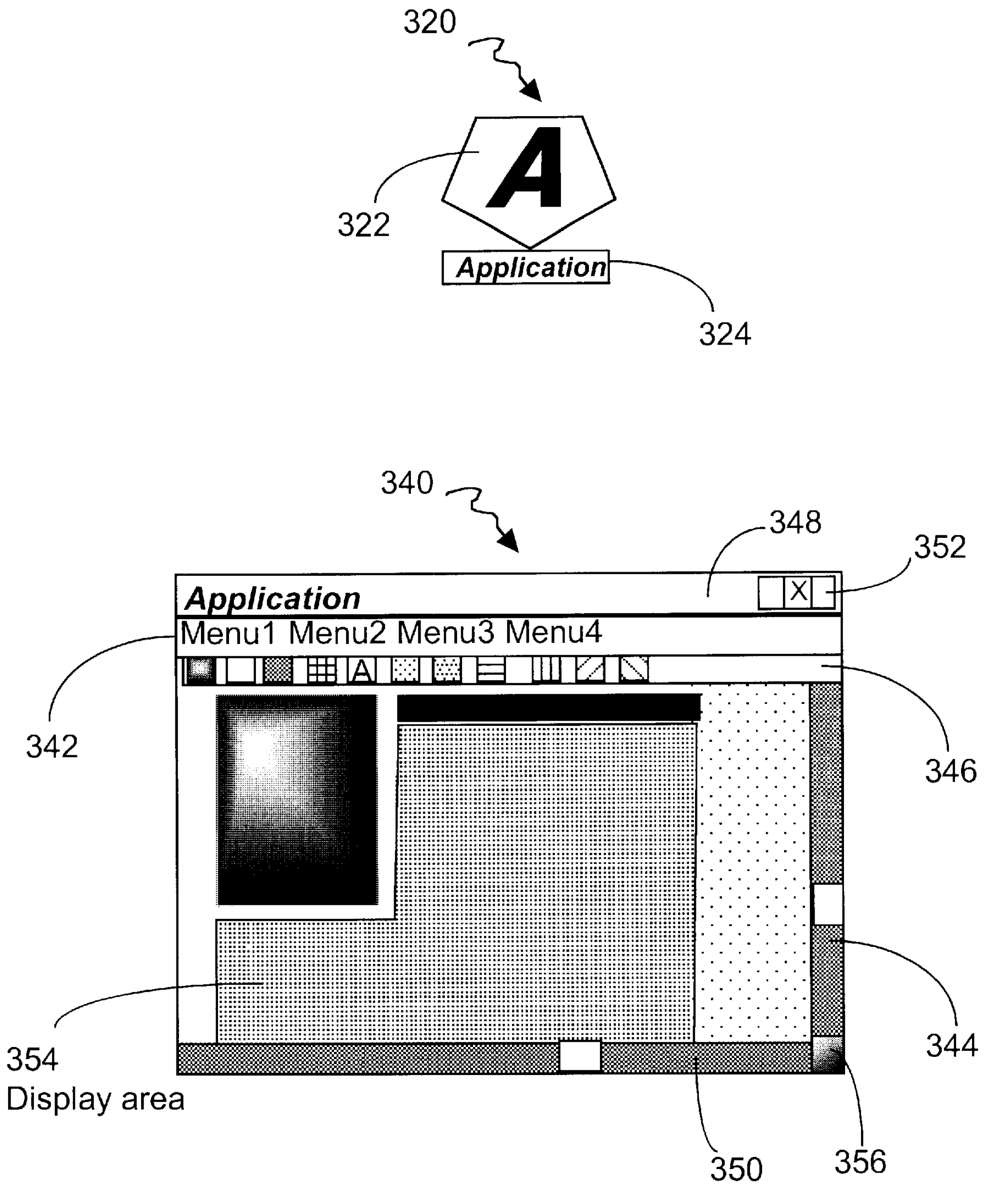


Fig.3
(Prior Art)

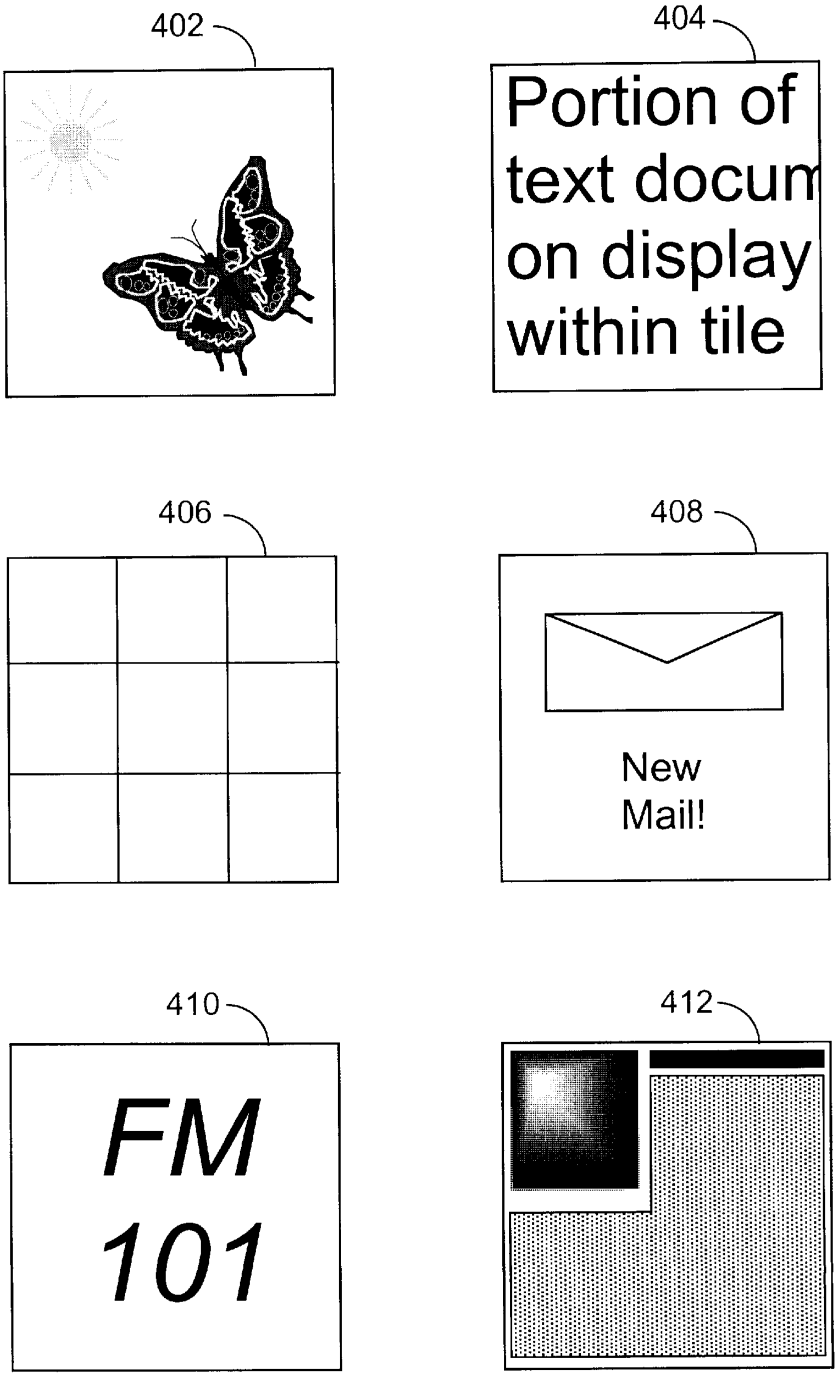

402
404
Portion of
text docum
on display
within tile
406
408
New
Mail!
410
FM
101
412

Fig. 4

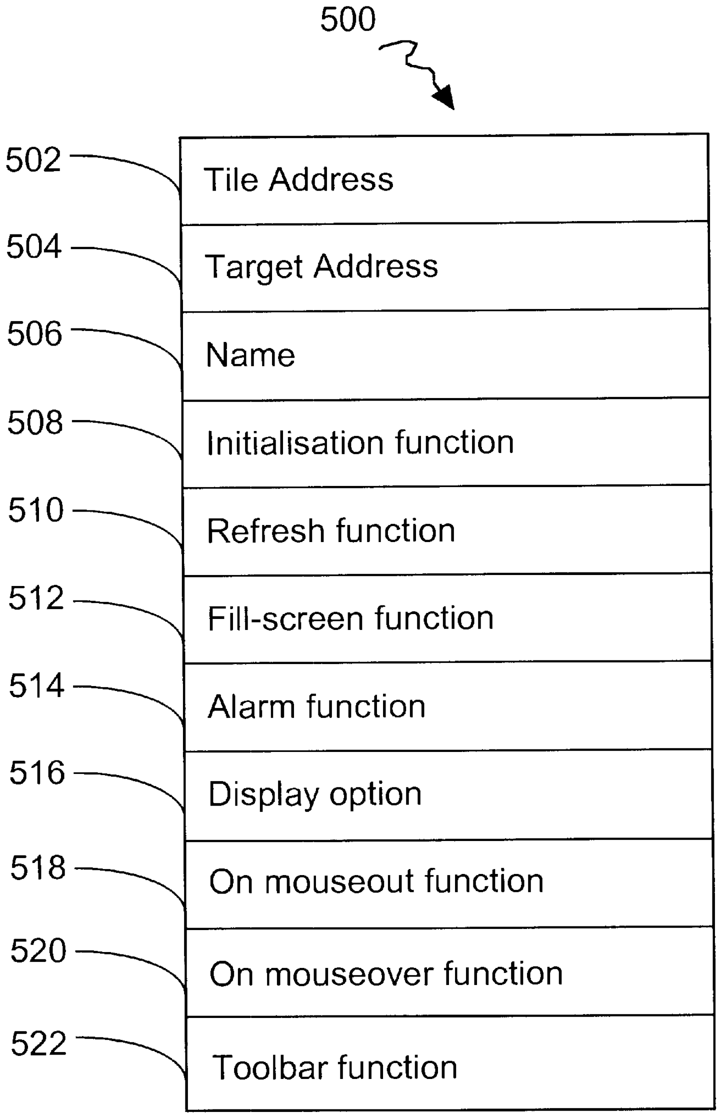
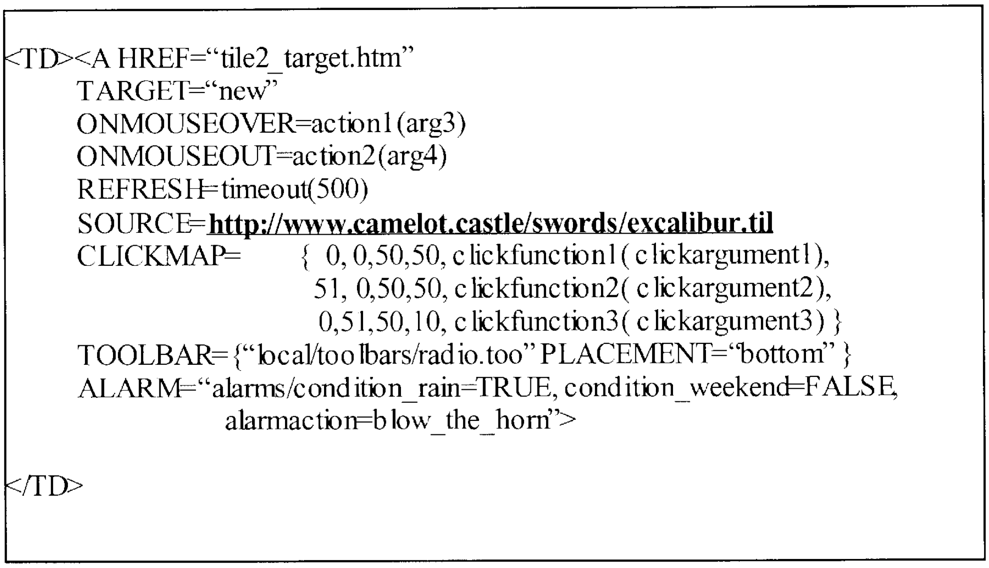

500
502
Tile Address
504
Target Address
506
Name
508
Initialisation function
510
Refresh function
512
Fill-screen function
514
Alarm function
516
Display option
518
On mouseout function
520
On mouseover function
522
Toolbar function

Fig. 5



```
<TD><A HREF="tile2_target.htm"
      TARGET="new"
      ONMOUSEOVER=action1(arg3)
      ONMOUSEOUT=action2(arg4)
      REFRESH=timeout(500)
      SOURCE=http://www.camelot.castle/swords/excalibur.til
      CLICKMAP=      { 0,0,50,50, clickfunction1( clickargument1),
                        51, 0,50,50, clickfunction2( clickargument2),
                        0,51,50,10, clickfunction3( clickargument3) }
      TOOLBAR={"local/toolbars/radio.too" PLACEMENT="bottom" }
      ALARM="alarms/condition_rain=TRUE, condition_weekend=FALSE,
                  alarmaction=blow_the_horn">

</TD>
```

Fig. 6

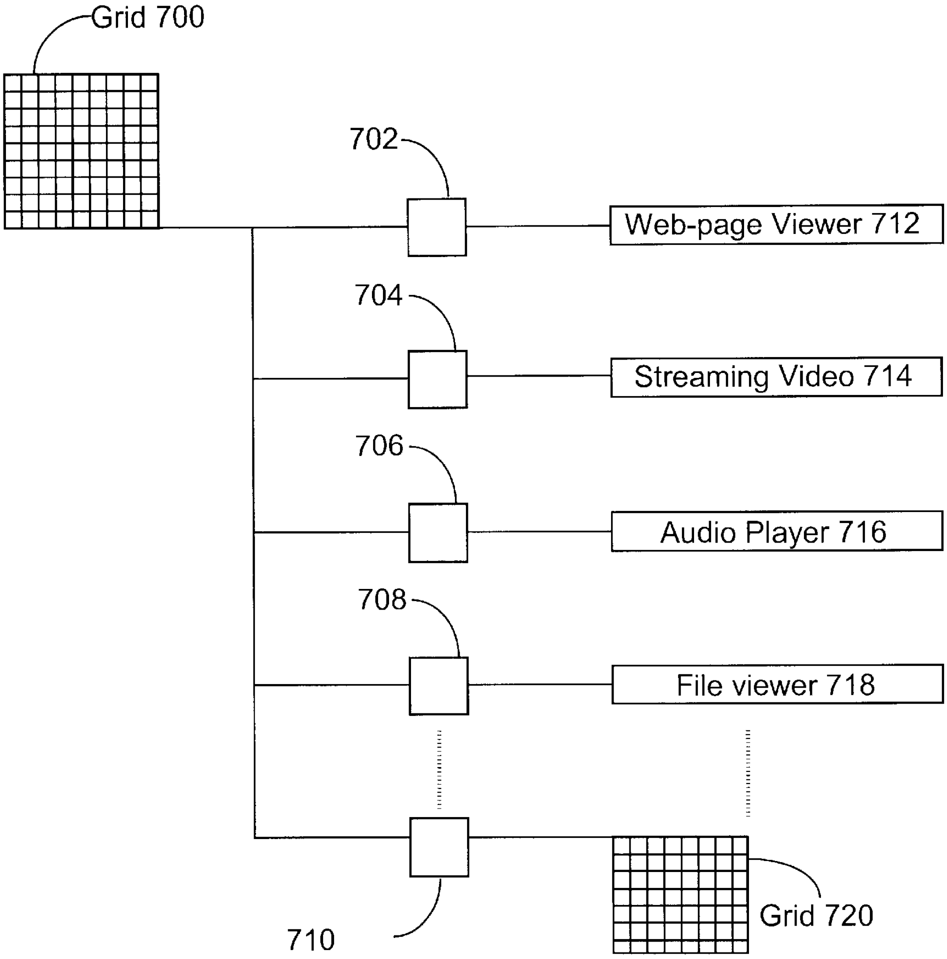

Grid 700
702
Web-page Viewer 712
704
Streaming Video 714
706
Audio Player 716
708
File viewer 718
710
Grid 720

Fig. 7

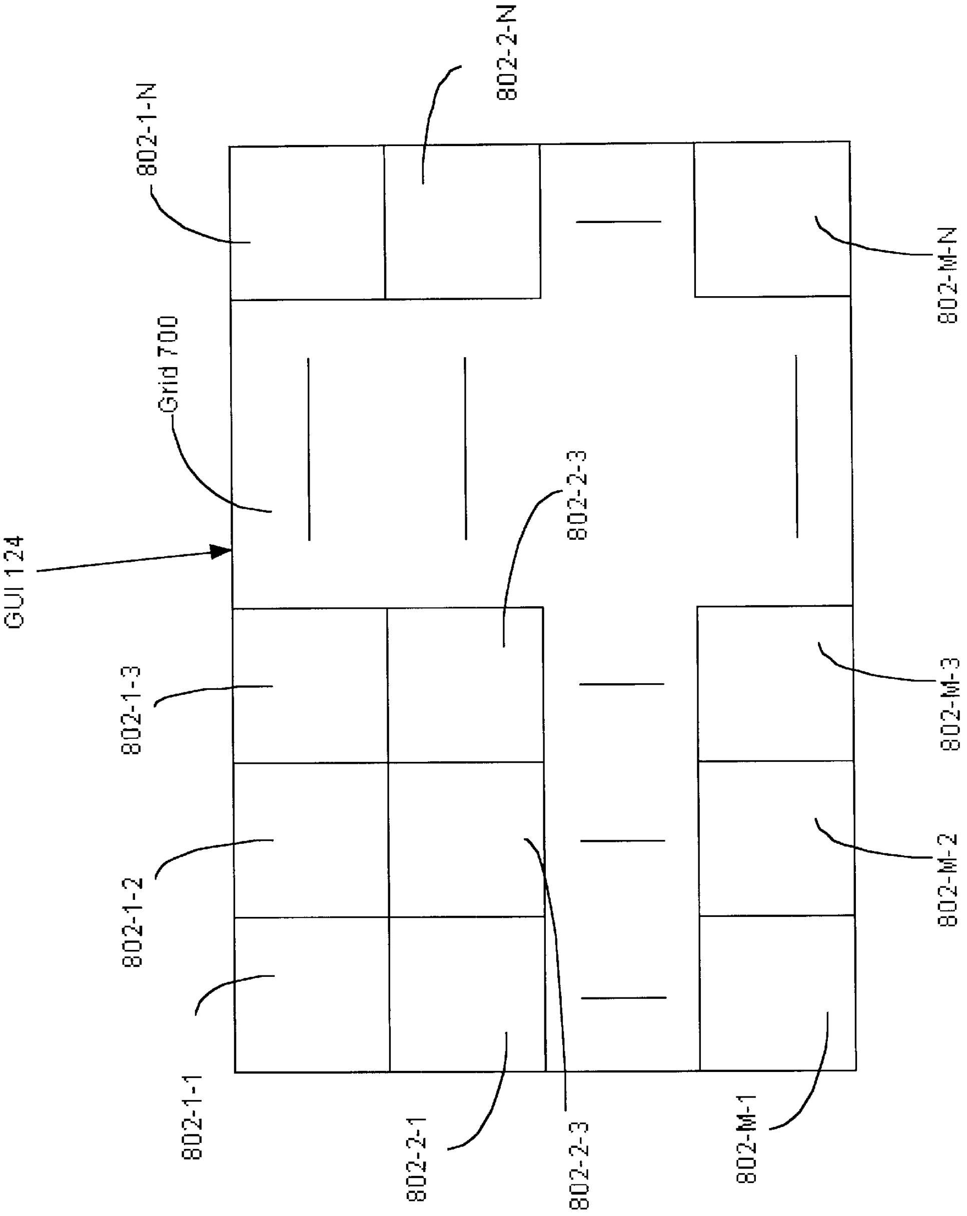

GUI 124
Grid 700
802-1-N
802-2-N
802-M-N
802-2-3
802-1-3
802-M-3
802-1-2
802-M-2
802-1-1
802-2-1
802-2-3
802-M-1

Fig. 8

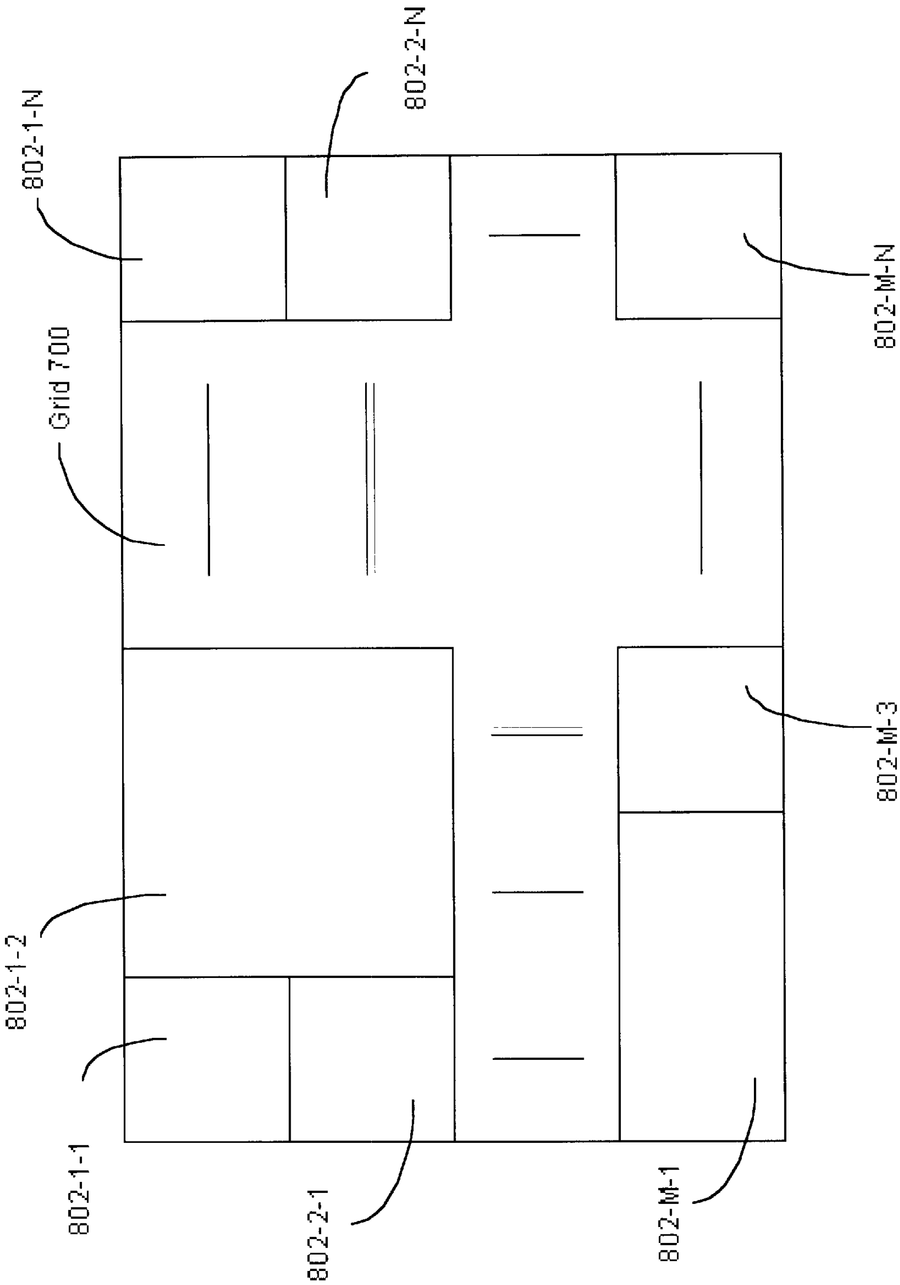


Fig. 9

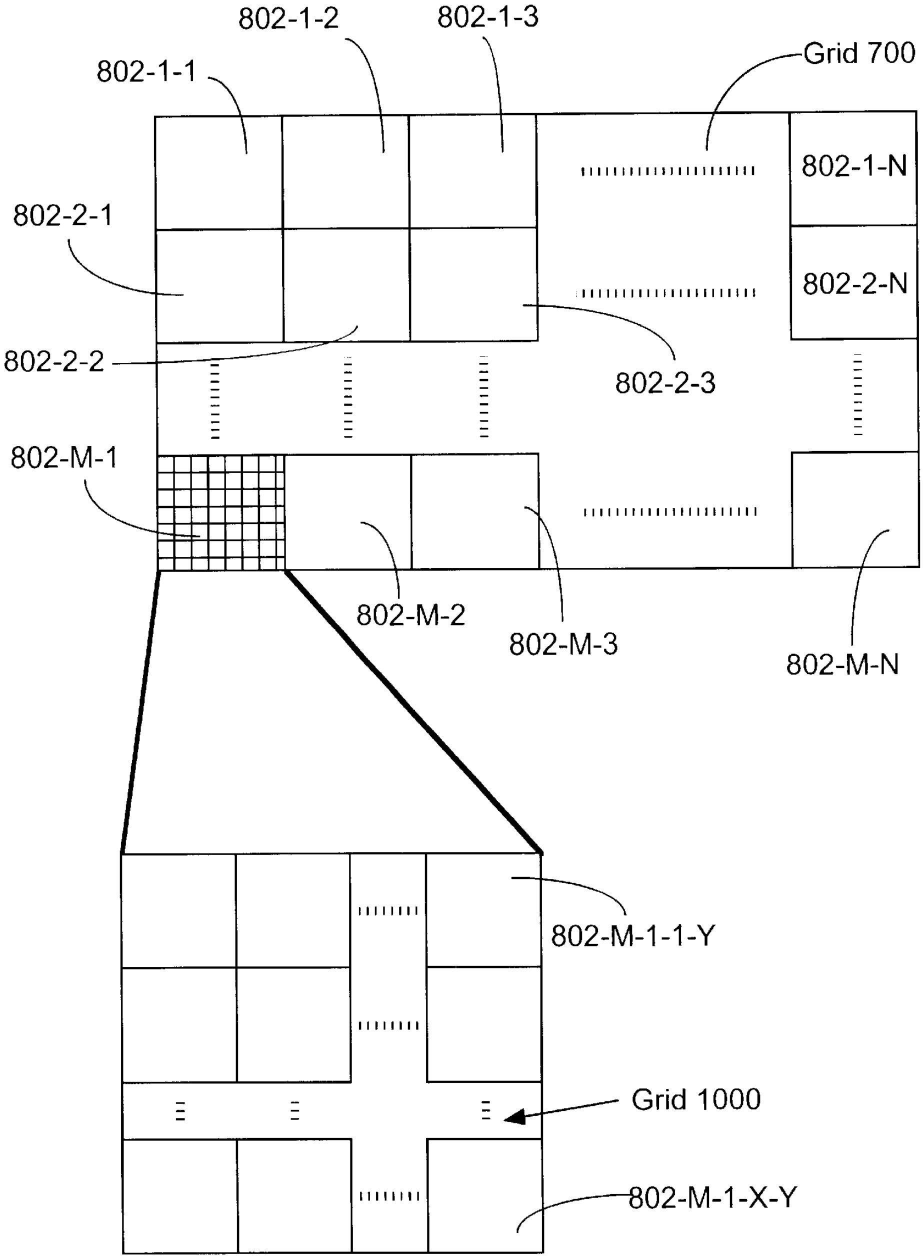

802-1-2
802-1-3
Grid 700
802-1-1
802-1-N
802-2-1
802-2-N
802-2-2
802-2-3
802-M-1
802-M-2
802-M-3
802-M-N
802-M-1-1-Y
Grid 1000
802-M-1-X-Y

Fig. 10

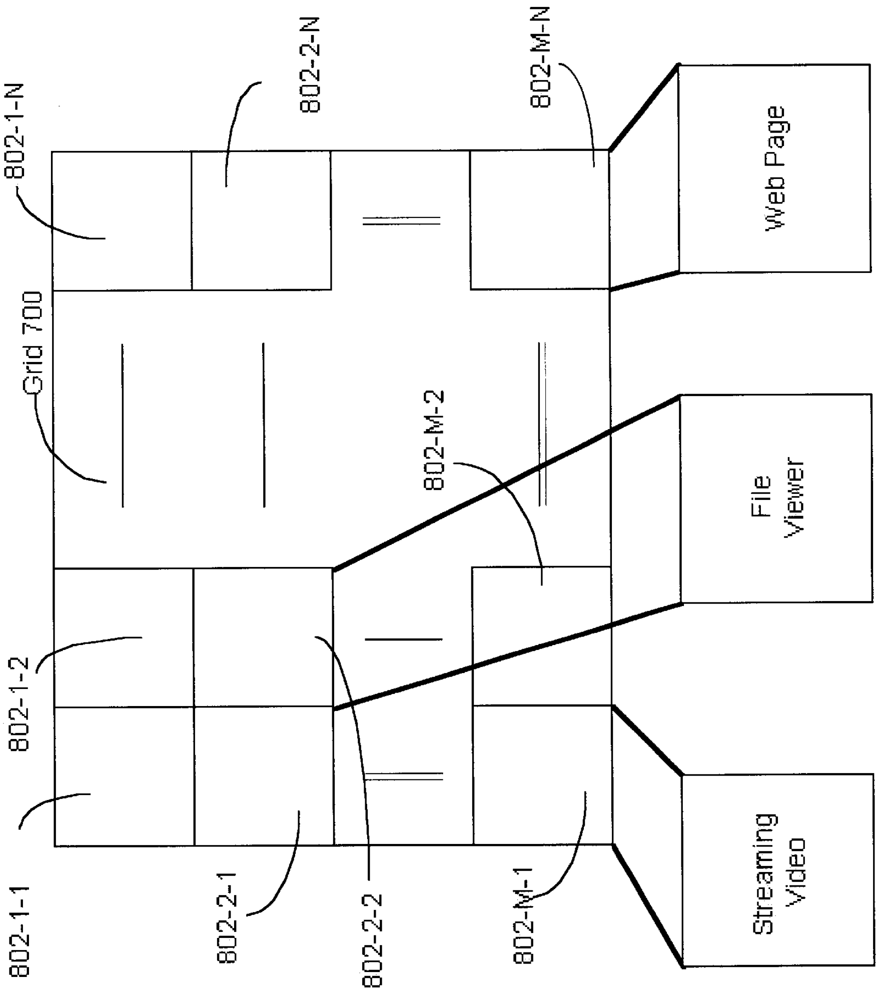
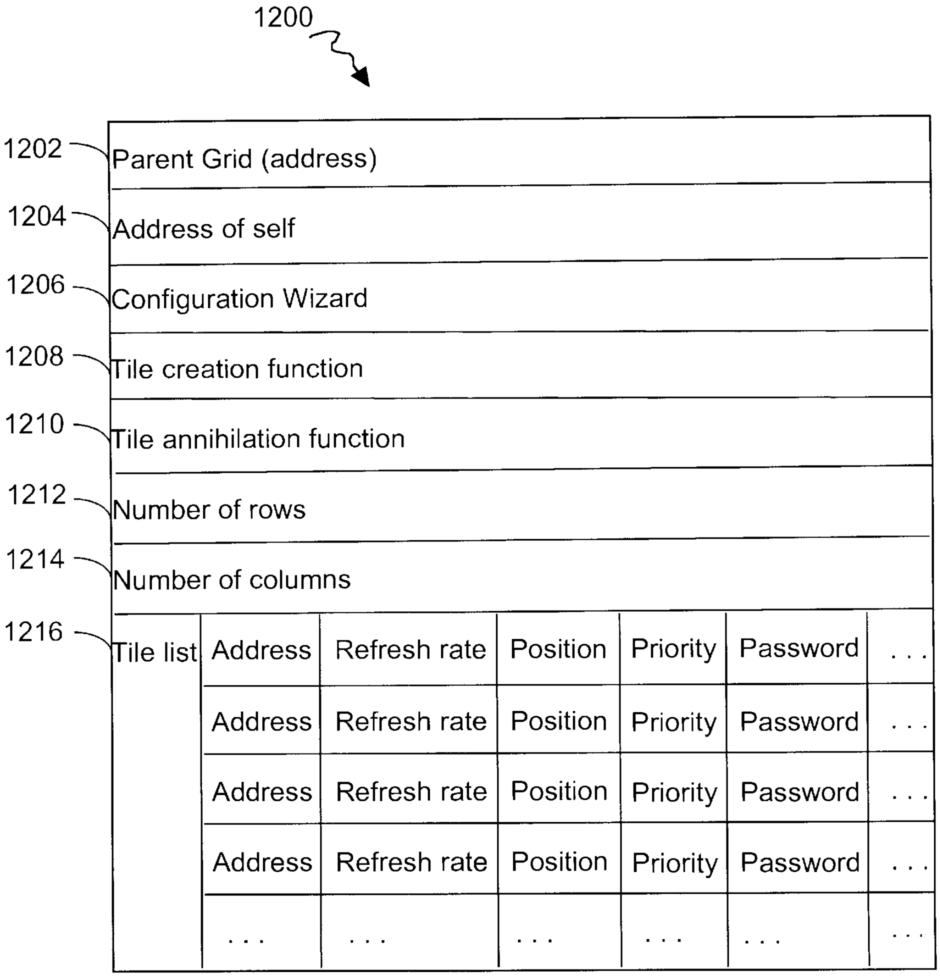
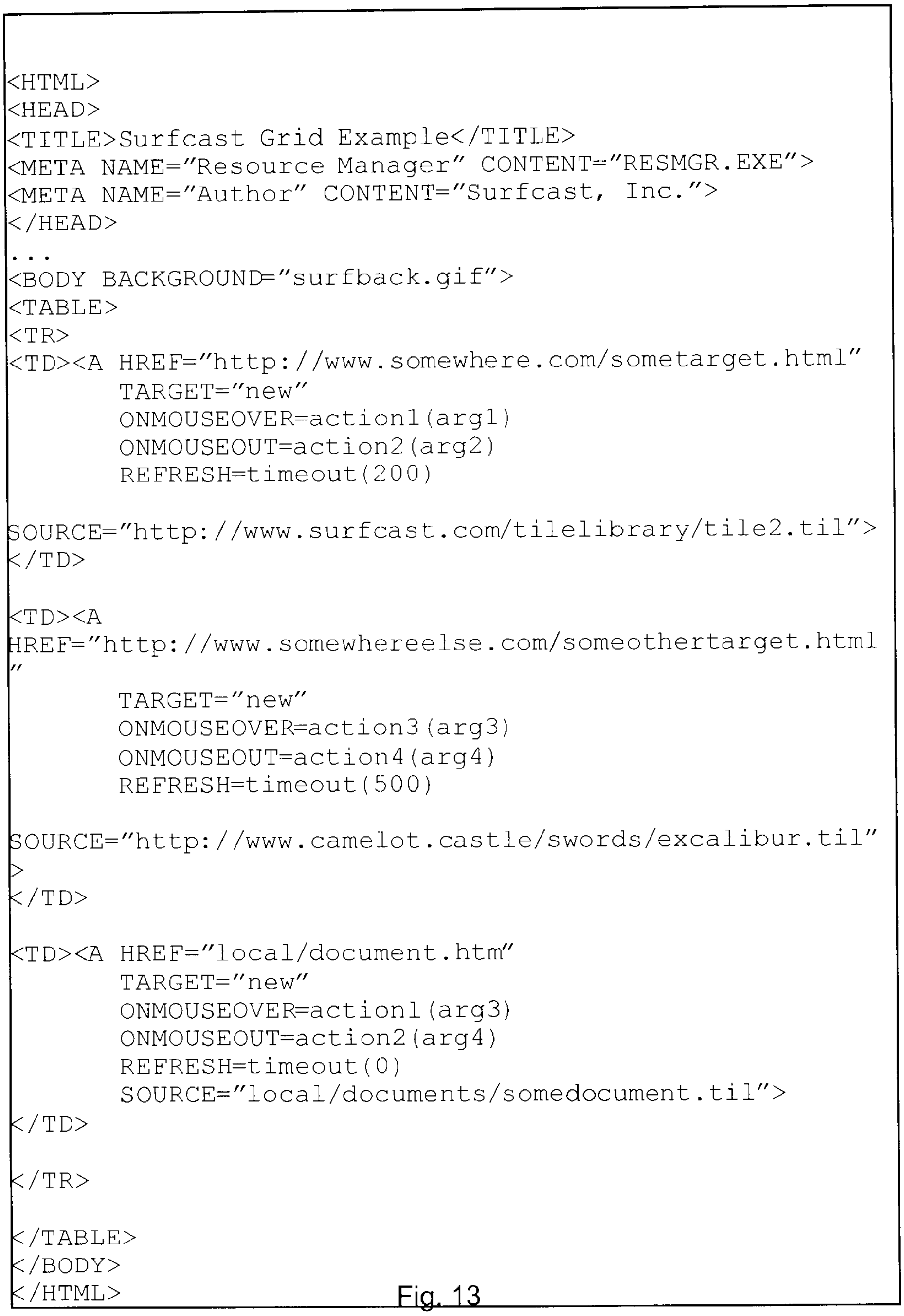

802-1-N
802-2-N
802-M-N
Grid 700
802-M-2
Web Page
File Viewer
802-1-2
802-1-1
802-2-1
802-2-2
802-M-1
Streaming Video

Fig. 11



Parent Grid (address)						
Address of self						
Configuration Wizard						
Tile creation function						
Tile annihilation function						
Number of rows						
Number of columns						
Tile list	Address	Refresh rate	Position	Priority	Password	. . .
	Address	Refresh rate	Position	Priority	Password	. . .
	Address	Refresh rate	Position	Priority	Password	. . .
	Address	Refresh rate	Position	Priority	Password	. . .
	. . .	. . .	. . .	. . .	. . .	. . .

Fig. 12



```
<HTML>
<HEAD>
<TITLE>Surfcast Grid Example</TITLE>
<META NAME=”Resource Manager” CONTENT=”RESMGR.EXE”>
<META NAME=”Author” CONTENT=”Surfcast, Inc.”>
</HEAD>
. . .
<BODY BACKGROUND=”surfback.gif”>
<TABLE>
<TR>
<TD><A HREF=”http://www.somewhere.com/sometarget.html”
       TARGET=”new”
       ONMOUSEOVER=action1(arg1)
       ONMOUSEOUT=action2(arg2)
       REFRESH=timeout(200)

SOURCE=”http://www.surfcast.com/tilelibrary/tile2.til”>
</TD>

<TD><A
HREF=”http://www.somewhereelse.com/someothertarget.html
”
       TARGET=”new”
       ONMOUSEOVER=action3(arg3)
       ONMOUSEOUT=action4(arg4)
       REFRESH=timeout(500)

SOURCE=”http://www.camelot.castle/swords/excalibur.til”
>
</TD>

<TD><A HREF=”local/document.htm”
       TARGET=”new”
       ONMOUSEOVER=action1(arg3)
       ONMOUSEOUT=action2(arg4)
       REFRESH=timeout(0)
       SOURCE=”local/documents/somedocument.til”>
</TD>

</TR>

</TABLE>
</BODY>
</HTML>
```

Fig. 13

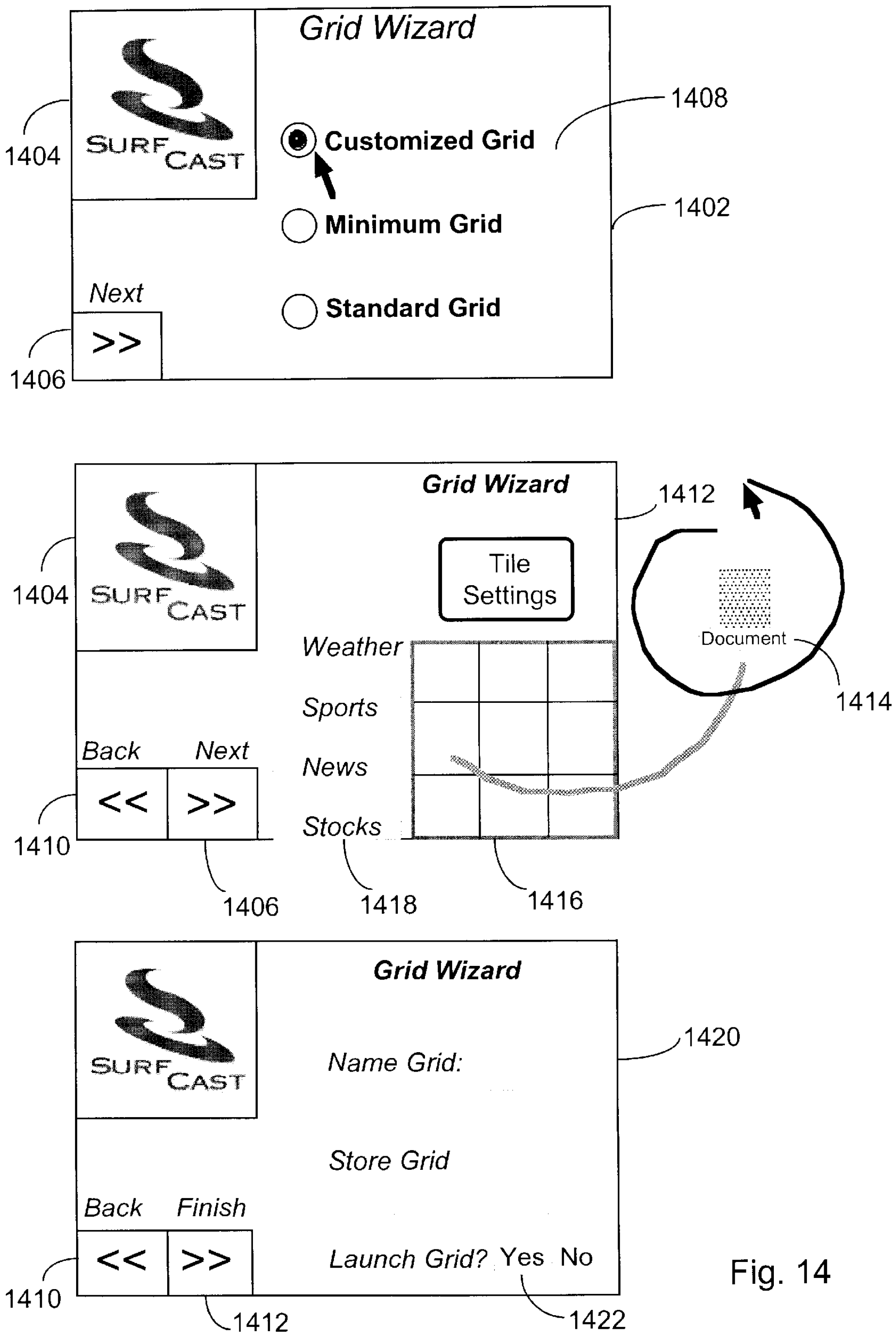

Grid Wizard
SURF CAST
● Customized Grid
○ Minimum Grid
○ Standard Grid
Next
>>
1404
1406
1408
1402
Grid Wizard
SURF CAST
Tile Settings
Weather
Sports
News
Stocks
Document
Back
Next
<<
>>
1404
1410
1406
1418
1416
1412
1414
Grid Wizard
SURF CAST
Name Grid:
Store Grid
Launch Grid? Yes No
Back
Finish
<<
>>
1420
1410
1412
1422

Fig. 14

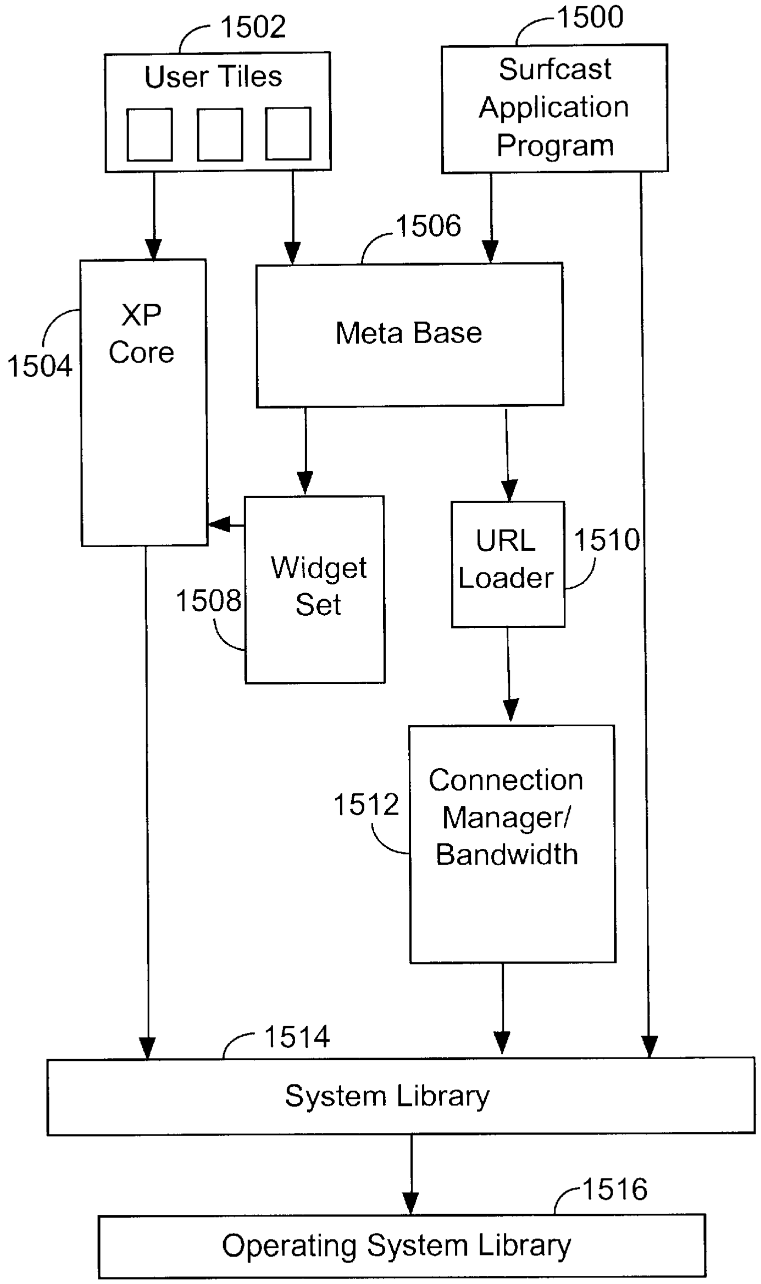

1502
User Tiles
1500
Surfcast
Application
Program
1506
XP
Core
1504
Meta Base
Widget
Set
1508
URL
Loader
1510
1512
Connection
Manager/
Bandwidth
1514
System Library
1516
Operating System Library

Fig. 15

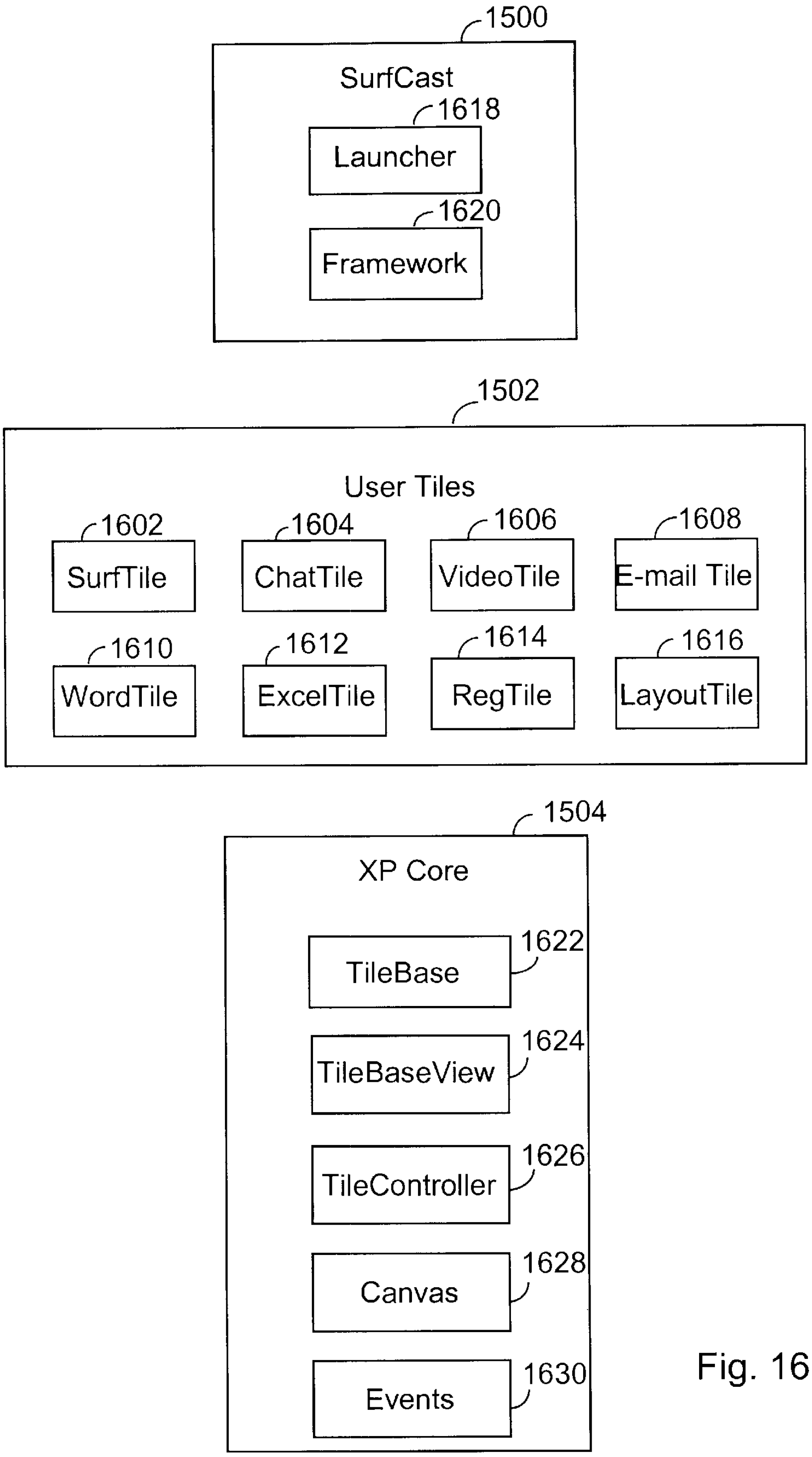

1500
SurfCast
1618
Launcher
1620
Framework
1502
User Tiles
1602
SurfTile
1604
ChatTile
1606
VideoTile
1608
E-mail Tile
1610
WordTile
1612
ExcelTile
1614
RegTile
1616
LayoutTile
1504
XP Core
TileBase
1622
TileBaseView
1624
TileController
1626
Canvas
1628
Events
1630

Fig. 16

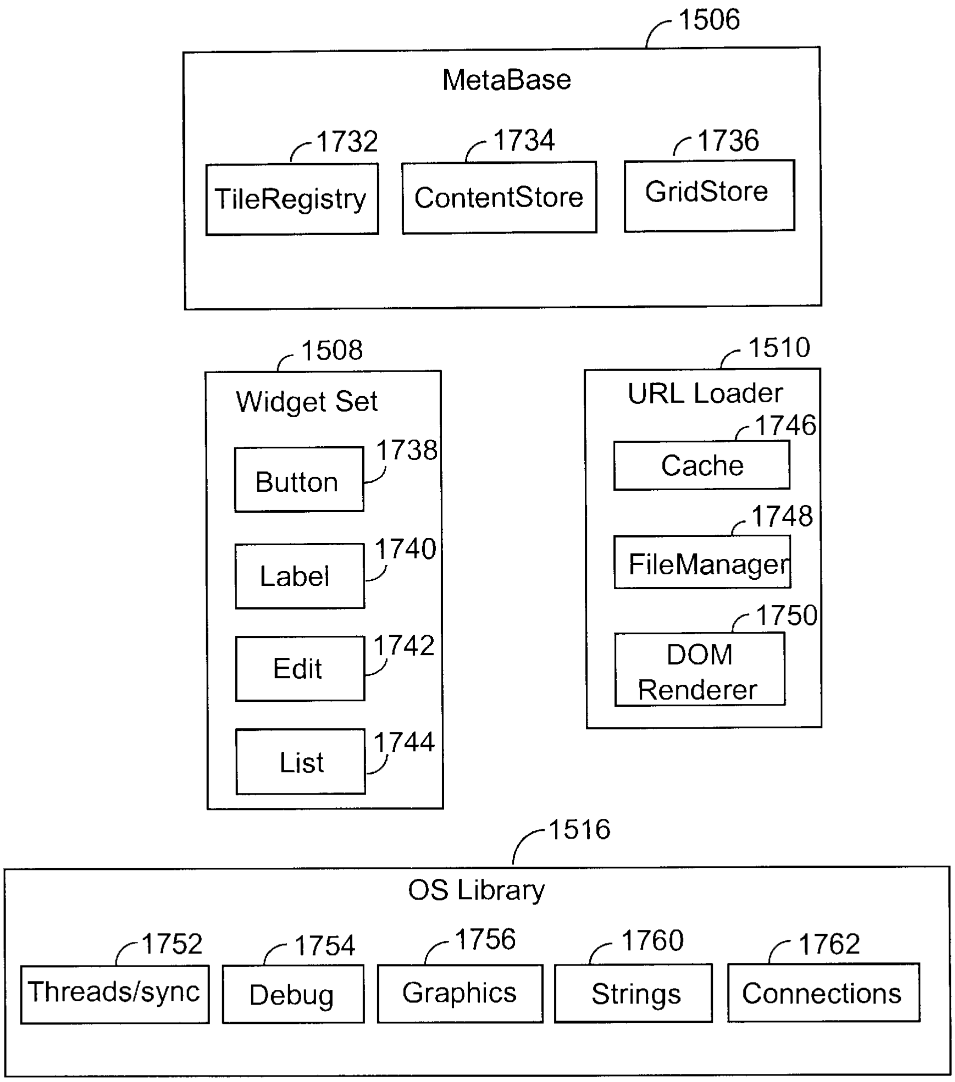

1506
MetaBase
1732
TileRegistry
1734
ContentStore
1736
GridStore
1508
Widget Set
1738
Button
1740
Label
1742
Edit
1744
List
1510
URL Loader
1746
Cache
1748
FileManager
1750
DOM Renderer
1516
OS Library
1752
Threads/sync
1754
Debug
1756
Graphics
1760
Strings
1762
Connections

Fig. 17

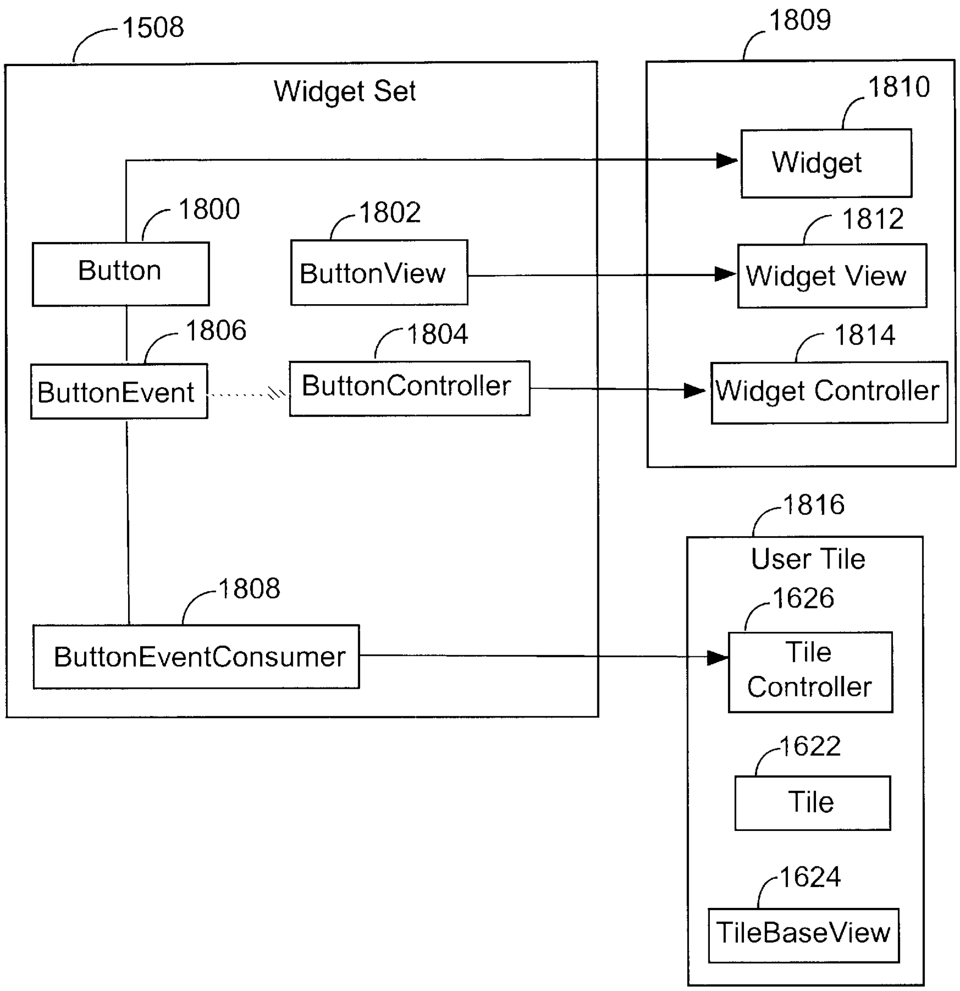

1508
Widget Set
1800
Button
1802
ButtonView
1806
ButtonEvent
1804
ButtonController
1808
ButtonEventConsumer
1809
1810
Widget
1812
Widget View
1814
Widget Controller
1816
User Tile
1626
Tile Controller
1622
Tile
1624
TileBaseView

Fig. 18

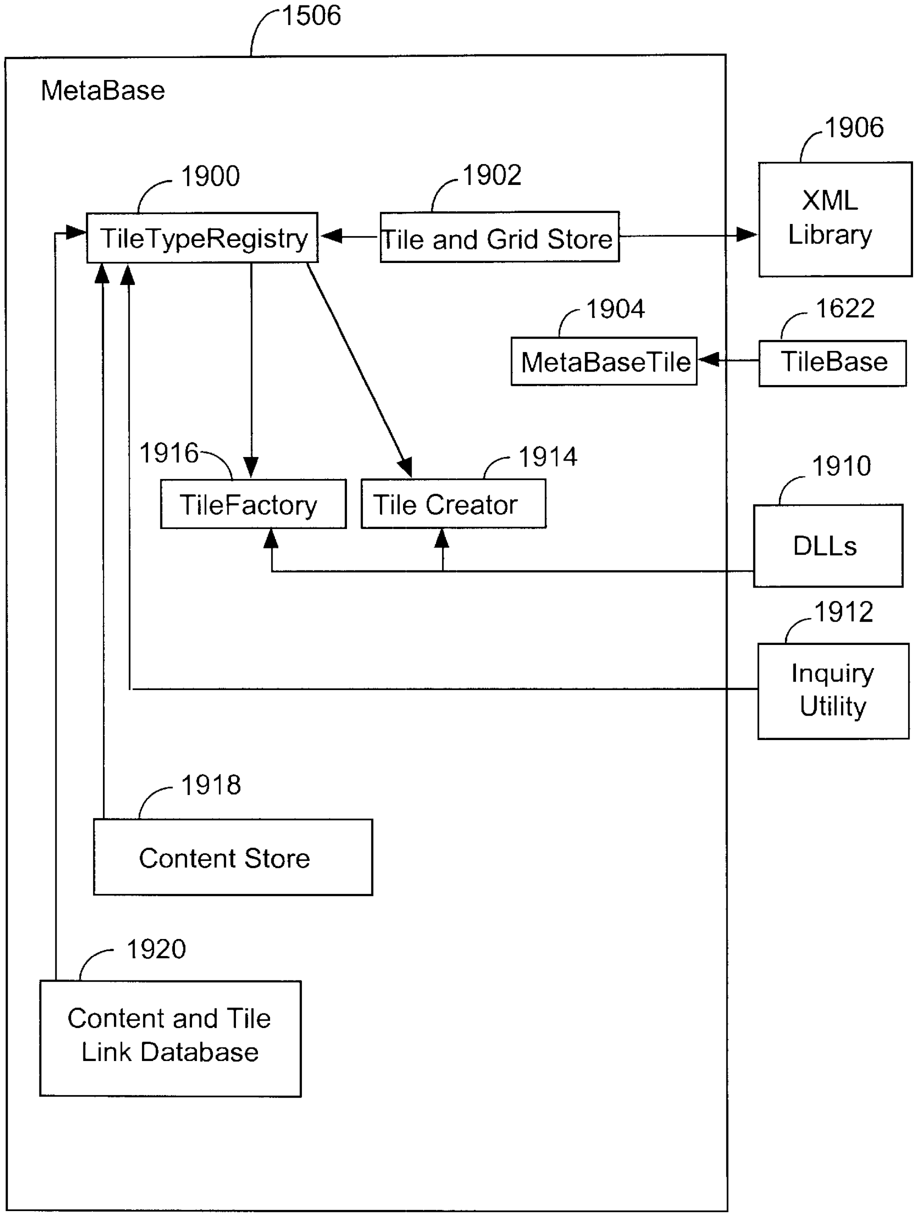

1506
MetaBase
1900
TileTypeRegistry
1902
Tile and Grid Store
1906
XML Library
1904
MetaBaseTile
1622
TileBase
1916
TileFactory
1914
Tile Creator
1910
DLLs
1912
Inquiry Utility
1918
Content Store
1920
Content and Tile Link Database

Fig. 19

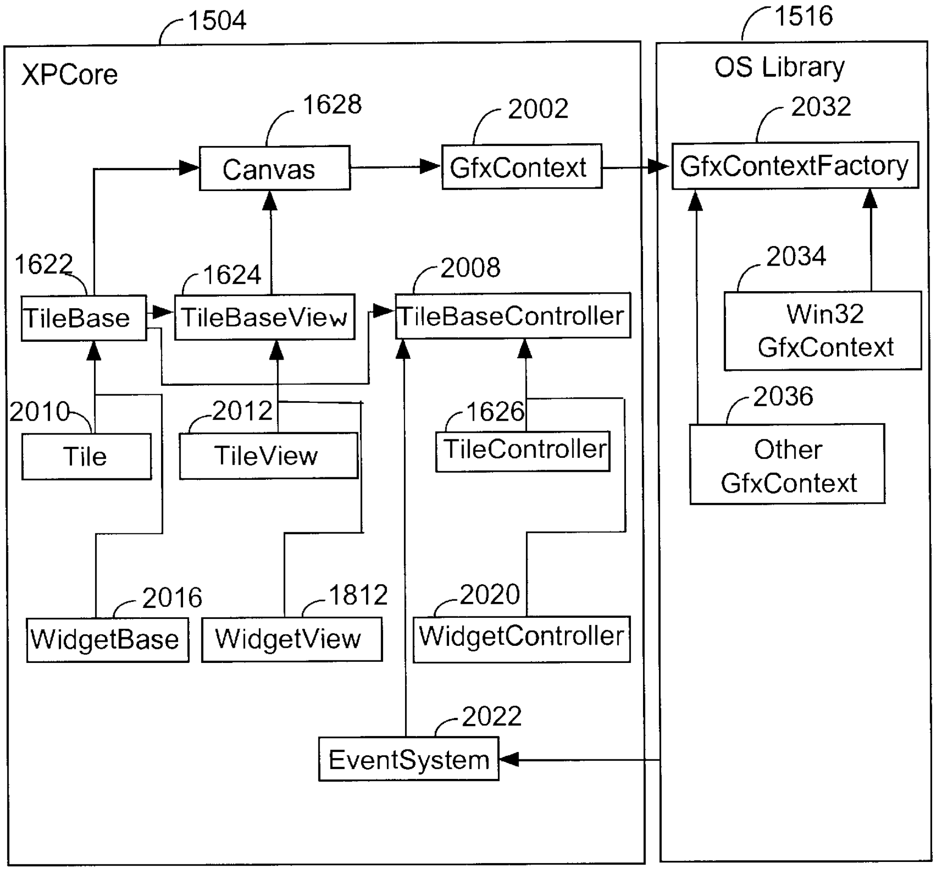

1504
XPCore
1628
Canvas
2002
GfxContext
1622
TileBase
1624
TileBaseView
2008
TileBaseController
2010
Tile
2012
TileView
1626
TileController
2016
WidgetBase
1812
WidgetView
2020
WidgetController
2022
EventSystem
1516
OS Library
2032
GfxContextFactory
2034
Win32
GfxContext
2036
Other
GfxContext

Fig. 20

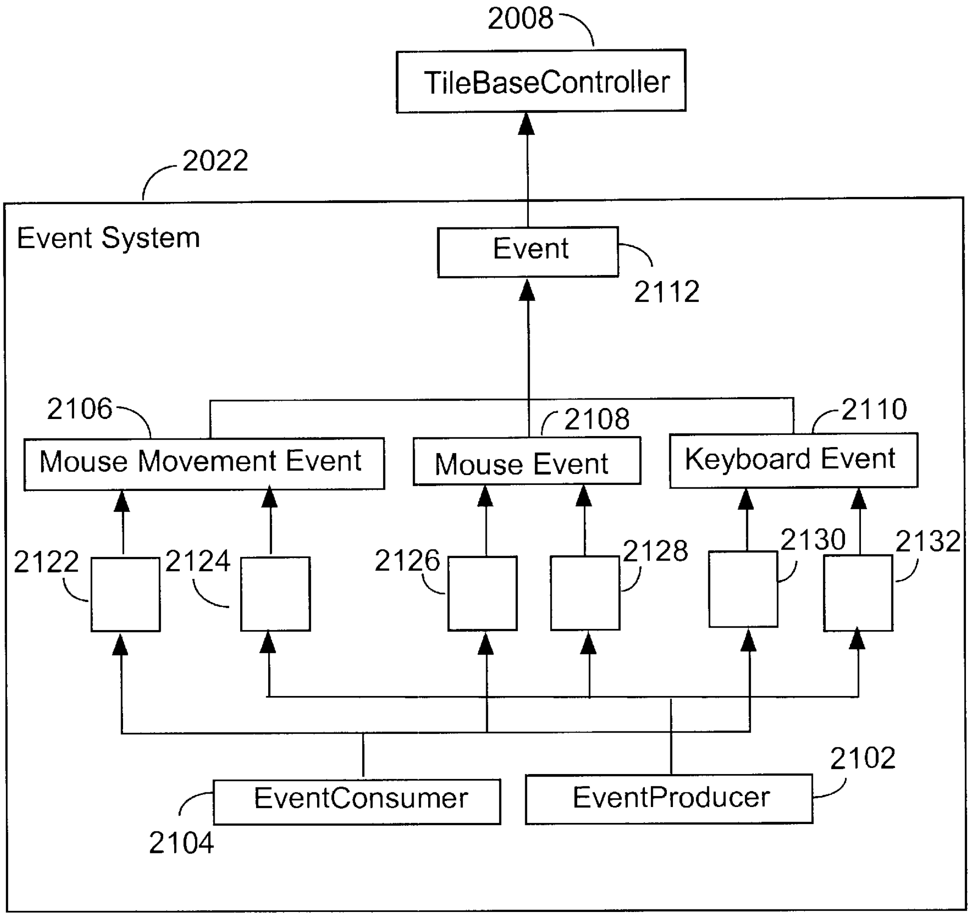

2008
TileBaseController
2022
Event System
Event
2112
2106
Mouse Movement Event
2108
Mouse Event
2110
Keyboard Event
2122
2124
2126
2128
2130
2132
EventConsumer
2104
EventProducer
2102

Fig. 21

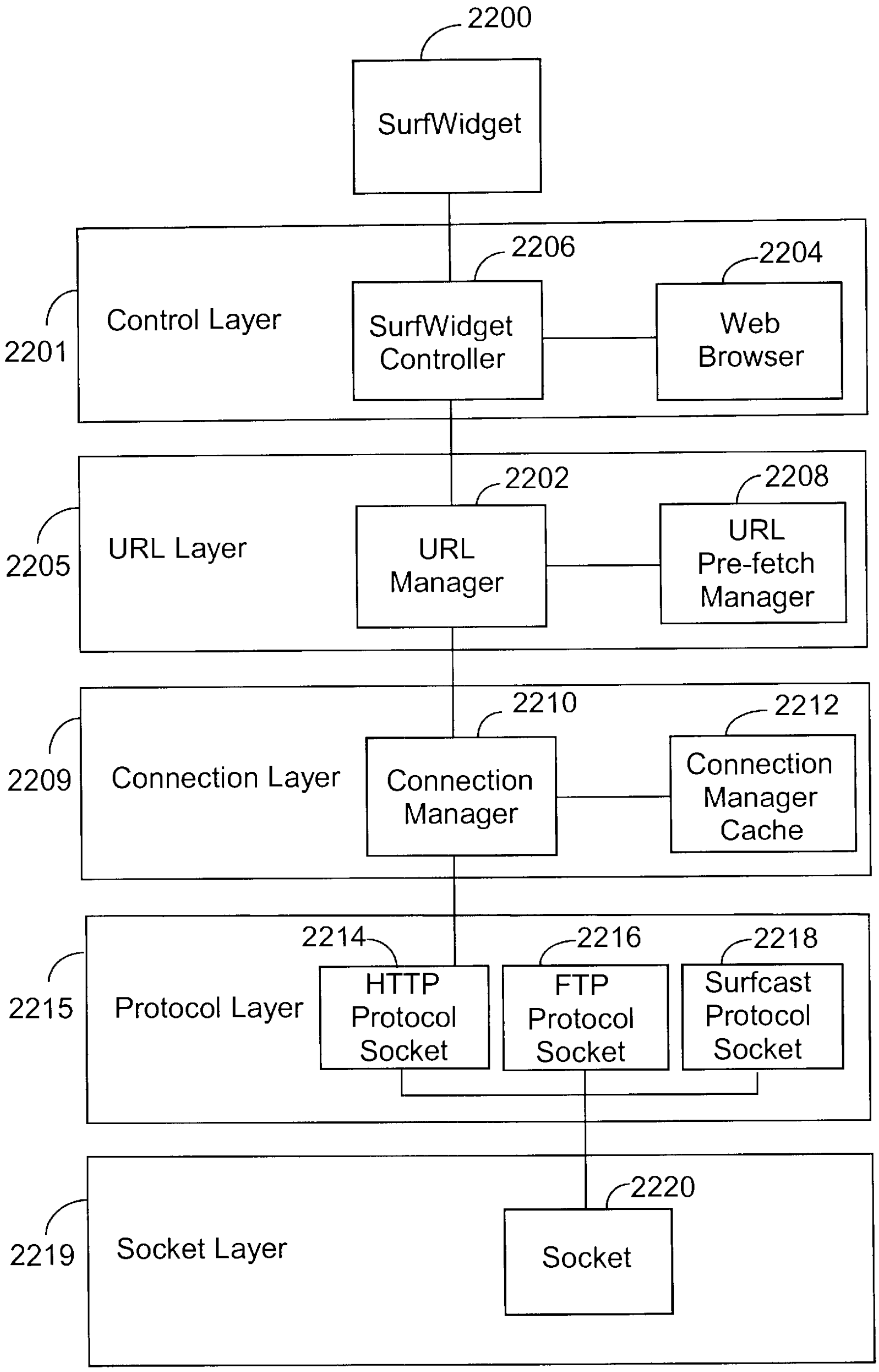

2200
SurfWidget
2201
Control Layer
2206
SurfWidget Controller
2204
Web Browser
2205
URL Layer
2202
URL Manager
2208
URL Pre-fetch Manager
2209
Connection Layer
2210
Connection Manager
2212
Connection Manager Cache
2215
Protocol Layer
2214
HTTP Protocol Socket
2216
FTP Protocol Socket
2218
Surfcast Protocol Socket
2219
Socket Layer
2220
Socket

Fig. 22

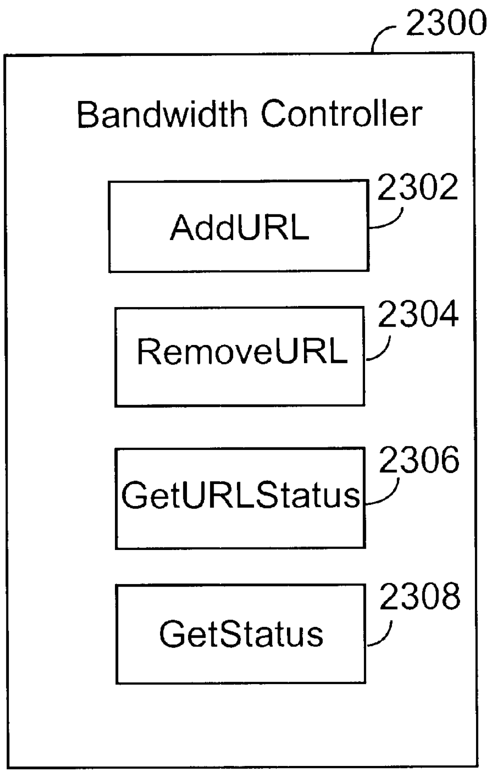

2300
Bandwidth Controller
AddURL
2302
RemoveURL
2304
GetURLStatus
2306
GetStatus
2308

Fig. 23

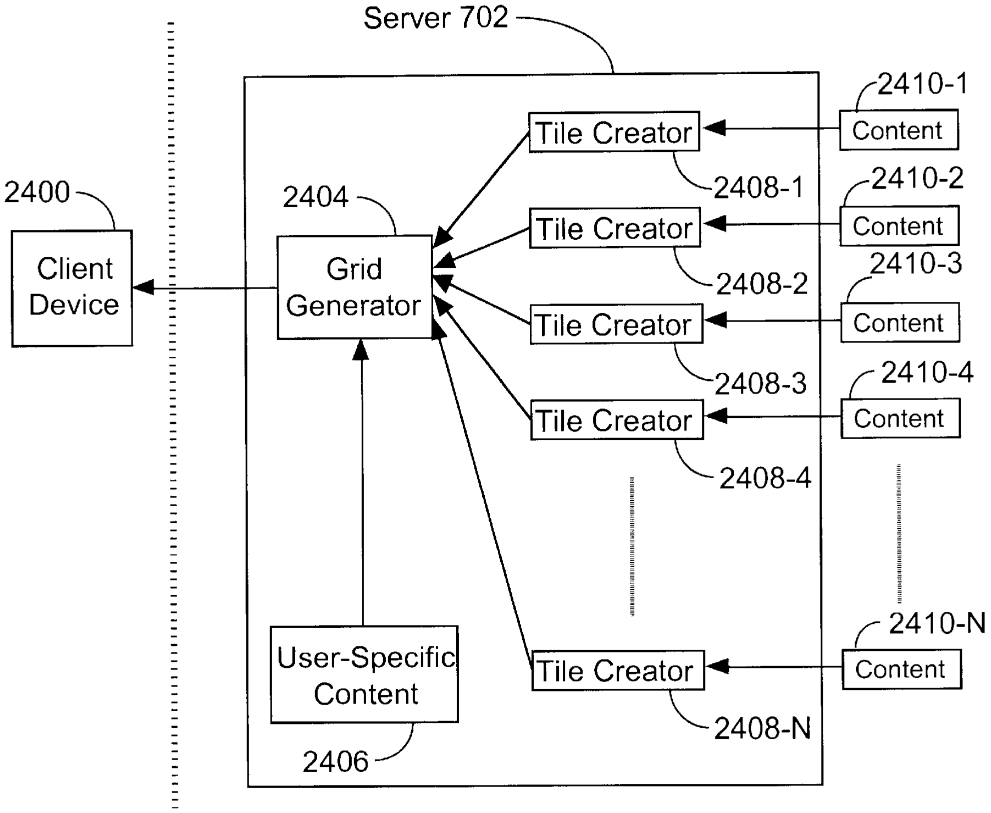

Server 702
2400
Client Device
2404
Grid Generator
User-Specific Content
2406
Tile Creator
2408-1
Tile Creator
2408-2
Tile Creator
2408-3
Tile Creator
2408-4
Tile Creator
2408-N
2410-1
Content
2410-2
Content
2410-3
Content
2410-4
Content
2410-N
Content

Fig. 24

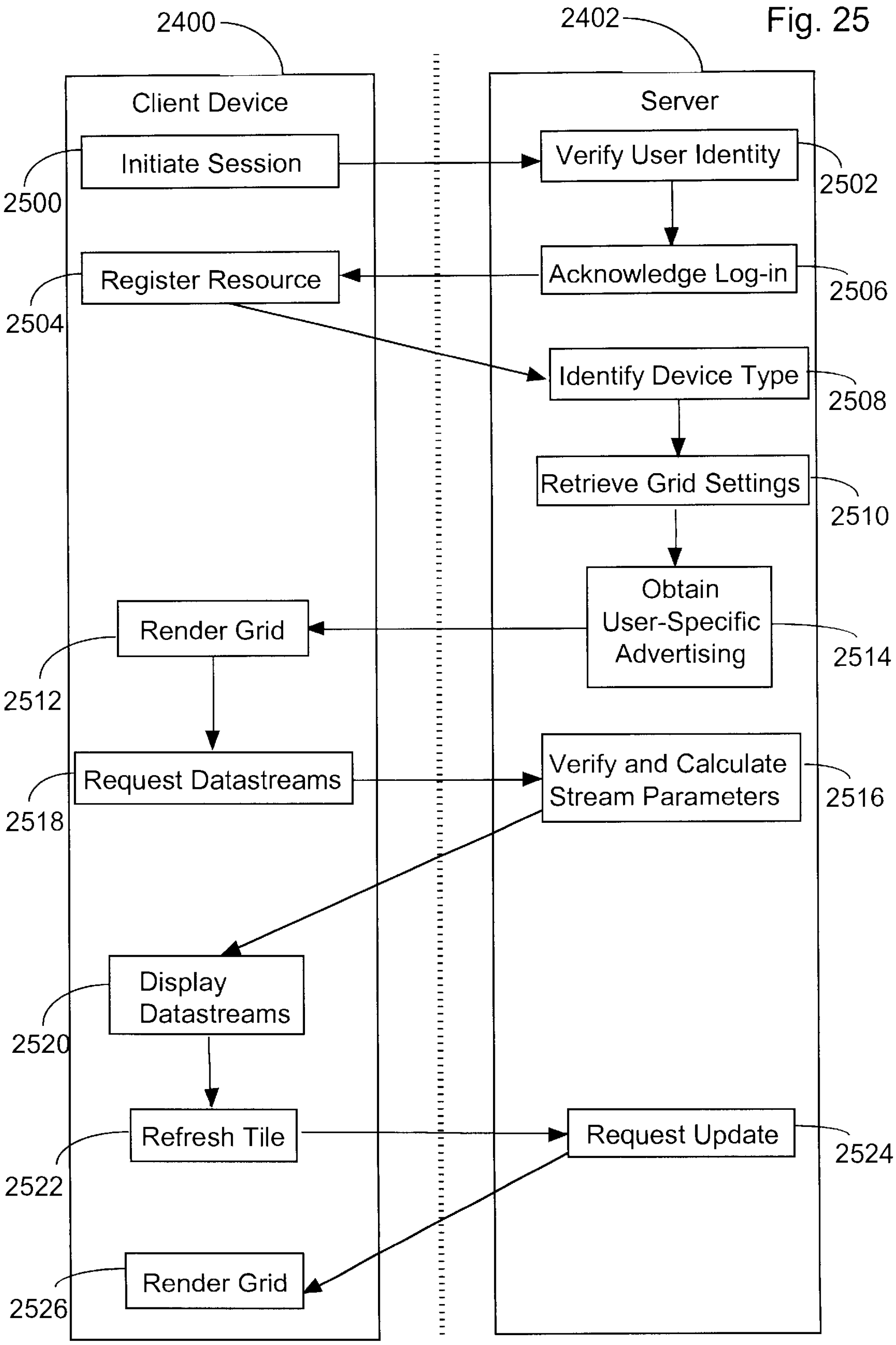


Fig. 25
2400
2402
Client Device
Server
Initiate Session
2500
Verify User Identity
2502
Register Resource
2504
Acknowledge Log-in
2506
Identify Device Type
2508
Retrieve Grid Settings
2510
Render Grid
2512
Obtain User-Specific Advertising
2514
Request Datastreams
2518
Verify and Calculate Stream Parameters
2516
Display Datastreams
2520
Refresh Tile
2522
Request Update
2524
Render Grid
2526

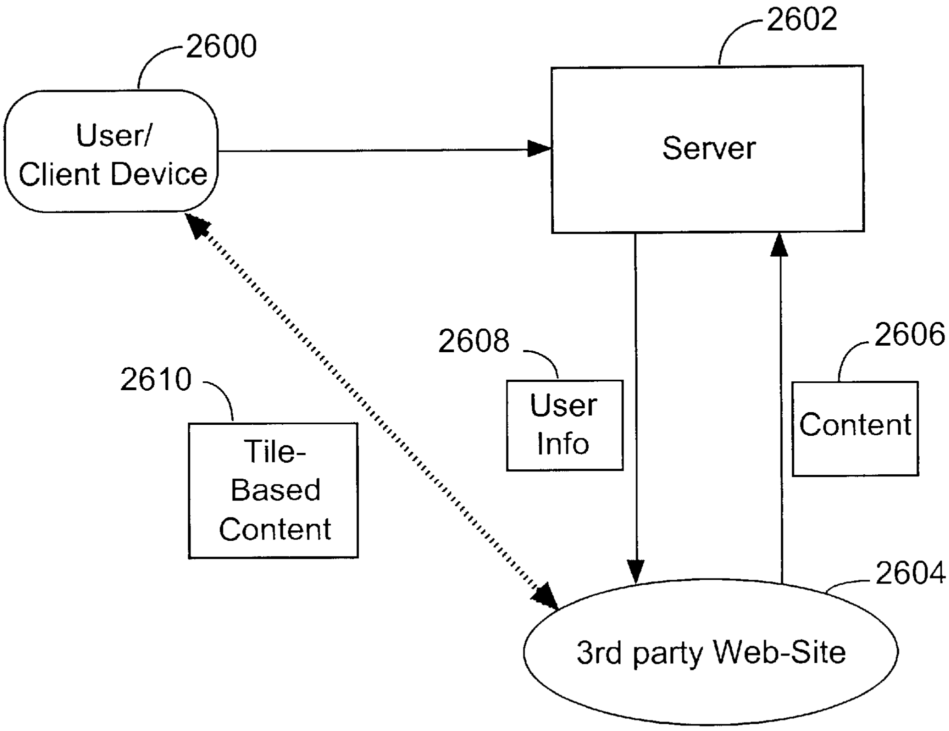

2600
User/
Client Device
2602
Server
2608
User
Info
2606
Content
2610
Tile-
Based
Content
2604
3rd party Web-Site

Fig. 26

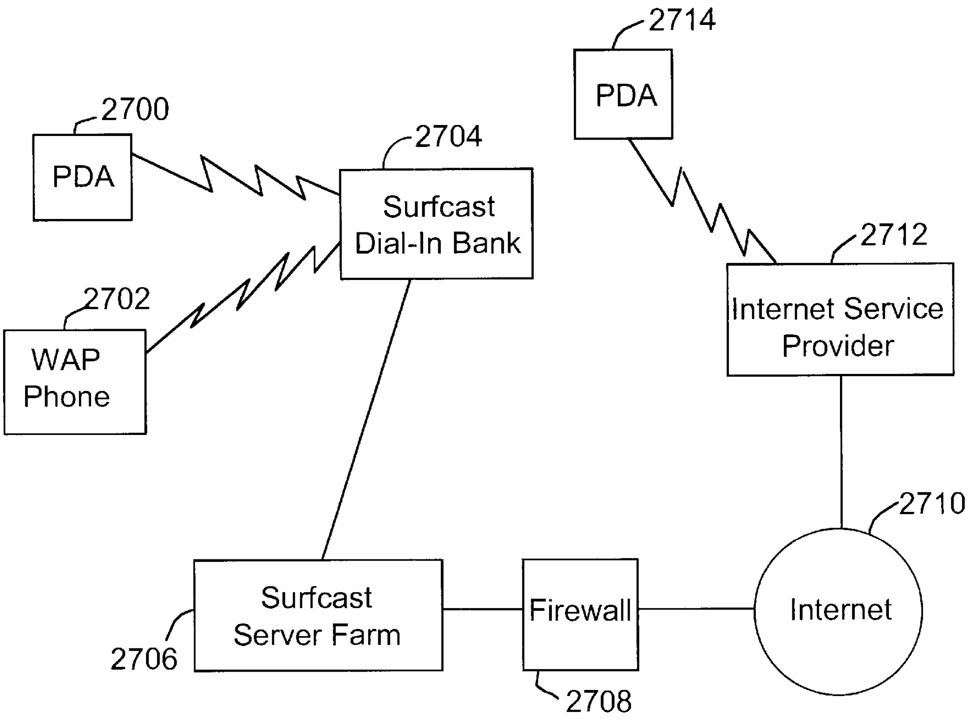

2714
PDA
2700
PDA
2704
Surfcast Dial-In Bank
2712
Internet Service Provider
2702
WAP Phone
2710
Surfcast Server Farm
2706
Firewall
2708
Internet

Fig. 27

SYSTEM AND METHOD FOR SIMULTANEOUS DISPLAY OF MULTIPLE INFORMATION SOURCES

This application claims priority to provisional patent application entitled "System and Method For Simultaneous Display of Multiple Datastreams", Ser. No. 60/162,522, filed Oct. 29, 1999.

FIELD OF THE INVENTION

The present invention relates to methods of presenting information from a variety of sources on a display device. Specifically the present invention describes a graphical user interface for organizing simultaneous display of information from a multitude of sources.

BACKGROUND OF THE INVENTION

The scope of the global communications capacity, comprising fixed link and wireless networks, continues to expand rapidly. The variety and complexity of communication devices proliferates and the number of users escalates. As a result, users are faced with increasingly complex systems and interfaces with which to manage multiple sources of information. At the same time, society has increased its demands on time and productivity so that users no longer have the luxury of focusing their attention on a single source of information or means of communication. Instead, the norm today is for people to carry out many tasks simultaneously.

As might be expected, these demands have exposed substantial problems in current communications technology. In particular, users are faced with insufficient resources to manage and access the volume and variety of information available to them in an efficient and productive manner. While a variety of tools designed to assist in accessing and managing these resources have been created, these tools remain unsatisfactory. Consequently, users are impeded by the myriad of information sources, each with its own method of use and often with its own login and password requirements, as well as by slow retrieval times to access the information. The result is an unacceptable delay for many operations.

Under the present art, for example, it is usually the case that a user lacks the bandwidth resources to receive multiple video signals simultaneously. If an individual were receiving one video signal, it is usually impractical to receive a second at the same time due to bandwidth constraints. Thus, the user could not, for example, monitor multiple video data streams of sporting or news events; instead, the user could monitor only one video data stream at a time.

To address such bandwidth resource limitations, the current art only accesses information when the user requests it. As a result, there is an inevitable delay between the user's request for information and the communications device's presentation of it. For example, if a user wants to monitor sources of news information on the Internet using current browser technology, the user must continuously and manually request the news data from its source to determine whether the data has been updated. Prior to requesting and subsequently receiving the data, the user has no way of knowing whether the data has been updated. In any case, the user is unlikely to want to refresh the status of each application by manual intervention himself at the frequency necessary to ensure that the information is up to date. Additionally, if a user wishes to view two or more webpages simultaneously, he must run two or more copies of the web-browser program. The act of manually refreshing the content of alternate programs in order to ascertain which have any new material to offer is fundamentally inefficient.

Similarly, the user's access to such data is not in real-time or even near real-time because each time the user wants to view the information, he must request it from its source and wait for the source to transmit it to him. Thereafter, he must wait until his communications device has received and processed the information before it is presented. For complex information such as a video signal, this can take longer than a minute to occur; and, even for simple information, this process can take many seconds. Thus, the user is denied real-time or near real-time access to the information.

Present technology that locally stores or "caches" previously accessed information to make it available to the user more rapidly does not solve this problem, because the cached information is necessarily old. The user's communications device must still verify the accuracy of the information with the source before the system displays the cached information. As a result, the user is denied real-time or near real-time access to updated information.

Similarly, if a user wishes to make two or more simultaneous downloads there is no control over the relative rates at which the respective downloads would occur. So-called "push technologies" attempt to address this problem by organizing information from a number of related sources and sending it periodically to a user. While this arrangement frees a user from actively participating in the download, the price is that the user has little control over the organization of the information and can only practically handle a small number of such transmissions at any one time. Each transmission is subject to the bandwidth available.

Of course, not all tasks require the same allocation of resources and, correspondingly, not all tasks have equal priorities for a given user. In particular, a user may wish to customize the information environment in such a way that many processes are occurring synchronously, yet each is communicating with the user at a rate that is acceptable. For example, a television viewer may wish to know what is being broadcast on several channels at the same time but only care to watch one of them closely. An Internet user may wish to be continually in touch with sources of data from audio, video, chat-room, video-conferencing and e-mail checker utilities, but not wish all of them to update at the same frequency; the user would be satisfied merely to see at a glance a recent status of each. Some of these processes, such as chat-room activities entail very little data transmission and can, indeed, be effectively updated on a continuous basis, whereas others require a great deal of bandwidth but could usefully be sampled at a lower rate. The current art lacks any technology for controlling the respective refresh rates of several simultaneous information sources.

At the same time that users are limited by system resources, they are also finding that they have no effective way of managing the multiplicity of available data types and information sources. It is difficult both to conduct two or more different types of computing activities at the same time or to monitor two or more different information sources simultaneously because the tools available are confusing, inflexible, and/or otherwise difficult to implement. Users require immediate access to a wide variety of up to date content presented in a flexible, easily customized interface.

In addition to restrictions in the capacity of today's networks, there is very little conformity amongst the information content. A typical communication device, such as a personal computer, television or mobile telephone, comprises a display unit connected to a processing unit that can accept information from many different sources. As described above, the signals, data and/or datastreams that are available to such a device are diverse, including, for example, HTML content, e-mail, or streaming audio and video. Correspondingly, the software tools that interpret and process the different information sources present each in a different way to the user. From a user's perspective, distinctions between the different types of information could usefully be removed so that each is viewed in a similar way and such that the current presentation associated with any information source gives an immediate indication of its current content. The present reality is different, however. The user must contend with a wide range of icons and program windows that may occupy space on a user's display screen. Another lack of conformity is the different mode of behavior for programs that address different types of information. An effort to standardize the ways in which different types of information are presented to the user would be advantageous. Equally, unification of the way in which those types of information are managed would save time and increase user productivity, for productivity is reduced when users must cope with different attributes of different programs and learn distinct paradigms for different types of information.

The nature of the application program windows and their respective icons predominantly found on today's computer displays is restrictive. The application window typically displays the current content or output of only a single program and program icons convey nothing of the program's current state or content. Often, an icon is a static image which is merely characteristic of the program or data represented thereby rather than the program's current state or its information content. In the present art, there is no intermediate between a window or an icon.

Thus, while a window may be resized as appropriate, it will frequently occupy the full display area, effectively limiting the user to a view of a single program. It may have active areas around its borders such as menu bars, scroll bars, or tool bars designed to allow the user to control aspects of the window's appearance or to set parameters specific to the operation of the program controlling it. Icons, in contrast, offer ease of display when multiple programs are active, but they do not permit viewing or control of the underlying program or data represented thereby. Instead, icons require user intervention, typically in the form of a mouse-click on an icon of interest, to view or control the program or information. Consequently, the user's viewing options are limited to a choice between one presenting very limited information about a multitude of programs and information and one presenting full information, but of only a single program or data source.

The fact that the GUI's of the present art are largely restricted to icons and windows diminishes the capacity to organize, manage, and access available information. With the Internet representing an ever expanding view of currently accessible global information, the need for flexible information management tools has become crucial. Similarly, with the current expansion of television programming available, for example, through cable television and satellite broadcasting, the need to manage this audiovisual content becomes cute. The convergence of television programming and computers increases these management needs all the more.

Current computer operating system software utilizes bookmarking schemes for managing Internet locations and complex database technologies for managing specialist information. Neither provides visual immediacy or ease of layout. Bookmark hierarchies are presented as cascading textual menus and database technologies arrange information into rigidly defined structures. The missing capability is a visual categorization in which an area of the display unit itself becomes the bookmark and the arrangement on the display becomes the categorization, independent of the type of content.

While the most common way of accessing information sources is via a personal computer, present day technology exists to communicate via a television, handheld computing device, or even mobile telephones, in which case Internet content and other data can be displayed as some portion of the screen. There is a growing convergence of technologies: televisions are beginning to find application as viewers of non-television data, (for example through use of "Vertical Blanking Interval" technology in which a signal is inserted into the main video signal or through set-top boxes providing limited computer and communications functionality); computers are already finding application for the display of movies, real-time data streams, and the playing of audio data; handheld computing devices and mobile telephones are also being enabled to access the Internet and other information sources.

To summarize the current state of the art, display technologies currently lack an interface which is capable of organizing any type of information, presenting such information to the user in a consistent manner and in such a way that all currently open channels are able to indicate their activity on a continual basis and which could run on any device.

SUMMARY OF THE INVENTION

Accordingly, the present invention provides an easy to use graphical interface that facilitates the organization and management of multiple data sources corresponding to a user's needs and interests. The present invention comprises a grid of tiles that resides on the user's computer desktop. The grid of tiles provides a uniform, graphical environment in which a user can access, operate, and/or control multiple data sources on electronic devices. The graphical environment is uniform with respect to the source of accessed information and can manage multiple streams of content, entirely of the user's choice. For example, the invention presents video clips, e-mail messages, television shows, Internet sites, application programs, data files and folders, live video streams, music, radio shows, and any other form of analog signal, digital data or electronically stored information, to the user uniformly and simultaneously, regardless of whether the information is stored locally or available via modem, T1 line, infrared, or any other form of communication. The user's impression of the interface is also independent of the type of electronic device upon which it is implemented.

The present invention comprises a method executed by a computer under the control of a program stored in computer memory, said method comprising the steps of: partitioning a visual display of a computer into an array of tiles in a non-overlapping configuration; assigning a first refresh rate to a first tile of said array of tiles and a second refresh rate to a second tile of said array of tiles; updating information presented to said first tile in accordance with said first refresh rate; and updating information presented to said second tile in accordance with said second refresh rate.

The present invention additionally includes an electronic readable memory to direct an electronic device to function in a specified manner, comprising: a first set of instructions to control simultaneous communication with a plurality of datastreams: a second set of instructions to partition a display into an array of tiles; a third set of instructions to associate a first datastream of said plurality of datastreams to a first tile of said array of tiles and a second datastream of said plurality of datastreams to a second tile of said array of tiles; a fourth set of instructions to retrieve data from said first datastream in accordance with a first retrieval rate and retrieve data from said second datastream in accordance with a second retrieval rate; and a fifth set of instructions to present data to said first tile in accordance with said first retrieval rate and present data to said second tile in accordance with said second retrieval rate.

The application program of the present invention runs on many different devices, including, but not limited to set-top box, personal computer and hand-held device. The grid and tiles retain the same characteristics, regardless of operating device. For example, the tiles remain individually configurable and can offer near real-time views of their data content. The application therefore permits the user's interaction with a range of electronic devices to be unified.

BRIEF DESCRIPTION OF THE DRAWINGS

Additional objects and features of the invention will be more readily apparent from the following detailed description and appended claims when taken in conjunction with the drawings, in which:

FIG. **1** shows a representative embodiment of the user interface of the present invention comprising a grid of tiles as might be depicted on a display screen.

FIG. **2** depicts a system that, in accordance with the present invention, accepts data in at least one form through at least one port and which additionally displays data to a user.

FIG. **3** shows stylised examples of an icon and an application window as are commonly found in computer display systems of the background art.

FIG. **4** shows several tiles as might be found in a typical embodiment of the present invention.

FIG. **5** shows an exemplary data structure of the tile object within the graphical user interface of the current invention.

FIG. **6** shows one embodiment of a tile in markup language.

FIG. **7** shows the hierarchy of software objects underlying the current invention, comprising a grid object, tile objects and files or application software.

FIG. **8** shows an exemplary layout of the display produced by the current invention.

FIG. **9** shows an alternative exemplary layout of the display produced by the current invention.

FIG. **10** shows an exemplary layout of the display of the current invention wherein a tile contains another instantiation of a grid.

FIG. **11** shows an exemplary layout of the display of the current invention including specific examples of tile contents.

FIG. **12** shows the data structure of the grid object which forms part of the graphical user interface of the current invention.

FIG. **13** shows one embodiment of a grid in markup language.

FIG. **14** shows a sequence of windows that demonstrate how a grid might be set up for initial use by a "wizard" tool in one embodiment of the present invention.

FIG. **15** shows an example of the architecture of the computer program in a preferred embodiment of the present invention.

FIG. **16** shows the architecture of the application program and its components in a preferred embodiment of the present invention.

FIG. **17** shows the architecture of components of the computer program in a preferred embodiment of the present invention.

FIG. **18** shows an outline of the widget-set used in a preferred embodiment of the present invention.

FIG. **19** shows an outline of the metabase according to a preferred embodiment of the present invention.

FIG. **20** shows an outline of the XP Core and its interaction with the operating system library in a preferred embodiment of the present invention.

FIG. **21** shows an overview of the event system utilized in a preferred embodiment of the present invention.

FIG. **22** shows an overview of the connection layers that are responsible for controlling the download of multiple web-pages from the world wide web.

FIG. **23** shows a number of functions used by the bandwidth controller.

FIG. **24** is a schematic representation of the relationship between a server and a client device.

FIG. **25** shows a series of interactions between a client device and a server.

FIG. **26** shows how a user, a server and third party content providers communicate in accordance with an embodiment of the invention.

FIG. **27** shows an embodiment in which the application program communicates with one or more wireless devices.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

FIG. **1** shows an illustrative configuration of the graphical user interface of the present invention. A grid **1** consisting of a 3 by 3 matrix of nine tiles demonstrates some of the different contents that tiles can display. Tile **10** points to a database of stock quotes. Tile **20** displays the active folders in an electronic mail utility. Tile **30** displays a portion of an alphabetical list of quoted companies. Tiles **40**, **50**, **60**, **70** and **80** point to websites displaying, respectively, high technology news, electronic goods for sale, categories of business news, items available by auction and the Wall Street Journal. Tile **90** points to the file-viewer of a windows-based operating system and displays the currently accessible disc drives.

Within the scope of the present invention, an information source may comprise any analog signal, source of digital data or a datastream, including, but not limited to, video, audio, text and graphics. The information may be in any format, including but not limited to ASCII, bitmap, MP3, JPEG, GIF, TIFF, a mark-up language such as HTML, XML, VRML, HDML, formatted text such as rich text format, or binary.

FIG. **2** is a general representation of a data display system **100** within which the present invention may be implemented. System **100** comprises a central processing unit **104**, an input device **106**, data connection ports **108-1** through **108**-N, a display **110** and a main memory **112**, all connected via bus **116**. Residing in the memory **112** is an operating system **120**, a file system **122**, a cache **124** for temporary storage of information, application programs

128-1 through **128**-N and a graphical user interface (GUI) program **126** that is responsible for presenting information on to display **110**. Information may enter the system through any one of the ports **108-1** through **108**-N which may themselves be connected to a tuner **114**, or via a network interface **134** to a communications network. If a tuner **114** is employed, it may channel input from a wireless signal **130** or a cable network **132**.

In one embodiment of the present invention, system **100** is a personal computer such as a desktop workstation or a portable notebook computer. In that case the input device **106** may be a keyboard, mouse, trackpad, trackball, or any combination thereof and the display **110** may be a conventional cathode ray tube (CRT) or active matrix flat-screen display. The network interface **134** may then be a connection to the internet or to a local area network via a cable and modem or a digital subscriber line.

In another embodiment of the present invention, system **100** is a mobile phone or personal digital assistant and the input device **106** may consist of several buttons on a keypad, a touch-sensitive screen with a touching device or a microphone and voice-recognition software. In this embodiment, the display **110** is preferably an LCD screen or an electroluminescent display and ports **108** receive data from radio signals or a portable modem.

In yet another embodiment of the present invention, system **100** comprises a set-top box wherein display **110** is a TV screen or monitor and tuner **114** accepts input in the form of a wireless signal **130** from broadcast transmissions or cable signals from the cable network **132**. Input device **106** may be a hand-held remote control apparatus or buttons located on the set-top box or touch-sensitive areas of display **110**.

As is known to those skilled in the art, a graphical user interface is a computer program that resides in memory **112** of some data processing system and that provides means for presenting information, input to and output from application programs **128** or content of datastreams from ports **108** on an associated display. In the background art each datastream is associated with a window. The graphical user interface allows a user to control the arrangement and display format of the data content in each window. Usually a graphical user interface permits a user to specify and alter operating parameters of application programs running on the system though, at any given time, a particular application program will take priority, meaning that a particular window will be displaying continually updating content. Typical operating parameters that may be changed depend upon the application program but include the number of buttons on a tool bar and number of visible toolbars, the size of the text displayed and the color of the background.

By contrast, the graphical user interface of the current invention not only permits a user to control the layout of the data content but to prioritize each application program running on the system and each datastream of interest. A novel feature of the present invention is that the data content of any number of the programs can vary in real time and the rate at which the display of each is updated can be controlled by the user.

Tile Objects

In the ensuing discussion, tile objects are introduced and described and contrasted with existing elements of graphical user interfaces. A tile presents content from any information source.

Conventional graphical user interfaces of the background art provide two distinct representations of programs, files and datastreams, as shown in FIG. **3**. One representation is an icon **320**, the other is a window **340**. An icon typically occupies only a relatively small proportion of the available display area and is an easily recognizable depiction of the program or file, either through its logo **322** or some characteristic picture with the name **324** of the program visible. An icon can be selected by, for example, a touch screen pointer, a cursor controlled by a mouse with a button or by a keyboard stroke or any combination of the foregoing. In response to a further selection operation on an icon, for example a double-click of a mouse button, the graphical user interface will provide a window that can be used to communicate further information to the program or review the associated datastream.

The window may occupy a substantial percentage of available screen space, usually 90–100%. The window **340** usually comprises a title bar **348** and a display area **354**. The window **340** can commonly be resized by the user for example by using buttons **352** or a draggable area **356** and has a format which contains many active areas around its borders. Examples of active areas include a menu bar **342**, a vertical scroll bar **344**, a horizontal scroll bar **350** and one or more tool bars **346**. Each active area may be used to control aspects of the window's appearance or to set parameters specific to the operation of the program associated with it such as text formatting options in a word-processing package or redirection of a web-browser to its stored home location.

In the present invention, a third graphical representation of programs and files, herein called a tile, is introduced. Tiles permit "dynamic bookmarking" of information in that each tile is a viewer of a single information source—including streaming data sources—and can be customized with the user's choice of content.

A tile is different from an icon because it provides a real-time or near-real time view of the underlying information in that it contains continually refreshed content. A tile is different from a window because a tile will typically be smaller in size, allowing the user to view multiple tiles simultaneously if desired.

A tile provides an at-a-glance view of the current status of the program or file associated with it but does not necessarily have the large number of active areas associated with windows such as title bar, menu bar and scroll bars. Therefore tiles lead to a reduction in clutter on the display screen because many tiles may be displayed simultaneously without overlapping with one another in the way that windows must necessarily do. Tiles are superior to icons because they give an immediate indication of the current state of the file or program and have utility functions associated with them, as described below. Another advantage of using tiles is a uniformity of appearance between tiles which correspond to different programs and datastreams. The display content of a tile will differ from application to application though its size and format need not.

A tile is associated with a program, file or datastream, in the same way that an icon and a window are. A tile may present data in any of a number of ways. For example, in the preferred embodiment, the tiles may present a miniaturized, "thumbnail" view of the underlying information; a "porthole" view of a portion of the underlying information as viewed at full size; a symbol indicating whether the information has been updated since it was last viewed; or a custom interface designed to allow rapid access to the underlying information. The way in which a tile displays content may be independently configured for each tile.

FIG. **4** illustrates representative tiles. Tile **402** displays a picture or graphic such as may be stored in a bit-map, JPEG, TIFF or GIF file, or on a world-wide web page. The content of tile **402** is typically a miniaturized representation of a graphic or still-frame from a datastream. Tile **404** displays a portion of a text document or text of a world-wide web page. In this sense the tile functions as a transparent panel placed on top of a document, thus permitting a portion of the document to be displayed. Tile **406** displays a further array of tiles that may be displayed in full by expanding tile **406** to occupy the full area of the display. Tile **408** has been configured to link to an electronic mail program. An alarm setting associated with tile **408** has been configured so that the tile displays an envelope and the message "New Mail!" when an unread message has been received by the mail program. Tile **410** displays a name, "FM 101", denoting the title of a broadcast signal, in this case audio, that is associated with the tile. Tile **412** displays a "thumbnail" of the document viewed in a window such as **340**. A "thumbnail", as used herein, is a miniaturized representation of an image that retains sufficient characteristics to permit easy recognition of the image. For example, if the document displayed in a window were a multipage document, tile **412** may display a thumbnail of the first page of the document.

Tiles are selectable and live. When a tile is selected, whether by mouse click or otherwise, the tile instantly provides the user with access to the underlying information, whether that data be a hierarchical menuing system leading the user to a different level or tiles, a word processing file stored on a local area network, a spreadsheet stored locally on a user's computer, HTML file on the Internet, or a television signal. The tiles are live in that each contains real-time or near real-time information.

In a preferred embodiment a selection operation is associated with a tile. For example, clicking on a tile will cause it to immediately refresh its contents. Selecting tile **402** causes the most recent frame of the video stream to be displayed by the tile, or, if the picture were obtained from a static graphic file, causes the most recent version of the file to be displayed by the tile. Selecting tile **404** or tile **412** causes the most recent version of the document to be displayed.

In a preferred embodiment, double-clicking a tile causes that tile to occupy a substantial fraction of the whole display area. In this embodiment, double clicking tile **402** or **412** causes the image to be expanded to fit the whole display area. In an alternate embodiment, selecting a tile causes it to occupy an area in the middle of the display that is larger in area than a single tile but does not occupy the full display. If tile **404** were double-clicked, as much of the document as could be displayed in the enlarged tile in regular font size becomes visible. Double-clicking tile **408** causes the mailbox window of the e-mail utility to be sufficiently enlarged to allow new messages to be selected and read. Double-clicking tile **410** causes the audio stream to become audible over the appropriate channel of the system.

A representative tile data structure **500** is shown in FIG. **5**. It is important to understand that the tile itself is an image that at any given instant is resident on the file system. This image is separate and distinct from the application program or file associated with the tile. The tile data structure **500** comprises two addresses: a tile address **502** that defines the location on the file system where the tile image is stored; and a target address **504** that is the location at which the file or application program associated with the tile can be found. Additionally, the tile data structure contains a name **506** that may be displayed on the tile in certain circumstances. Tiles of the present invention may be assigned at least seven functions, including but not limited to: an initialization function **508** that is responsible for establishing a connection with the target address **504**; a refresh function **510** that handles updates to the tile image stored at the tile address **502**; a screen-size function **512** that stores the dimensions of the display area filled by the tile upon receipt of a request; an alarm function **514** that permits the tile to display an alarm or warning when the application program associated with the tile encounters a designated event; an on mouseover function **518** and an on mouseout function **520** which control the behavior of the tile when a selection tool such as a mouse-controlled cursor is placed respectively on and off the tile; and a toolbar function **522** which may permit an array of special buttons to appear on or adjacent to the tile for the purposes of adjusting properties of the tile. In one embodiment, a tile is configured so that a right-click of a 2-or 3-button mouse while the cursor is over the tile would activate the tool-bar function. In a preferred embodiment, right-clicking on a tile can reveal a menu of options that enables the user to ascertain properties of the tile, such as the bandwidth it is consuming.

In one embodiment of the present invention, the tile is itself a document created in a markup language such as HTML or XML as shown in FIG. **6** and is suitable for display in a web-browser. In this embodiment, a tile occupies a position in a table defined with elements of mark-up language which will be familiar to one skilled in the art. Tile-specific attributes are introduced which control the way in which a web-browser displays the tile. In the example in FIG. **6**, the tile has a clickable map, i.e., separate areas of the tile produce separate results when clicked. Also, this tile has a toolbar, which in one embodiment may appear if the mouse is right-clicked when the cursor is over the tile.

In another embodiment of the present invention, tiles communicate with one another and have conditional content. That is, the content of one tile depends upon the content of another.

Grid Object

The arrangement, layout and independent functioning of the tiles on the display and exemplary software for their control is now described with respect to FIGS. **7**–**13**.

The overall hierarchy of a graphical user interface embodied by the present invention is summarized in FIG. **7**. The Grid **700** is the top level functionality to which application programs are subordinate. The grid can replace the functionality of a user's computer desktop and offers similar and additional features. Grid **700** comprises a matrix of tiles of which tiles **702**, **704**, **706**, **708** and **710** are representative. The user can control the grid in such a way that the tiles **702**, **704**, **706**, **708**, and **710** present simultaneous information content from a plurality of sources. The grid controls the layout and priorities of the tiles. Each tile is associated with a data stream or application program and allows a choice of displayed images. In particular, different tiles can be associated with different contents. For example, tile **702** is connected to a web-page viewer **712** such as a browser. Tile **704** is connected to a source of streaming video **714**, such as Real Player. Tile **706** is connected to an audio player **716** such as a CD-player program or a source of streaming audio such as Real Audio. Tile **708** is connected to a file viewer **718** such as a text-editor or a word-processing package. Tile **710** is associated with another grid object **720**, thereby permitting a "layering" of information hierarchies. In a preferred embodiment, a grid embodies a similar underlying data structure to a tile.

Each tile is separately associated with a source of information, for example, an application program, datastream or file, any one of which may be another grid object. Such a hierarchical structure permits a user to organize programs and information through the graphical user interface. For example, separate categories of information can be displayed on separate grids allowing each grid to be associated with a theme.

By using the native attributes of a tile, a user may specify a presentation of the grid, consisting of its dimensions, (i.e., the number of tiles to display and their arrangement), and the programs or files to be associated with each tile. A single grid, composed of multiple tiles, may therefore present a number of information sources simultaneously.

Together, the grid and tiles comprise the application through which a user can view simultaneously information from a multitude of his available sources including multiple sites on the Internet, receive signals from multiple broadcast channels, and open and view multiple files. In its initial embodiment, the application may be run through conventional computer operating systems, whereupon it overlays the user's desktop and acts as if it were a "borderless browser". Therefore the application resides over existing applications without replacing any of them; rather it enables them to be called from the grid itself. The application, therefore, becomes a graphical file manager in which the content of continuously changing files, i.e., datastreams, is being displayed in real-time or near real-time, depending on the assigned priority. Effectively, the application replaces the user's desktop with a more visually intuitive dynamic menuing system.

In a preferred embodiment, a grid of tiles replaces the functionality of the computer "desktop" utilized by many modern computer operating systems. Whereas a desktop is typically populated by static icons and tool bars, a grid of tiles instead presents to the user an array of snapshots of current programs and files. The essence of the grid is that its content is dynamic and informnative. As previously discussed, icons are inherently limited in the information that they can present and windows clutter the entire desktop. By contrast, each tile on the grid can show the current status of the data or datastream associated with it. The fact that a tile may refer to a separate grid permits nesting of grids and consequently a hierarchy of organized information sources.

The grid also understands the interests of the user and acts as a repository for passwords and identifiers to subscription services. In this way, it is not necessary for a user to remember and enter a user name and password to access data requiring such a log-in. In this same vein, Internet sites of interest can be "bookmarked" and stored by the grid, each such site possessing its own tile. The grid is also nestable in that a tile in one grid may point to a separate grid and tiles in the separate grid may point back to the first grid or to yet more grids. If desired, a user can impose a "theme" on a grid and thereby categorize, group, and/or otherwise manage his data sources. In this manner, the user can group tiles relating to particular subject matters, Internet sites, documents, or otherwise in a grid according to some speciality. A tile in such a grid could link to another grid to provide a connection between related categories. Equally, a tile on more than one grid can point to the same information source.

In a preferred embodiment, the grid permits a regular layout of tiles on the display screen such that the tiles are uniform in size and shape, as depicted in FIGS. **8–11**. Each tile is indexed by its position on the grid. For example **802-2-1** is the first tile in the second row. Tiles in the first row of the grid are **802-1-1**, **802-1-2**, **802-1-3** and so on, to **802-1** Tiles on the second row are **802-2-1**, **802-2-2**, **802-2-3** and so on, to **802-2**-N. And, tiles on the bottom row are **802**-M-**1**, **802**-M-**2**, **802**-M-**3** and so on to **802**-M-N. There are no gaps between the tiles, the tiles are not permitted to overlap and the whole grid is covered by tiles. FIG. **8** shows one embodiment in which all tiles are the same size and are presented in an array comprising M rows and N columns. There is no particular requirement that the arrangement consists of more than one row and more than one column. On the display of a mobile telephone or smart phone, for example, a single row of tiles may be apposite.

FIG. **9** shows an arrangement in which there is a unit tile size, that of tile **802-1-1**, but tile **802-1-2** and tile **802**-N-**1** have each been configured to occupy regions of the grid equal to exact multiples of the unit tile size. Such an arrangement may be useful and important if one or more datastreams is of particular interest but the others are also to be monitored at the same time.

FIG. **10** shows an arrangement of tiles in which Tile **802**-M-**1** is associated with a further grid **1000**. The lower half of the figure shows an enlarged perspective of that tile showing a grid with Y columns and X rows.

FIG. **11** shows an arrangement of tiles in which are depicted different application programs associated with three of the tiles. Tile **802-2-2** links to a file viewer displaying a specific file; tile **802**-N-**1** presents streaming video; Tile **802**-M-N depicts a page of information on the world wide web.

FIG. **12** shows a schematic data structure of the attributes of one embodiment of grid object **700**. The architecture shown in FIG. **12** applies to any grid, including a top-level grid **700** shown in FIG. **7** as well as to a grid that is contained within a tile.

Significantly, the grid manages the flow of information to the tiles. For example, the grid can communicate with the display device in order to determine its current configuration and allocation of resources. In one embodiment the grid continually cycles around the currently displayed tiles, one by one, refreshing the content of a tile each time it is accessed. When a given tile is refreshed, the refresh operation is completed before refreshing the next tile in sequence. In this way, the cycling rate may be set so that the current content of all tiles are reasonably up to date. The cycling can be interrupted by a user selecting a given tile, so that that tile alone becomes continuously updated. In this way, the user does not need to worry about manually refreshing multiple tiles.

In a preferred embodiment, according to priorities that may be applied to individual tiles on a tile by tile basis if desired, the grid manages the refresh rate of each tile in the grid. For example, for locally stored word processing or spread sheet files, the user might configure the tiles to refresh only when the underlying data is written to the local hard drive. Similarly, a user might configure tiles containing infrequently updated HTML data from the Internet to refresh at a certain time each day. At the other extreme, a user might configure an active tile to display a television channel at a refresh rate of 29 frames per second, while at the same time configuring inactive tiles to display different channels at a refresh rate of once every five seconds. In this way, a user could monitor many channels until program content of interest appeared in one of the tiles without the burden of actively refreshing each tile.

The grid itself has an address **1204** that specifies its location within the file system of the device in which the application program program runs. The grid has associated with it several utility programs: a configuration wizard **1206** that may be called by the user when setting up a new grid; a tile creation function **1208** utilized by the configuration wizard when initializing new tiles; a tile annihilation function **1210** utilized in case of error or when resizing the grid.

The grid object stores the number of rows **1212** and the number of columns **1214** of tiles that are present. The grid also stores a tile list **1216** containing attributes of each respective tile. In particular, the address of each tile, its priority and its refresh rate are stored by the grid program. The grid also stores other attributes of tiles such as their respective positions on the grid as given by their column and row number. The priority of a tile may be used to determine its refresh rate in one embodiment of the present invention. A tile can have a password feature built into it if it is desired to restrict access to the tile's content. The grid itself can have a toolbar by which its attributes may be accessed and modified.

In a preferred embodiment, a grid is a special form of a tile. It is a tile that can create and manage an array of other tiles. Accordingly, its data structure also comprises those elements of a tile data structure shown in FIG. **5** in addition to those shown in FIG. **12**. If the grid has a parent grid, the address of the parent grid **1202** is stored. For example, grid **1000**, associated with tile **802**-M-**1** of FIG. **10**, has grid **700** as its parent.

The grid can be configured to contain any number of tiles from one to as many as can reasonably fit on the user's display.

In one embodiment of the present invention, the grid is a document created in a markup language such as HTML, SGML or XML, FIG. **13** and is therefore suitable for display via a web-browser. In this embodiment, the addresses of the parent grid, the grid itself and each of the tiles are expressed as a Universal Resource Locator (URL). The various functions controlled by the grid are accessible through function calls devised according to methods familiar to one skilled in the art. For example, "dynamic HTML", java applets or simple CGI-scripts could provide the technological basis for enabling various grid utilities.

The application may be downloaded from a predetermined web-site and operates in a client-server mode. Users may download preconfigured grids from the predetermined server. A grid configuration "wizard" program which guides a user through a step by step set up of a custom-grid may also be downloaded. Other web hosts are able to deliver content to end-users via the predetermined server. Some basic functions of the grid can be carried out on the predetermined server and provided to the user.

Grid Configuration Wizard

In one embodiment of the present invention, the set up of a particular grid is achieved through a grid configuration program ("wizard") that is downloaded to the display device from a remote site. The grid configuration program permits a user to define and install one or more grids on the client system. When a tile is partitioned into a further array of tiles, the grid configuration program can also be used. One embodiment of the user interface of the grid configuration wizard is shown in FIG. **14**.

A first screen displayed by the grid configuration wizard comprises an application program logo **1404**, button **1406** to guide the user to the next screen and a number of choices, such as **1408**. The user is offered the choice of preconfigured, or "standard" grid configurations, selected from a list. Examples of such grids include grids themed by content such as sport-related grids or by type of data such as grids whose content is video-based.

Additionally, the user is permitted to configure "customized" grids in which each tile can be taken from a list of predefined samples or can be initialized according to the user's wishes. In a second screen **1412** displayed by the grid configuration wizard, a tiled area **1416** represents the grid that the user is building. Sample tile categories **1418** such as "weather", "news", "stocks", or "sports" are listed. In an alternate embodiment, a grid can be filled by the "drag and drop" technique in which a selected document **1414** is moved on to the display area of the grid configuration program and automatically becomes a tile. A button **1410** offers the user the chance to go back one screen.

In a third screen **1420** displayed by the grid configuration wizard, the user can name the grid and, optionally, store it for future reference, for example in an archive of preferred grids. The user can elect to finish grid construction by clicking the "Finish" button **1412**, or launch the grid immediately by activating button **1422**. When launching the grid immediately, the grid is automatically constructed on the fly according to the content and tile types specified by the user during the set up procedure.

Architecture of Application Program Software

The hierarchy of the software in a preferred embodiment of the present invention is illustrated in FIG. **15**. The software comprises a number of modules. In FIG. **15**, an arrow connecting two modules means that one module uses an interface of the other. The arrow comes from the module invoking the interface towards the module whose interface is being invoked. An interface may simply be a function call between the two modules or, for example, a call to a dynamic linked library (DLL).

The Surfcast application program **1500** takes its underlying data from two sources, metabase **1506** and a system library **1514** that resides on the device on which the application program is running. For example, the application program **1500** may be required to call functions from the system library while it is running. Actual tiles that a user visualizes can be spawned from the metabase **1506**.

Data structures for user tiles **1502** are obtained from the XP core **1504** which itself also utilizes components from the system library **1514**. The XP core is an abstraction layer for the operating system environment. Tiles that utilize components such as buttons take these components from the widget set **1508**. A widget is a basic user interface element such as a button or a text input box. The metabase also uses an interface of the widget set **1508**, and can therefore use functions within the widget set. The widget set requests functions from the XP core.

Objects in the metabase **1506** that retrieve content from remote sources such as world wide web pages utilize a connection manager and bandwidth controller **1512**. A URL loader **1510** decides whether content should be obtained afresh by contacting the connection manager **1512**, or from content previously stored in cache. Effectively, the URL loader manages the connection manager, and calls functions within it.

Underlying all of the application program's operations are functions from the operating system library **1516** that is supplied with the device on which the application program is running.

Each of the objects shown in FIG. **15** is now described in further detail by reference to FIGS. **16**–**21**.

The Surfcast application program **1500**, in FIG. **16**, comprises a launcher **1618** and a framework **1620**. The launcher opens the program from scratch whereas the framework is responsible for managing grids and tiles. The user interacts directly with the framework to set up a preferred arrangement of tiles on the display. In one embodiment, the framework initially contains a prescribed set of tiles. The framework controls communication between tiles, for example, in the case of conditional content.

Tiles **1502**, FIG. **16**, are the equivalent of an application in a conventional GUI system. They can be built in C++ using the Surfcast tile builder application program interface (API) using XP core classes, or via a utility such as a custom tile editor or via a script file. Some predefined tiles are included with the basic Surfcast system including a web browser tile called a surftile **1602**, tiles for contacts and communications such as a chat tile **1604** and an e-mail tile **1608**. Media-player tiles such as a video tile **1606** are also supplied, as are tiles that interface to commonly used desktop programs. A word tile **1610** interfaces to a word processor; excel tile **1612** interfaces to a spreadsheet program. A general content viewer that can compose pieces of content clipped from a variety of sources, layout tile **1616**, is also provided. Reg tile **1614** is a general purpose tile that permits a user to define his own tile type.

In a preferred embodiment, all tiles have a common base class, and each specialized type of tile has its own class that builds upon the base class. There are many ways in which specific tile classes can be derived from the conmmon base class. Functions for specific tile classes are readily apparent to one of skill in the art. Tiles may additionally be represented by markup language files and viewed within a web-browser environment.

The data classes in XP core **1504**, FIG. **16**, comprise base classes and utility classes with which tiles and widgets are built. In general, classes within the XP core describe how tiles can communicate with one another and with the overall application program framework **1620**. In particular, XP core classes are generic and portable, thereby permitting cross-platform capability.

XP core classes include: tilebase **1622** for the generic class that underlies all tiles; tile base view **1624**; tile controller **1626**; canvas **1628**; and classes for event handling **1630**. A view is what a tile uses to draw itself. Tile base view is the base class for all views associated with a visual object. A controller processes events, for example mouse moves, clicks, keyboard events and external events. Tile controller is the base class for controllers associated with a tile. A canvas is an area of screen on which to render some image, for example a tile.

The metabase **1506**, FIG. **17**, is a local store for platform-specific implementations of the descriptions of tiles, grids and other objects. Tiles, grids and content are created, saved and restored via the metabase. The metabase contains tile type and content type registries such as tile registry **1732**, and a local database of grids and content such as content store **1734** and grid store **1736**. Unknown tile and content types can be obtained from remote servers. Tile types are abstracted so that if a grid contains a particular type of tile, for example an e-mail tile, the metabase provides whatever is appropriate for the device the application is running on. Items in the metabase are "persistent", that is they are not saved explicitly but are preserved from session to session.

The widget set **1508**, FIG. **17**, comprises a platform-specific set of visual components that tiles can use. Widget set contains the predefined widgets that are included with the system. It can be extended with new widgets. It includes such useful widgets as a button **1738**, a label **1740**, an edit widget **1742** that enables a user to enter text into an editable field and a list widget **1744** that enables a user to select from a set of options. Items within the widget set can be used with tile types such as text input tiles, web browser tiles, and a streaming video tile. The term widget can also include more complex objects such as a video player that can be inserted into a tile as easily as a button.

The URL loader **1510**, FIG. **17**, provides a mechanism for retrieving content. The URL loader **1510** interacts with connection manager **1512** for tiles which need to make a network connection. Tiles and the metabase ask for content for a given URL and the content manager will attempt to retrieve it. The metabase also contacts the connection manager through the URL loader to ascertain whether there is sufficient bandwidth for the transfer. In particular, the connection manager decides whether the URL loader should furnish tile content from the cache **1746**, as would be the case if the content has been recently displayed and stored locally. Alternatively, if the content is not cached, the URL loader supervises loading of content from the location specified by the URL.

The URL loader also comprises a file manager **1748** for organizing the cached content. The URL loader additionally comprises a DOM (data object model) renderer **1750** that administers the parsing of pages in markup languages such as XML and HTML. The URL loader may also comprise an implementation of an API for XML rendering such as SAX. In an alternate embodiment, such an API may reside in the XP core module.

The system library **1514** comprises commonly used utilities within the application program, including, but not limited to, code for string manipulations, file handling and server communication. The system library module comprises generic code and can be compiled for any operating system.

The operating system library **1516**, FIG. **17**, comprises utilities that differ in their implementation from system to system but are needed for operation of the application program. For example, utilities that may be found in system library **1516** are those that provide support for threads and synchronization **1752**, debugging tools **1754**, graphics libraries **1756**, further basic string manipulations **1760** and connections **1762**. Additionally, not shown in FIG. **17**, useful items in the operating system library include utilities that permit definitions of objects, sockets, input devices and hardware devices. Items in the operating system library are accessible through classes with documented public interfaces.

The operation of the widget set **1508** is further described with respect to FIG. **18**. As previously mentioned, the widget set contains widgets for various functions such as buttons, text labels, etc. Illustrated in FIG. **18** is a set of widgets for buttons.

FIG. **18** shows the class hierarchy for the button widget. Base classes for widgets are grouped together in box **1809**. Widget class **1810** is a container for other classes. All widgets use the base class widget **1810** stored in the XP core module and further described later. Accordingly, button **1800** is a specific class inherited from widget **1810**.

Widget view **1812** is a class, also stored in XP core, that defines the look and feel of the widget. Button view **1802** inherits from widget view and controls how a button draws itself.

In the scheme of FIG. **18**, user tile **1816** comprises three objects, tile controller **1626**, the base class tile **1622** and tile base view **1624**, each of which is found in XP core and is further described later. Tile base view is responsible for drawing the tile and can employ one or more widgets.

A button is something that a user can click on. A button provides button events to a button event consumer. A button event consumer is also known as a client, i.e., clients that use buttons implement button event consumer **1808** to be notified of button events. For example, a button event consumer may be a "play" function in a video tile. The button event consumer interacts with a control structure tile controller **1626** associated with a user tile **1816** by telling the tile that the button has been activated. Button event consumer **1808** is itself a class that inherits from an event consumer class, described later.

Button controller **1804** controls how button events **1806** are processed, for example mouse and keyboard events. It inherits class structure from widget controller **1814** stored in XP core. Not all button events are recognized by the button controller: for example, a particular key-stroke may have no effect on the state of the button.

Other widgets follow a similar pattern. They include: textedit, for inputting text; textlabel, for displaying text; textspinner, for selecting from a choice of options; datewidget, for entering a date; list, for selecting from a choice of options; titlebar; and a toolbar.

Metabase **1506** is further described with respect to FIG. **19**. Arrows between the components in FIG. **19** denote interfaces. The metabase comprises a registry **1900** and store **1902** of tile and grid types, a content store **1918** and a content and tile link database **1920**.

Content store **1918** is a cache that contains content of tiles for the previous session. Content tile link database **1920** copes with descriptions of relationships between tiles and also themes of content for related tiles. This database can also be used in the context of "knowledge management", i.e., those operations that monitor a user's activities and attempt to suggest further sources of content based on it. Both content store **1918** and the database **1920** interface with tile type registry **1900**.

The tile type registry **1900** contains a list of tile and widget types along with information about how to create them and what they are called, along with other information such as pointers to objects from which tiles are created. The tile and grid store **1902** contains a library of stored grids and tiles. Tiles can save themselves or be restored. In a preferred embodiment, grids are just special cases of tiles. Tiles from the tile library can be called and displayed on the screen by asking the registry to load tiles. The tile and grid store interfaces to an XML library **1906**.

Tiles from the tile and grid store can also be saved outside the metabase in a library of markup language files, e.g., an XML library **1906**.

Also in the metabase is metabase tile **1904** which utilizes tile base **1622** from the XP core module. All tiles inherit from this class.

DLLs can contain additional tiles and widgets that can be created by independent third parties. These can be implemented within the metabase through the tile factory **1916** and tile creator **1914**, both of which interface to the tile type registry **1900**. The tile factory **1916** contains the description and classes necessary for someone to register a new tile type. Tile creator **1914** is the code that does the tile creation at runtime. In general, independent creation of tiles is facilitated by supplying a tile toolkit to third parties.

Inquiry utility **1912** is an optional means for an outside user to interface to the metabase, for example to ascertain the class structures of stored tile classes.

The content store **1918** follows a similar pattern to the tile type registry and the tile store. The content and tile link database **1920** is a database of information about how tiles and grids are related, and how content is related between them.

FIG. **20** shows a further description of classes found in the XP core and their interaction with the operating system library **1516**. As discussed below, some arrows between objects within XP core denote inheritance of classes. Canvas **1628** describes an area of screen that can be drawn to.

GfxContext **2002** is a set of graphics primitives, for example for line-drawing, colors and space filling. It is a generic version that encapsulates and abstracts operating system-specific features inherited from GfxContextFactory **2032** in the operating system library **1516**. Win32 GfxContext **2034** is an example of a graphics context used with the Windows operating system. Other GfxContent **2036** includes alternative platform dependent graphics contexts.

The foundation classes are tile base **1622**, tile base view **1624** and tile base controller **2008**. Tilebase **1622** is the base class for all visual objects such as tiles, widgets and grids. The tile class **2010**, and widget base class **2016**, inherit from tile base. A widget effectively functions as a special kind of tile that can be placed inside a tile. Widget base **2016** is not meant to be instantiated on its own but is a foundation for the widget class **1810**, FIG. **18**, used by a generic widget.

Tile base view **1624** is the base class for all views associated with a visual object. A view is what a tile uses to draw itself. Also inheriting from tile base view is tile view **2012**, the base class for all views associated with tiles and widgetview **1812**, the base class for all views associated with widgets. Tile base view and tile base interface with canvas **1628**.

Tile base controller **2008** is the base class for all controllers associated with a visual object. Inheriting from tile base controller are tile controller **1626**, the base class for controllers associated with a tile and widget controller **2020**, the base class for all controllers associated with widgets. A controller processes all events. Tile base interfaces with both tile base view **1624** and tile base controller **2008**. Tile base controller interfaces to event system **2022**. Finally, event system **2022** communicates between the operating system library **1516** and the tile box controller **2008**.

Event system **2022** is further described with respect to FIG. **21**. An event can be any one of a mouse movement event, another mouse event such as a mouse-click, or a keyboard event such as a keystroke. In FIG. **21**, event **2112** is the base class. Other classes such as mouse movement event **2106**, mouse event **2108** and keyboard event **2110** derive from the base class by inheritance. The event consumer **2104** is a class responsible for directing events to the controller. The event producer **2102** is interpreting system events into Surfcast events for an event consumer. The boxes **2122**, **2124**, **2126**, **2128**, **2130** and **2132** are multiplexes handling the case where multiple clients are affected by multiple types of events. An event is communicated to tile base controller **2008**.

Managing Connections to More than One Datastream

When two or more tiles connect to sources of data available over a network, communication must be established in such a way that the rate at which updated data is transmitted to the grid can be controlled. In practice, for an embodiment of the application which resides on a user's computer, a flow control protocol such as TCP is required.

In this way, each tile can communicate with the remote datastream to which it is linked and a determination can be made of available bandwidth at the time of data transfer. Alternatively, in a client-server mode, flow control is not necessary because communication with the server suffices, as is described below.

It is not practical to fire up a separate browser program from each tile that wishes to download data from a site on the world wide web. A web-browser is very greedy on memory and resources and the user would have little or no control over the respective rates at which data was downloaded from different sites.

Instead, in a preferred embodiment of the present invention, a hierarchy of layers manages the simultaneous connection and allocation of resources to different world wide web sites, as shown in FIG. **22**. The layer structure applies to the way in which any given tile downloads content.

At the highest level there exists a widget, referred to as "Surf widget" **2200**, which is the basic browser control within the application program of the present invention. Ideally, this widget will operate in conjunction with any commonly used world wide web browser. It will typically be associated with a tile type such as surftile **1602**.

The surf widget communicates with a surf widget controller **2202** in a control layer **2201**. Also in the control layer is a web browser application program **2204**. Examples of such a program include Microsoft's Internet Explorer and Netscape's Navigator software. The surf widget controller **2202** handles the interaction between the surf widget **2200** and the web-browser **2204**. The surf widget controller also passes on requests from the browser to the URL Manager **2206** in the next layer, the URL layer **2205**. The surf widget controller then pipes back the results to be rendered to surf widget. A typical example of this process in operation would be: a user clicks on a hyper-link in a web-page; the web-browser makes a request for that page to surf widget controller **2202**; the request gets handed to the URL manager **2206**; once the page is loaded, the URL manager **2206** notifies the surf widget controller, which in turn sends the information to the web-browser for rendering.

The responsibility for obtaining pages of content is that of the URL layer **2205**. When a URL is requested, the URL manager **2206** issues a request for the page and any subsequent media to the connection manager **2210**. The URL manager keeps track of the requested URL for future use, if it is requested again. The URL manager also queues up URL's that have been requested according to their focus, i.e., the tile that a user has currently selected and according to the respective priorities of the active tiles.

In a preferred embodiment, a pre-fetch utility such as URL pre-fetch manager **2208** can be implemented. It saves the user time if items can be pre-fetched instead of waiting for their download. Several strategies can be used to obtain pre-fetch items for the user. Using a history of a user's previous browsing habits, it is possible to predict what the user will probably want next. Another function of a pre-fetch utility is to periodically check the validity of items in the cache and to make sure they are up to date. As would be familiar to one skilled in the art, some of the new HTTP1.1 methods would prove very useful for this; namely the conditional gets. Another strategy is to start loading links from the page that a user is browsing, regardless of whether the user has selected the links. Although such an approach could be very wasteful of resources if there are a lot of links and very few are ultimately accessed and also because a lot of links tend to be advertisers, in situations where very high capacity bandwidth exists, this approach could be effective.

The connection layer **2209** handles each individual request for download passed to it through the URL manager, regardless of whether it is an HTML page, a graphic or sound file. The connection manager **2210** understands the total bandwidth available for allocation, for example, whether the device is connected to a modem or a T-1 line. It will also manage the connection to the requested site and maintain its own cache. Before making a network request for an item, connection manager **2210** first checks its cache, the connection manager cache **2212**. If the item is not in the cache, the connection manager then passes the request off to the HTTP protocol socket **2214** in the protocol layer **2215**. The way in which HTTP protocols and caches work is familiar to one skilled in the art.

The protocol layer **2215** consists of a suite of different socket types, **2214**, **2216** and **2218**, intended to support different communication protocols, such as HTTP, FTP and also a client server protocol specific to the application program via the surfcast protocol socket **2218**.

The socket layer **2219** comprises at least one socket **2220**. The socket layer wraps up all the system implementation specifics for a given platform and allows generic socket types us to be built on top. The socket keeps track of its bandwidth usage, which can then be queried at the connection layer. The socket layer then facilitates bandwidth management.

With all communications going through the same socket layer it is possible to easily collect data about a socket's bandwidth usage. If, at the connection layer, it is noticed that the total bandwidth allocation has been exceeded, it is a simple case of blocking further data transfer until such time as total bandwidth usage falls back under what has been allocated.

As a user switches focus from one tile to another priorities can be dynamically re-allocated to ensure the fastest possible loading of the selected page. All other communications can then abide by the same rules, allowing for complete control.

The sequence of events and functions in a "dynamic bandwidth allocation" feature of the present invention are described as follows. The dynamic bandwidth allocation feature involves the URL loader, the connection manager and the bandwidth controller.

The tiles that need access to the network resource for downloading content from a URL, pass certain parameters to the URL loader which manages all such requests from the tiles. These parameters include the URL itself, the priority of the tile, the minimum bandwidth requirement if any, and the maximum bandwidth requirement, if any.

The URL loader detects the need for a connection to a network resource, as would be notified to it by the connection manager. In the case of dial-up connections, the connection manager is responsible for allocating the modem resource and making the dial-up. Once a connection is made and the network resource is available, the URL loader requests the bandwidth controller to start delivering the required content, taking into account the additional parameters for each request.

The bandwidth controller **2300** is an object that comprises a number of functions, as shown in FIG. **23**. AddURL **2302** is a function used by the URL loader to add an additional connection request to those already considered. RemoveURL **2304** is used by URL loader when cancelling or aborting a request. GetURLStatus **2306** is used by URL loader to obtain a status report for a given request. GetStatus **2308** is used to obtain a general status report for the overall bandwidth. The bandwidth controller **2300** is additionally able to execute a main cycle loop over the outstanding URL requests for the purposes of managing the bandwidth among them. The following pseudocode describes steps within the main cycle loop, according to one embodiment of the present invention.

```
while (uncompleted requests outstanding)
{
    calculate bandwidth obtained per request;
    check for need to postpone, stall or cancel lower priority requests;
    check for need to increase higher priority requests;
    detect completed requests and notify requestor;
    detect undeliverable requests and reissue or cancel if necessary;
    carry out other service and statistics functions, as required;
}
```

Each of these steps is explained, as follows. One or more of the steps may be performed in a different order from that presented above without departing from the scope of the present invention.

The step of calculating the bandwidth obtained per request utilizes a function that calculates obtained bandwidth for each of the managed requests, including requests that are stalled (or postponed) due to priority issues. This calculation takes place frequently because of the nature of the network. The bandwidth obtained can vary drastically even during the course of the individual network transaction, and therefore a priority based dynamic system must continuously accommodate such fluctuations. The result of calculating the bandwidth obtained is used both for feedback to users and for making decisions in the subsequent steps within the main cycle loop.

The step of checking for the need to postpone, stall or cancel lower priority requests is another precautionary mechanism to use for the purposes of adjusting the currently active connections. If outstanding requests are not achieving the desired minimum bandwidth, or an actual bandwidth in line with the priority of a given request is not achieved, bandwidth must be made available to the higher priority requests, by stalling, cancelling or postponing lower priority requests. A throttle feature implemented consistent with the layer nature of the stack can be applied wherein the frequency of issuing requests can be decreased. A complete cancellation of a request followed by reload from cache is usually the last resort.

The step of checking for the need to increase higher priority requests has the opposite effect. If applied, higher priority requests are increased by means of spawning additional simultaneous requests, or by increasing the throttle mentioned above.

Both the steps of checking the need to postpone and checking the need to increase a priority can provide feedback to the user in terms of their success or failure. Status parameters can additionally be collected and calculated.

The step of detecting completed requests and notifying the requestor utilizes a function for handling successful completed requests. Conversely, the step of detecting undeliverable requests followed by reissuing or, canceling the request if necessary is the mechanism for avoiding undeliverable requests, or retrying temporarily unavailable requests.

Other service and statistics functions may also be called, as may be necessary for supplying information to the layers above the connection layer.

Client-Server Interaction

In one embodiment of the present invention, FIG. **24**, the user at client device **2400** interacts with server software on a server **2402**. The server stores locally a profile comprising user-specific content **2406** that can feed customized data to the user. For example, the user may store pre-defined grid configurations on the server. Additionally, passwords for specific web-sites can be stored along with the user's profile. A grid generator **2404** on the server creates a grid of tiles according to user-specified content. Each tile has been created on the server by producing an image from the location specified. For example, tile creator **2408-1** produces a tile from content **2410-1**. Thus, when a user logs on to the server, for example through a conventional web-browser, a grid of tiles is downloaded to the user's system.

The client-server embodiment provides a number of advantages. For example, individual users and the devices they use may be differentiated. Therefore, the tile-content that the server delivers can be customized according to the rendering device.

Turning next to FIG. **25**, each time a user logs on to the server **2402**, a "session" is initiated, step **2500**. The server verifies the user's identity, step **2502** and acknowledges the log-in, step **2506**. The client registers one or more resources, such as connection bandwidth, cost per transmission unit and properties of local playback device, step **2504**. From this information, the server identifies the client device type, for example, set-top box or personal computer, step **2508**. At step **2510**, the server retrieves grid settings specific to the user, if appropriate. Details of a session, as defined by tile content and priorities can be held over from one session to another, both for the purposes of permitting a user to continue with ongoing work and in order to protect against the adverse consequences of abnormal disconnections. Additionally, targeted advertising and messaging can be delivered to subsets of users via the predetermined server.

Having rendered a grid, step **2512**, and delivered it to the client device, the user can request datastreams, step **2518**. Any number of datastreams can be requested at once, the corresponding stream request information which defines parameters for each stream is communicated to the server.

Upon completion of these steps, the server knows the client characteristics and is able to distribute bandwidth available to the client among multiple content servers. In this example, if the client has an incoming bandwidth of say 56 Kbits/second, and is requesting datastreams from three sources with equal priority, then the server will respond for each request that a bandwidth of 56/3=18.7 Kbits/sec is available for each datastream.

The requested datastreams are displayed on the client device, step **2520** and, if the user changes the content of a tile or explicitly requests an update or refresh operation on the tile, step **2522**, an update request is sent to the server, step **2524**. Once the new content has been received, the grid is rendered anew, step **2526**.

Intensive operations on each displayed tile are also channeled through the predetermined server. For example, refresh operations on a tile generate a refresh request that is sent to the predetermined server. Similarly, requests to thumbnail a given image can be carried out, by request, on the predetermined server and the resulting compressed image transmitted to the user.

In the foregoing embodiment, the server component may reside locally on the client machine, in which case the server is known as the "Resource Manager", or it may be a remote server.

In a preferred embodiment of client server operation, shown in FIG. **26**, aspects of a user's grid profile are transmitted to third parties so that the third parties may then communicate tile based content directly to the user. For example, a user's custom grid may contain a tile that points to a third party web-site **2604**. Content **2606** from the 3^{rd} party web-site is typically transferred to the server for dissemination to the user. The server **2602** notifies the 3^{rd} party web-site that the user requires tiled data by, for example, transmitting user information **2608**. The third party then permits the tile based content of its web-site to be transmitted directly to the user.

The use of servers also allows for the latest versions of tiles to be downloaded and installed across all devices. Users are then able to share grids and tiles with other users. The server side technology utilized permits users of all client devices, from desktop PC's to mobile telephones with a consistent experience.

The majority of the server side code is written in Java, with C++ being used where necessary. Inter-server communication also utilizes XML to provide consistency with other aspects of the invention.

In a preferred embodiment, either Oracle 8i or SQL Server **2000** are used to provide a relational database (RDB) functionality of the server. Both of these RDB's now provide direct SQL to XML transformations. Databases are developed using the ANSI 92 SQL standard, which is usable by either of the RDB's.

Thin Client Technology

An object of the application program of the present invention is that it should operate on a variety of devices, including mobile telecommunications devices such as cellular phones, handheld web-browsers, palm pilots, personal digital assistants and other devices that can communicate by wireless application protocol (WAP).

Accordingly, because most handheld or mobile devices do not have the same amount of local storage and processing power as desktop computers and set-top boxes, a special version of the application program of the present invention is envisaged for mobile devices. The special version embodies so-called 'thin client' technology in which a lot of the operations are performed by a server instead of the device itself.

An "n-tier" architecture is one that is designed generically for a multitude of platforms, for example PC's, PDA's WAP phones and UNIX systems. Accordingly, in order to maximize the use of mobile devices, the application program of the present invention employs an n-tier architecture allowing the majority of the processing to be carried out on the server with the results being sent to the device for rendering. This model allows for a reduced size system to be stored on mobile devices, with the system logic residing on remote servers.

The features that are moved from client to server will be dependent upon the device in question. For example it is possible to provide the client with more features on a personal digital assistant such as a Palm pilot than on a WAP Phone.

In order to provide a level of consistency between the devices and the servers, the markup language XML is used to wrap the data that is being transmitted. As previously described, the system of the present invention uses a metabase to store information about the user's current grid and tile configurations. A synchronization procedure allows the metabase to be stored on a server and queried by any device. In this way it is possible to provide consistent grid and tile implementations independent of the device and its location.

FIG. **27** provides an example of the way in which wireless devices can interact with a server. A personal digital assistant (PDA) **2700** or a WAP phone **2702** communicates in a wireless manner with a dial-in bank **2704**. The dial-in bank communicates data to a server farm **2706** which is connected to the internet **2710**, optionally through a firewall **2708**. Alternatively, another personal digital assistant such as **2714** can communicate with an internet service provider such as **2712**, also connected to the internet **2710**. The server farm **2706** is able to provide content directly to a PDA such as **2700** or indirectly, over the internet, to a PDA such as **2714**.

What is claimed is:

1. A method executed by a device under the control of a program, said device including a memory for storing said program, said method comprising:

selecting a plurality of information sources;

partitioning a visual display of the device into an array of tiles, wherein each tile in said array of tiles is associated with an information source in said plurality of information sources;

assigning a first refresh rate to a first tile of said array of tiles and a second refresh rate to a second tile of said array of tiles;

updating information from a first information source in said plurality of information sources presented to said first tile in accordance with said first refresh rate; and simultaneously

updating information from a second information source in said plurality of information sources presented to said second tile in accordance with said second refresh rate.

2. The method of claim **1** wherein said partitioning includes partitioning said array of tiles in accordance with a user-defined array size.

3. The method of claim **1** wherein said partitioning includes partitioning said array of tiles in a non-overlapping configuration wherein each tile of said array of tiles is a uniform size and shape.

4. The method of claim **1** wherein said assigning includes assigning said first refresh rate and said second refresh rate in accordance with a first priority value of a first information source associated with said first tile and a second priority value of a second information source associated with said second tile.

5. The method of claim **1** wherein said assigning includes attributing a selected state or an unselected state to said first tile and said second tile.

6. The method of claim **1** wherein said first refresh rate is different from said second refresh rate.

7. The method of claim **1** wherein at least one of said first refresh rate and said second refresh rate is specified by a user.

8. The method of claim **1** wherein said array comprises more than one row and more than one column.

9. The method of claim **1** wherein at least one attribute of each tile in said array of tiles is assigned uniformly.

10. The method of claim **9** wherein said attribute is a refresh rate.

11. The method of claim **1** wherein said first refresh rate is assigned automatically.

12. The method of claim **1** wherein said array comprises a grid wherein each of said tiles occupies a fixed position on said grid.

13. The method of claim **1** wherein a unit tile size is associated with said array of tiles, dependent upon a maximum number of tiles displayed vertically and a maximum number of tiles displayed horizontally, and wherein each tile in said array of tiles has a fixed size that is equal to said unit tile size, or a multiple thereof.

14. The method of claim **1** wherein said array is displayed in a web-browser.

15. The method of claim **1** additionally comprising storing said array of tiles on a second device.

16. The method of claim **1** wherein said array of tiles is retrieved from a second device.

17. The method of claim **1** wherein said device is a mobile telephone.

18. The method of claim **1** wherein said device is a television.

19. The method of claim **1** wherein said device is a computer.

20. The method of claim **1** wherein said device is a personal digital assistant.

21. The method of claim **1** wherein said plurality of information sources comprises at least two information sources selected from the group consisting of: analog signal; video signal; audio signal; text; an e-mail program; a world-wide-web page; streaming video; streaming audio; and graphics.

22. An electronic readable memory to direct an electronic device to function in a specified manner, comprising:

a first set of instructions to control simultaneous communication with a plurality of information sources;

a second set of instructions to arrange a display into an array of tiles;

a third set of instructions to associate a first information source of said plurality of information sources to a first tile of said array of tiles and a second information source of said plurality of information sources to a second tile of said array of tiles;

a fourth set of instructions to retrieve information from said first information source in accordance with a first retrieval rate and retrieve information from said second information source in accordance with a second retrieval rate; and

a fifth set of instructions to present information to said first tile in accordance with said first retrieval rate and present information to said second tile in accordance with said second retrieval rate.

23. The electronic readable memory of claim **22** further comprising a set of instructions to process a network datastream from a network data source.

24. The electronic readable memory of claim **22** further comprising a set of instructions to process a tuner signal from a tuner device.

25. The electronic readable memory of claim **22** wherein said second set of instructions arrange said array of tiles on a display of a mobile telephone.

26. The electronic readable memory of claim **22** wherein said second set of instructions arrange said array of tiles on a display of a television.

27. The electronic readable memory of claim **22** wherein said second set of instructions arrange said array of tiles on a display of a computer.

28. The electronic readable memory of claim **22** wherein said second set of instructions arrange said array of tiles on a personal digital assistant.

29. The electronic readable memory of claim **22** further comprising a set of instructions to assign a password to a selected tile of said array of tiles.

30. The electronic readable memory of claim **22** wherein said second set of instructions produce an array of non-overlapping tiles wherein each tile has a uniform size and shape.

31. The electronic readable memory of claim **22** wherein said fourth set of instructions assign said first retrieval rate and said second retrieval rate in accordance with a predetermined priority scheme.

32. The electronic readable memory of claim **22** further comprising a set of instructions to selectively assign a selected or unselected state to specified tiles of said array of tiles.

33. The electronic readable memory of claim **22** further comprising a set of instructions to interrupt said first retrieval rate and present said first tile with static information from said first information source.

34. The electronic readable memory of claim **22** further comprising a set of instructions to deliver selected textual content from said first information source to said first tile.

35. The electronic readable memory of claim **22** further comprising a set of instructions to deliver a video signal to a selected tile of said array of tiles.

36. The electronic readable memory of claim **22** further comprising a set of instructions to deliver a frame of a broadcast TV signal to a selected tile of said array of tiles.

37. The electronic readable memory of claim **22** further comprising a set of instructions to deliver information from a network document to a selected tile of said array of tiles.

38. The electronic readable memory of claim **22** further comprising a set of instructions to deliver a web page to a selected tile of said array of tiles.

39. The electronic readable memory of claim **22** wherein said first retrieval rate is different from said second retrieval rate.

40. The electronic readable memory of claim **22** wherein said array comprises more than one row and more than one column.

41. The electronic readable memory of claim **22** further comprising a set of instructions for uniformly assigning at least one attribute of each tile in said array of tiles.

42. The electronic readable memory of claim **41** wherein said attribute is a retrieval rate.

43. The electronic readable memory of claim **22** wherein said first retrieval rate is assigned automatically.

44. The electronic readable memory of claim **22** additionally comprising instructions for storing said array of tiles on a second electronic device.

45. The electronic readable memory of claim **22** additionally comprising instructions for retrieving said array of tiles from a second electronic device.

46. A system for facilitating the organization and management of multiple data sources, comprising:

a device that includes a processor configured to execute instructions, a memory connected to said processor to store at least one program that includes a graphical user interface, and an input device, wherein said processor executes instructions to:

control simultaneous communication with a plurality of information sources;

arrange a display into an array of tiles;

associate a first information source of said plurality of information sources to a first tile of said array of tiles and a second information source of said plurality of information sources to a second tile of said array of tiles;

retrieve information from said first information source in accordance with a first retrieval rate and retrieve information from said second information source in accordance with a second retrieval rate; and present information to said first tile in accordance with said first retrieval rate and present information to said second tile in accordance with said second retrieval rate.

47. The system of claim **46** wherein said array comprises more than one row and more than one column.

48. The system of claim **46** wherein said processor additionally executes instructions for uniformly assigning at least one attribute of each tile in said array of tiles.

49. The system of claim **48** wherein said attribute is a retrieval rate.

50. The system of claim **46** wherein said first retrieval rate is assigned automatically.

51. The system of claim **46** wherein said processor additionally executes instructions for storing said array of tiles on a second device.

52. The system of claim **46** wherein said processor additionally executes instructions for retrieving said array of tiles from a second device.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 6,724,403 B1 Page 1 of 1
DATED : April 20, 2004
INVENTOR(S) : Ovid Santoro et al.

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Title page,
Item [*] Notice, “Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 471 days” should read -- Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 379 days. --.

Signed and Sealed this

Twenty-ninth Day of November, 2005

Jon W. Dudas

JON W. DUDAS
Director of the United States Patent and Trademark Office

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO.	: 6,724,403 B1	Page 1 of 1
DATED	: April 20, 2004	
INVENTOR(S)	: Ovid Santoro et al.	

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Title page,
Item [*] Notice, “Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 259 days” should read -- Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 379 days. --.

This certificate supersedes Certificate of Correction issued November 29, 2005.

Signed and Sealed this

Ninth Day of May, 2006

Jon W. Dudas

JON W. DUDAS
Director of the United States Patent and Trademark Office

APPENDIX J

US005241303A

United States Patent [19]

Register et al.

[11] **Patent Number: 5,241,303**

[45] **Date of Patent: Aug. 31, 1993**

[54] **PORTABLE COMPUTER WITH PHYSICAL RECONFIGURATION OF DISPLAY CONNECTION FOR STYLUS AND KEYBOARD ENTRY**

[75] Inventors: **David S. Register; J. Michael O'Dell,** both of Austin; **Robert Groover, III,** Dallas, all of Tex.

[73] Assignee: **Dell USA, L.P.,** Austin, Tex.

[21] Appl. No.: **814,338**

[22] Filed: **Dec. 26, 1991**

[51] **Int. Cl.**^{5} **G09G 3/02**

[52] **U.S. Cl.** **340/706**; 340/711; 340/712; 178/19

[58] **Field of Search** 340/700, 706, 711, 712; 364/708; 178/18, 19; 361/390, 391, 392, 393

[56] **References Cited**

U.S. PATENT DOCUMENTS

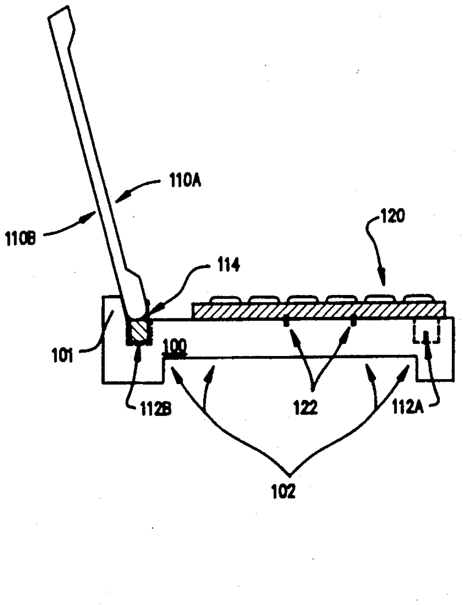
4,926,010	5/1990	Citron	178/18
4,937,563	7/1990	Shekita et al.	340/711
5,049,862	9/1991	Dao et al.	340/726
5,107,402	4/1992	Malgouires	364/708

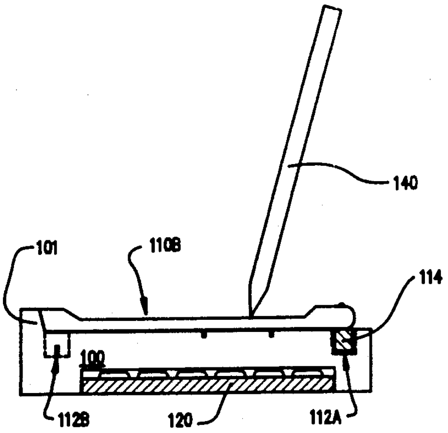
Primary Examiner—Alvin E. Oberley
Assistant Examiner—Xiao M. Wu
Attorney, Agent, or Firm—Thomas G. Devine; James W. Huffman

[57] **ABSTRACT**

A computer system which is reconfigurable to provide separate ergonomically advantageous positions for keyboard input and for stylus input. A primary system chassis contains a bay in its underside where a detachable keyboard can be stored. For one-hand stylus input, the keyboard is left in its bay while the display is mounted flat on top of the system chassis. For keyboard input, the keyboard is mounted on the system chassis, and the display is supported at an angle which makes it easily visible to a user typing on the keyboard.

16 Claims, 7 Drawing Sheets





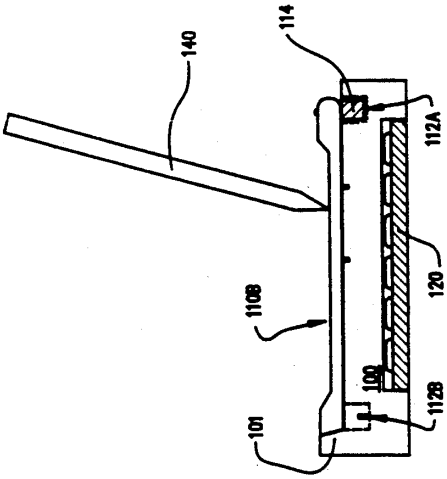


FIG. 1B
STYLUS-ENTRY
POSITION

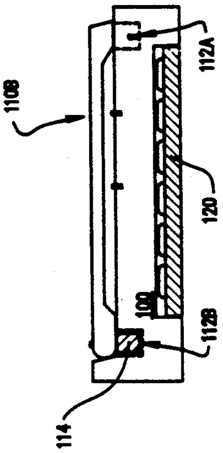


FIG. 1C
CLOSED POSITION

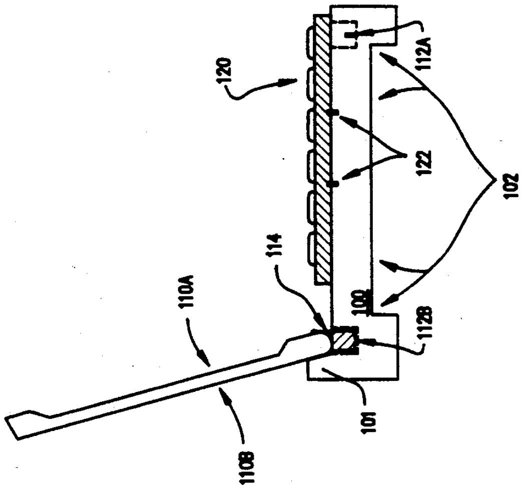


FIG. 1A
TYPING POSITION

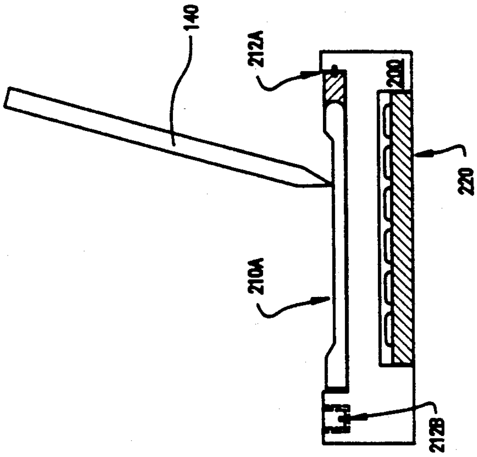


FIG. 2B
STYLUS-ENTRY
POSITION

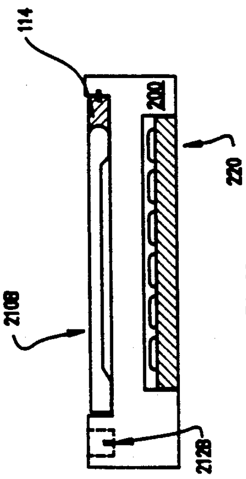


FIG. 2C
CLOSED POSITION

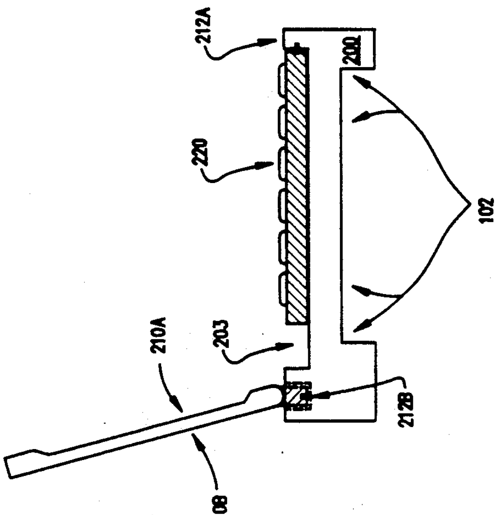


FIG. 2A
TYPING POSITION

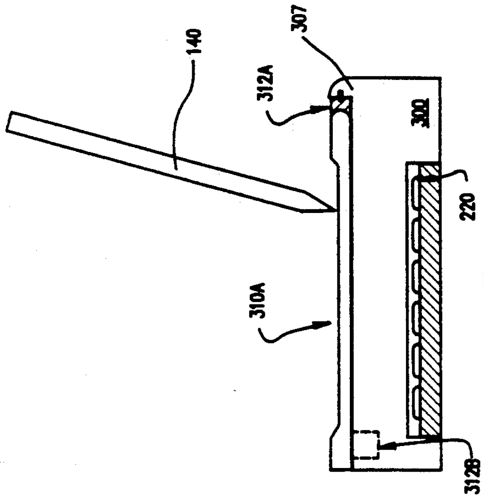


FIG. 3B
STYLUS-ENTRY POSITION

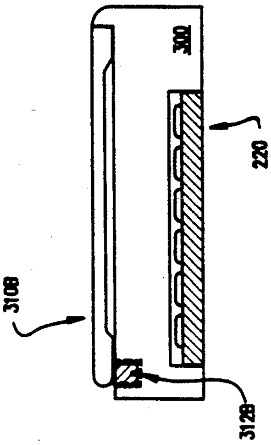


FIG. 3C
CLOSED POSITION

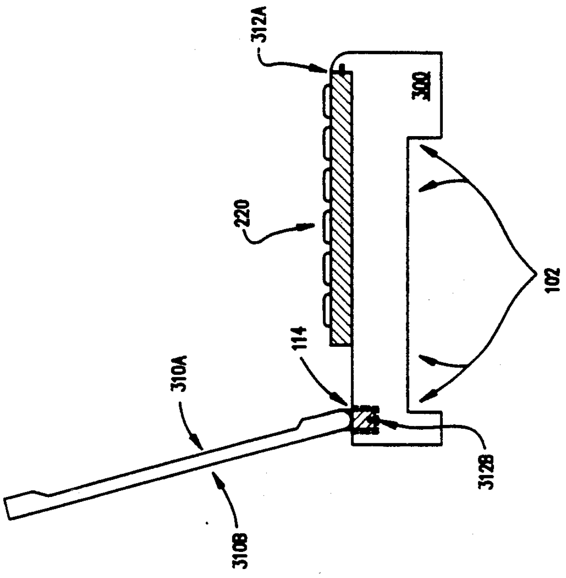


FIG. 3A
TYPING POSITION

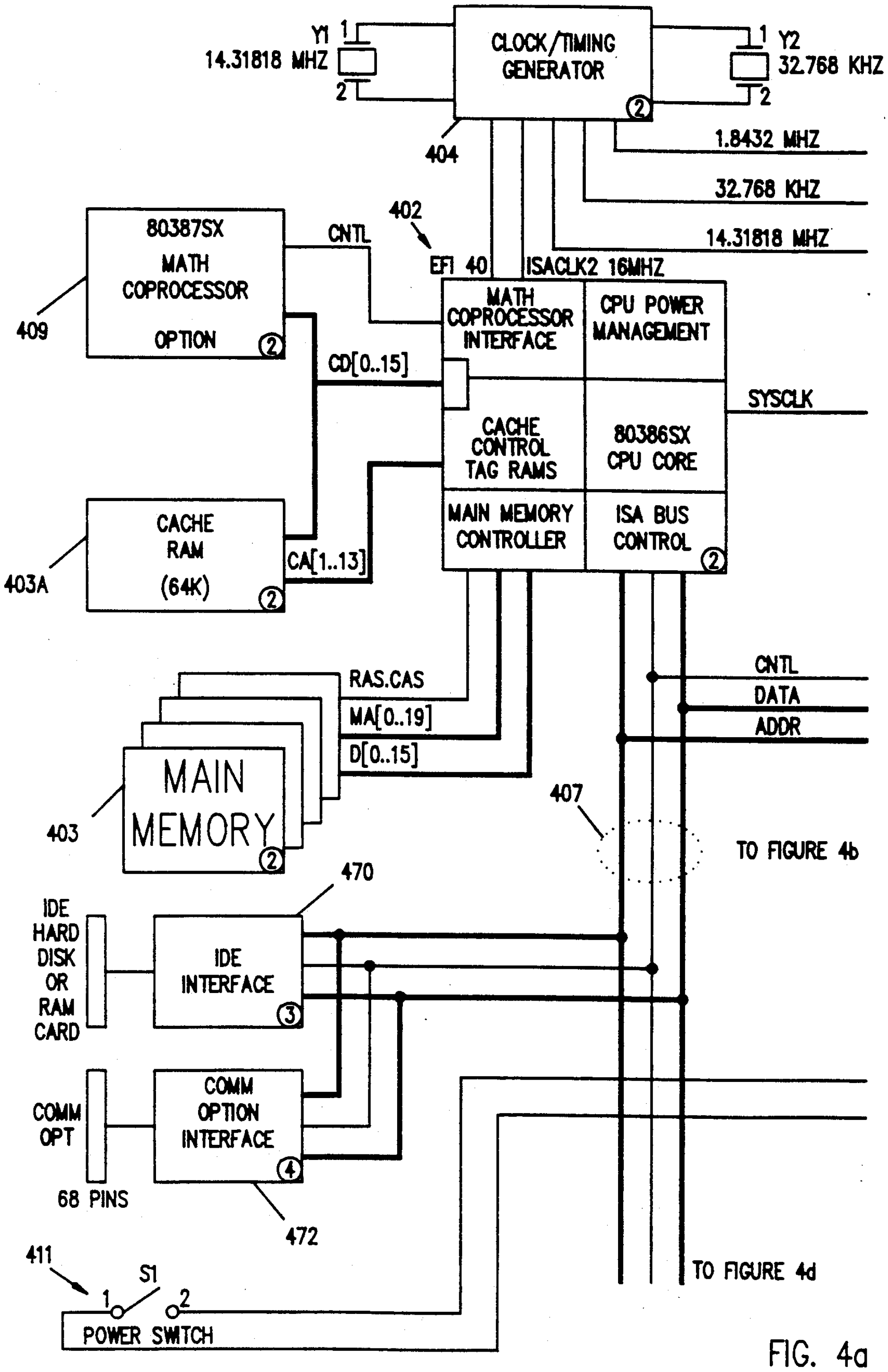

Y1
14.31818 MHZ
1
2
CLOCK/TIMING GENERATOR
2
Y2
32.768 KHZ
1
2
404
1.8432 MHZ
32.768 KHZ
14.31818 MHZ
402
80387SX MATH COPROCESSOR OPTION
2
409
CNTL
EFI 40
ISACLK2 16MHZ
MATH COPROCESSOR INTERFACE
CPU POWER MANAGEMENT
CD[0..15]
SYSCLK
CACHE CONTROL TAG RAMS
80386SX CPU CORE
CACHE RAM (64K)
2
403A
CA[1..13]
MAIN MEMORY CONTROLLER
ISA BUS CONTROL
2
RAS.CAS
MA[0..19]
D[0..15]
CNTL
DATA
ADDR
MAIN MEMORY
2
403
407
TO FIGURE 4b
470
IDE HARD DISK OR RAM CARD
IDE INTERFACE
3
COMM OPT
COMM OPTION INTERFACE
4
68 PINS
472
411
S1
1
2
POWER SWITCH
TO FIGURE 4d

FIG. 4a

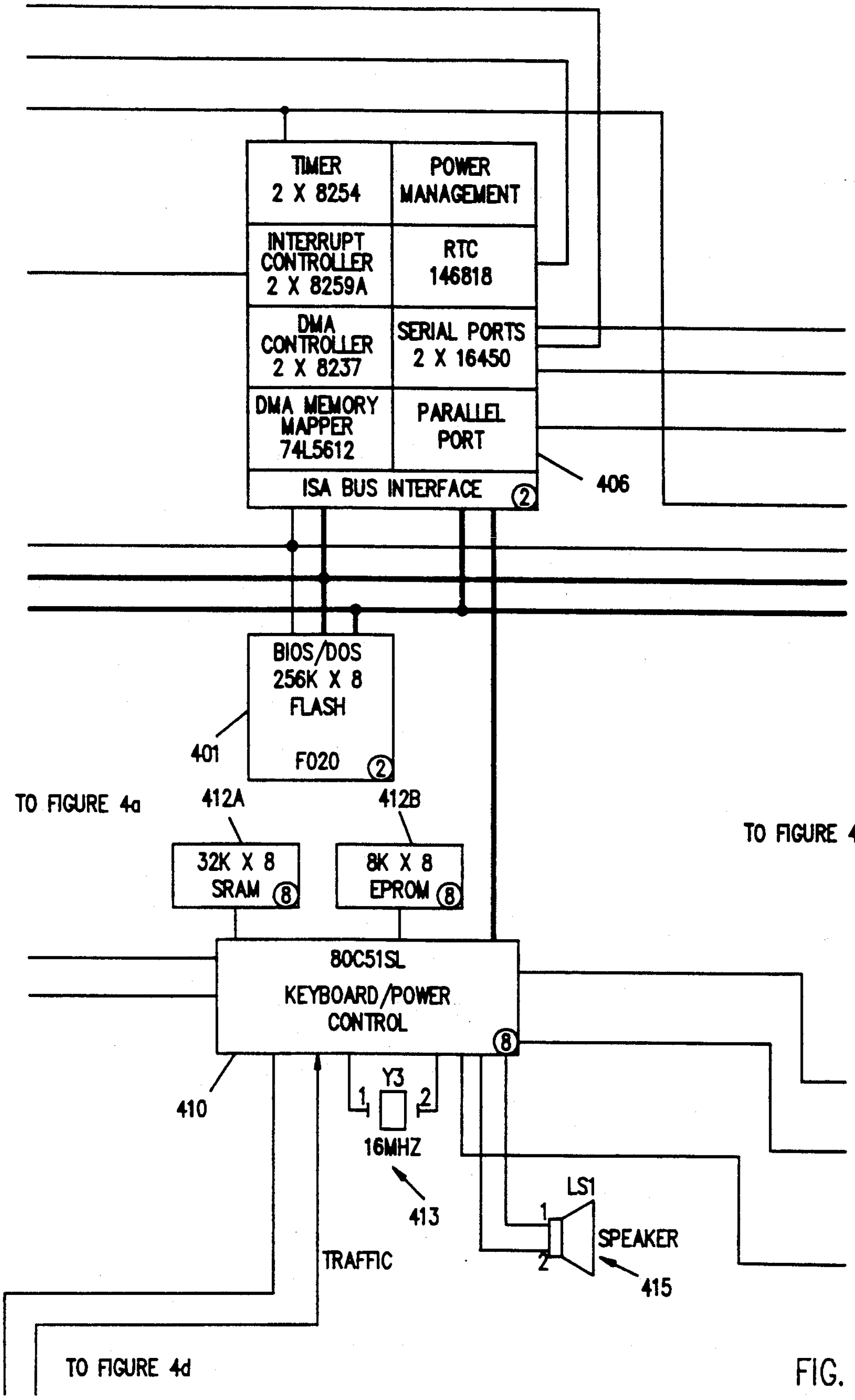

TIMER
2 X 8254
POWER
MANAGEMENT
INTERRUPT
CONTROLLER
2 X 8259A
RTC
146818
DMA
CONTROLLER
2 X 8237
SERIAL PORTS
2 X 16450
DMA MEMORY
MAPPER
74LS612
PARALLEL
PORT
ISA BUS INTERFACE
2
406
BIOS/DOS
256K X 8
FLASH
F020
2
401
TO FIGURE 4a
412A
412B
TO FIGURE 4c
32K X 8
SRAM
8
8K X 8
EPROM
8
80C51SL
KEYBOARD/POWER
CONTROL
8
410
Y3
1
2
16MHZ
413
LS1
1
2
SPEAKER
415
TRAFFIC
TO FIGURE 4d

FIG. 4b

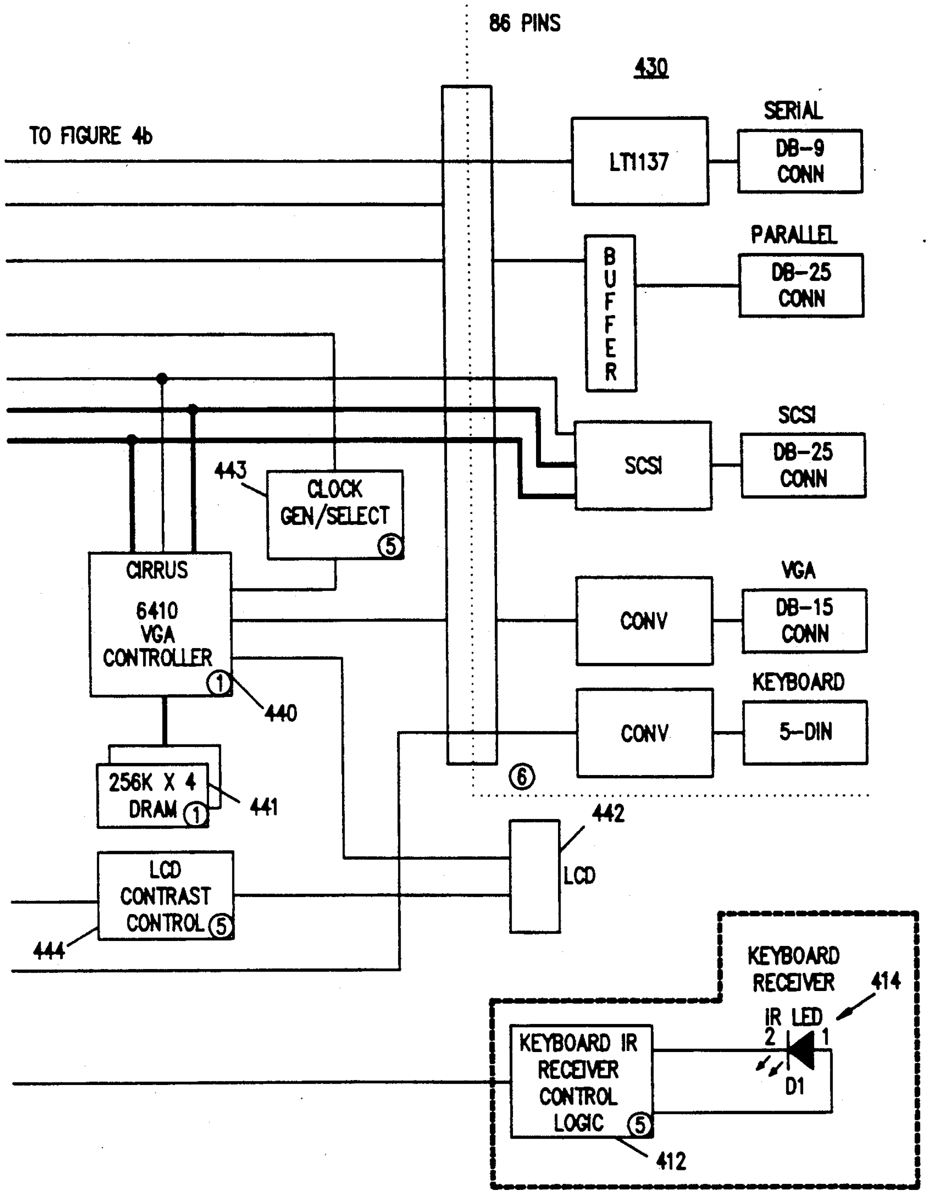

86 PINS
430
TO FIGURE 4b
LT1137
SERIAL
DB-9 CONN
BUFFER
PARALLEL
DB-25 CONN
SCSI
SCSI
DB-25 CONN
443
CLOCK GEN/SELECT 5
CIRRUS 6410 VGA CONTROLLER 1
440
CONV
VGA
DB-15 CONN
KEYBOARD
CONV
5-DIN
256K X 4 DRAM 1
441
6
442
LCD
LCD CONTRAST CONTROL 5
444
KEYBOARD RECEIVER
414
IR LED
2
1
D1
KEYBOARD IR RECEIVER CONTROL LOGIC 5
412

FIG. 4c

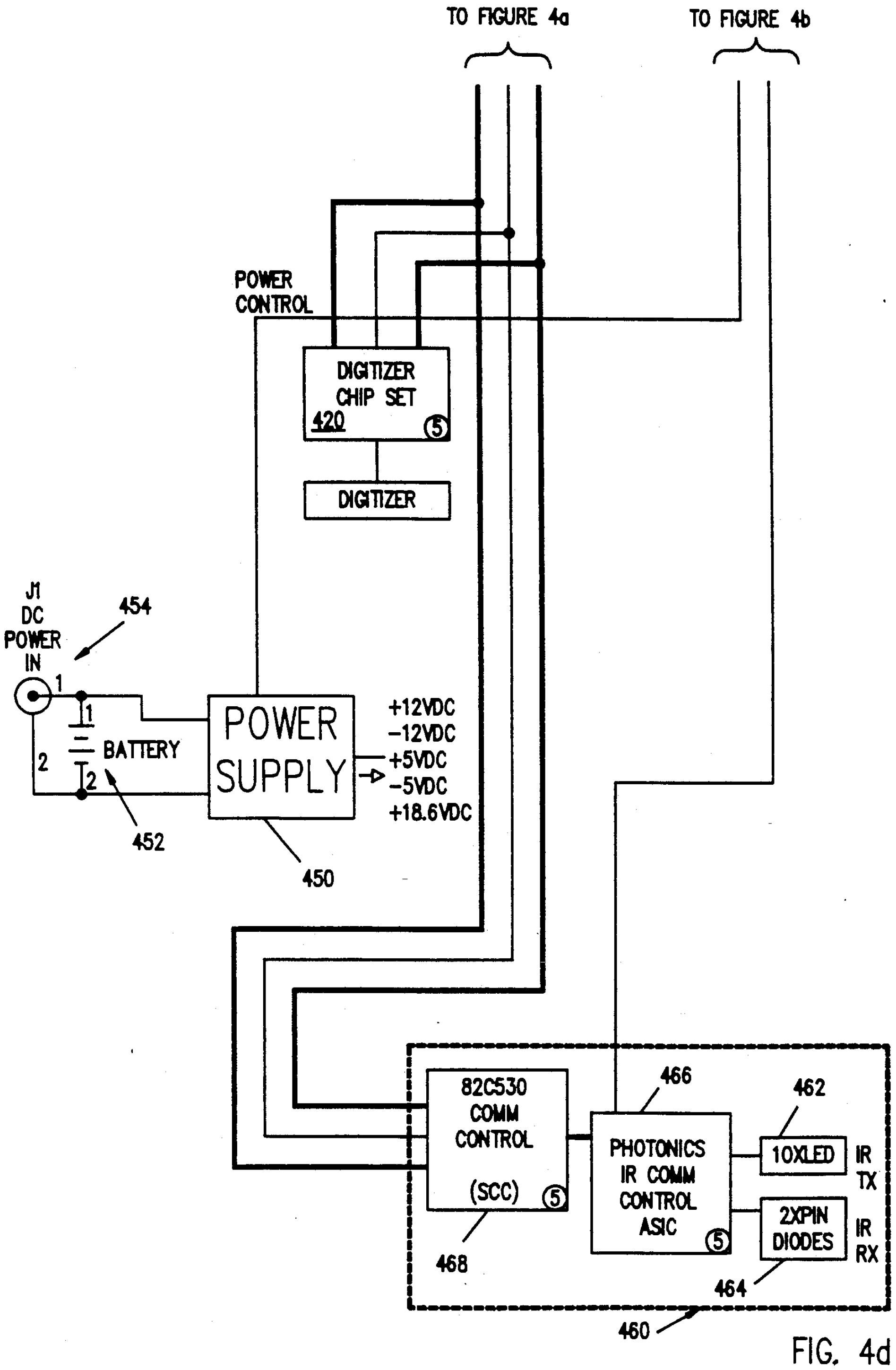

TO FIGURE 4a
TO FIGURE 4b
POWER CONTROL
DIGITIZER CHIP SET
420
5
DIGITIZER
J1
DC
POWER
IN
454
1
2
1
2
BATTERY
452
POWER SUPPLY
450
+12VDC
-12VDC
+5VDC
-5VDC
+18.6VDC
82C530
COMM
CONTROL
(SCC)
5
468
466
PHOTONICS
IR COMM
CONTROL
ASIC
5
462
10XLED
IR
TX
2XPIN
DIODES
IR
RX
464
460

FIG. 4d

PORTABLE COMPUTER WITH PHYSICAL RECONFIGURATION OF DISPLAY CONNECTION FOR STYLUS AND KEYBOARD ENTRY

PARTIAL WAIVER OF COPYRIGHT

All of the material in this patent application is subject to copyright protection under the copyright laws of the United States and of other countries. As of the first effective filing date of the present application, this material is protected as unpublished material.

Portions of the material in the specification and drawings of this patent application are also subject to protection under the maskwork registration laws of the United States and of other countries.

However, permission to copy this material is hereby granted to the extent that the owner of the copyright and maskwork rights has no objection to the facsimile reproduction by anyone of the patent document or patent disclosure, as it appears in the United States Patent and Trademark Office patent file or records, but otherwise reserves all copyright and maskwork rights whatsoever.

CROSS-REFERENCE TO OTHER APPLICATIONS

The following applications of common assignee each contain at least one drawings in common with the present application, and are believed to have effective filing dates identical with that of the present application, and are all hereby incorporated by reference:

Ser. No. 07/814,028, filed Dec. 26, 1991, entitled "Portable Computer with Screen Reversible for Stylus and Keyboard Entry";

Ser. No. 07/814,733, filed Dec. 26, 1991, entitled "Stylus-Operable Computer with Wireless Keyboard in Storage Bay";

Ser. No. 07/814,732, filed Dec. 26, 1991, entitled "Stylus-Operated Computer with Folding Cover Convertible to Display Stand";

all of which are hereby incorporated by reference.

BACKGROUND AND SUMMARY OF THE INVENTION

The present invention relates to computer systems (and particularly to small computer systems), and to methods for using them.

The innovations disclosed in the present application provide computer systems (especially very small portable personal computers) which have advantageous new capabilities for both keyboard and stylus input. To better explain the significance and advantages of these innovations, the following paragraphs (down to page 9) will review some technological context. This technological context is not necessarily prior art, but is intended to help in pointing out the disclosed inventions.

Stylus-Operable Computers[1]

As portable computers have continued to shrink, continuing efforts have been made to identify a new input channel to replace the keyboard. A great deal of design effort was needed to design notebook computers with a compacted keyboard which still had the key spacing to permit touch typing. However, at sizes below the "notebook" computer, there is no simply no room for a keyboard with standard key spacing and number of characters.

Thus, in the 1990s a great deal of acitivity has been devoted to computer systems which can be operated by a user using a stylus as a primary input device. Such an input device has many advantages: it is inherently very well suited to menu-based command inputs. It is inherently well suited to use with a small screen, since whatever screen area is available can be allocated among various boxes which can be checked or written in by the user. It is also inherently well suited to use by marginally literate persons, and to rapid input in environments where a user's full intention is not available, since the interface is inherently well-suited to graphical rather than character-based input.

In order to realize the full potential of stylys entry as an this input channel, some degree of recognition capability for handwritten stylus inputs is necessary. It appears that the processor capability, power-management capability, and display and touch screen technology required for such computers is now available. Large improvements in operating system software technology in this area are to be expected, but a first generation of functional system software is already available. However, a great deal of improvement remains for making this class of computers adequately comfortable and user-friendly.

Thus, as available computing power increases, stylus-capable computers (using recognition of handwritten inputs by a stylus acting upon the display surface) are becoming more practical.

A number of companies have recently announced pen-based computer products. See, for example, Shipley, "Pen-based PCs ready for prime time," in PC-COMPUTING, vol. 4 no. 11 (Nov. 1991), at 214 ff, which is hereby incorporated by reference. This and other articles[2] discuss machines announced by vendors including Tusk, Momenta, Grid, NCR, Samsung, Pi Systems, TelePad Inc. and other vendors.

In particular, portable computer applications are attractive candidates for stylus input systems, as mobile users often must enter data while standing, holding the computer unit in one hand, and the stylus in the other. For example, it has been suggested that a small portable dedicated computer, for specialized data entry or control applications, could advantageously be configured so that the user could hold the unit in one hand while marking on it with a stylus in the other hand.

Touch-Screen Menu Selection

An older system architecture, which attempted to fill some of these needs, used touch-screen input for menu selection. For example, a user would be should a menu with 8 boxes on the screen; and when the user touched one box, a new menu (with a new set of options depending on which box the user had previously touched) would appear.

For special applications, where custom software can be developed, such menus offer rapid access to a range of commands. Moreover, such systems provide a simple interface and compact size, but are inherently very

[1] Stylus-input computers are also commonly referred to as "pen-based" computers, or "slate" computers.

[2] See also Schroeder et al., "Momenta to head parade of pen PCs: firm to bundle new machine with proprietary OS, applications," in PC WEEK vol. 8 no. 39 (Sept 30, 1991), at page 4; Davis, "NCR's pen-based PC signals the birth of a new market," in PC WEEK vol. 8 no. 27 (Jul. 8, 1991) at page 134; Catchings et al., "NCR NotePad delivers 3-in-1 pen computing," in PC WEEK vol. 8 no. 26 (Jul. 1, 1991) at pages 1–2; all of which are hereby incorporated by reference.

limited in their ability to deviate from a standard menu sequence, or to accept unexpected user input. Thus, such a user interface is not very suitable to a general purpose computer.

It should also be noted that the technology of such simple touch-screen computers differs very substantially from that of the stylus-operated computers which are now beginning to appear. A general-purpose stylus-operated computer must have some capability for recognizing handwriting (at least handprinted letters, if not script). Thus, a relevant hardware parameter is the dimensional resolution of the touch-screen input: the spatial resolution needed for touch menu selection can be as coarse as a centimeter or more, whereas the spatial resolution needed for recognition of handwritten inputs should probably be 200 or more per inch.

Handwriting Recognition

Automatic recognition of handwritten inputs is an area which has been the subject of significant research effort in a large-computer environments. Sophisticated algorithms for handwriting recognition have been developed, but previously there was no driving application to migrate these algorithms to microcomputer-based hardware.

Stylus-Capable Operating Systems

Of course, pen-sensitive hardware would not be useful without appropriate software. Specifically, a viable pen computer market requires system software which can handle a user's pen inputs and pass them in a standard fashion to the application software.

This has been a very active areas of development in the 1990s. Currently the leading pen-based operating systems are PenPoint TM from Go Corp. and PenWindows TM from MicroSoft TM, but further rapid developments in this area are to be expected.

The Stylus

A variety of technologies can be used to allow the computer system to sense the position of the stylus point[3], but the choice of those technologies is not particularly relevant to the present invention.

Light Pen

Computer researchers have recognized for many years that stylus input, onto a display screen which was also a stylus-sensitive input screen, was an attractive input technology. Thus, for example, as early as the 1960s some large high-cost computers used a "light pen" interface, where the computer could sense position of a stylus which the user held up to the screen.[4] However, a protracted session of light pen use on a large vertical display can rapidly become extremely uncomfortable to the user's arms and shoulders. Thus, light pen technology never achieved widespread use.

Combined Stylus and Keyboard Input

It is generally acknowledged however, that a conventional keyboard is a faster means of data entry provided that both hands are free to type and the keyboard and display are in an ergonomically acceptable configuration. Thus, a computer which is easily converted from a dedicated clipboard-type stylus input configuration into a conventional clamshell keyboard input notebook configuration, and vice versa, is desirable.

Several attempts have been made to address the problem of both keyboard and stylus input. (Examples of such efforts include the Tusk and Momenta products in the articles cited above.)

One attempt which has been made to reconcile these demands was to provide a clipboard-type computer with a remote keyboard connected by an electrical cable. This structure is believed to be incorporated, for example, in the Momenta and Grid computers now available. The disadvantage of this is that the keyboard is not integral to computer, thus requiring the carrying and storage of an additional system component if keyboard data entry is anticipated. However, an advantage of this approach is that, if keyboard data entry is never needed, the system unit may be smaller and lighter.

Another suggestion was a clipboard-type computer which uses software to generate an image of a keyboard on the display. The user activates specific keystrokes by touching or tapping the display surface with the stylus. (This structure is believed to be incorporated, for example, in the Grid computer now available.) The disadvantage of this is that it is not compatible with touch-typing, and thus gives slow character entry speed. However, an advantage is that the simulated keyboard is integral with the system, requires no additional components, and does not any size or weight.

Ergonomics

Mobile users often must enter data while standing, holding the computer unit in one hand (with the display exposed and firmly supported), and the stylus in the other. In a sitting position, it should also be possible to use the stylus with one hand while balancing the computer on the user's knee, leaving one hand free.

On the other hand, a palmtop computer used for typing must meet the same criteria as any other key-board-entry device: the typing position should permit a seated user to keep both hands on the keyboard, and to see the display clearly, without strain in wrists, upper back, or neck.

The question is how the capability for both stylus input and keyboard entry can be achieved. It is difficult enough to achieve any sort of keyboard input capability, in a very small portable computer, which is adequate for rapid typing. It is even more difficult to combine this input with a capability for stylus input.

A particular problem is the ergonomics of display access. When the user is using the stylus, the display screen unit should ideally be thin and approximately flat (so the user can hold it in one hand or balance it on a knee); but when the user is using the keyboard, the unit should be supported in a position and configuration which makes the screen easily readable (i.e. 50 to 90 degrees from horizontal). No known design has provided a stand which is stable and provides this support angle and does not detract from the portability of the computer.

[3] Penpoint technologies may use an "active" stylus, which contains electronic circuits, or a purely passive stylus, which simply functions as a conductive element to contact a matrix of conductors, or even simply as a mechanical element to contact a 2-dimensional-sensing surface at a certain point. An active stylus may be cabled to the computer chassis or may be wireless.

[4] With a raster-scanned CRT display, this can be accomplished by connecting a photo-detector to sense illumination at the tip of the stylus. By looking for a pulsed component at the frame scan frequency, and then finding the phase of these pulses with respect to the vertical and horizontal blanking intervals, the X-Y position of the stylus within the raster scan field can be directly determined. The light pen could be. for example, simply an optical fiber connected to a transparent stylus tip, with the optical fiber plugged into a photo-detector on the computer chassis (or terminal chassis).

In particular, such a computer would typically be used in a very confined spaces (such as the snack tray of a coach airline seat). In such spaces, minimizing footprint is a key consideration. Ideally such a stand should not increase the computer's footprint at all, whether the stand is folded up or in use.

Innovative Computer System and Method

The present invention discloses a new way to permit both keyboard and stylus input in a very small compact computer.

The present application teaches a novel computer system which provides convenient and ergonomically advantageous capability for both keyboard input and stylus input. A primary system chassis contains the CPU and the power supply, and other key components, and also contains a bay in its underside where a detachable keyboard can be stored. To allow the keyboard to be as wide as possible, this bay preferably runs across the full width of the computer, and is enclosed on only two sides. A mechanical retainer holds the keyboard in this bay, with its keys protected, until the user needs to configure the computer for keyboard input. A separate detachable display module includes the display and a hinged connector. The display and keyboard can dock to the primary system chassis in different orientations, to produce at least three possible operating and/or transport configurations:

In a first position, the keyboard is mounted on the top of the chassis. The display module is plugged into a first connector at the edge of the chassis, and is folded backwards (from the keyboard location) so that a user can place the computer on a desk in front of him, and type on the keyboard with both hands while looking at the display. In this configuration, the portable computer of the present invention resembles a conventional notebook computer.

In a second position, the display module is mounted on top of the system chassis with its display side outward, while the keyboard is safely stored in its bay. In this configuration, the computer provides a small unit which can be carried in one hand, with the display/touch-screen exposed. Thus, the user can hold it in one hand and wield the stylus in the other, as is desirable for a stylus-operated computer.

In a third position, the display is folded flat against the system chassis, with its back (nondisplay) side facing outward, while the keyboard is safely stored in its bay. This position provides a conveniently stable and durable closed position for carrying the computer.

Three main classes of embodiments are disclosed. The first class of embodiments is shown in FIGS. 1A-1C, the second class of embodiments is shown in FIGS. 2A-2C, and the third class of embodiments is shown in FIGS. 3A-3C.

As the display element and keyboard are both detachable, a means of electrical interconnect is required. For the display element, the interconnect is effected by means of a multi-pin connector mounted in a housing which rotates approximately 120 degrees about an axis parallel to the attachment edge of the display element. The rotating connector provides both the system adaptability as well as the freedom of motion necessary for display element angle adjustment.

The keyboard interconnect is provided by a multi-pin connector on the bottom surface of the keyboard housing mating to a corresponding connector on the top surface of the system.

The system's center of gravity is not moved substantially by reconfiguration. The heavy components (battery, main circuit board, and disk drive if present) are contained within the system base. Thus, the system remains stable in the keyboard input configuration, despite the unbalancing effect of the display element.

BRIEF DESCRIPTION OF THE DRAWING

The present invention will be described with reference to the accompanying drawings, which show important sample embodiments of the invention and which are incorporated in the specification hereof by reference, wherein:

FIGS. **1A-1C** show a first computer system embodiment, in which the chassis includes a berm to support the display module in the typing position. FIG. **1A** shows this system configured for keyboard input, FIG. **1B** shows this system configured for stylus input, and FIG. **1C** shows this system in a closed position for storage or transport.

FIGS. **2A-2C** show a second computer system embodiment, in which the chassis includes a lateral connector in a topside recess. FIG. **2A** shows this system configured for keyboard input, FIG. **2B** shows it configured for stylus input, and FIG. **2C** shows it configured in a closed position for storage or transport.

FIGS. **3A-3C** show a third computer system embodiment, in which the chassis includes lateral contacts in a raised front lip. FIG. **3A** shows this system configured for keyboard input, FIG. **3B** shows it configured for stylus input, and FIG. **3C** shows it configured in a closed position for storage or transport.

FIG. **4** shows a block diagram of the electronic organization of a sample pen-and keyboard-operated computer according to FIGS. **1**, **2**, or **3**.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

The numerous innovative teachings of the present application will be described with particular reference to the presently preferred embodiment. However, it should be understood that this class of embodiments provides only a few examples of the many advantageous uses of the innovative teachings herein. In general, statements made in the specification of the present application do not necessarily delimit any of the various claimed inventions. Moreover, some statements may apply to some inventive features but not to others.

First Class of Embodiments

FIGS. **1A-1C** show a first embodiment of the computer system of the presently preferred embodiment. In this embodiment the chassis **100** includes a berm **101**, which helps to support the display module **110** in the typing position (shown).

FIG. **1A** shows the system of the first embodiment configured for keyboard input. Two multipin connector sockets **112A** and **112B** provide alternative connection positions for the display module **110**.

In the typing positions shown, the display module **110** is plugged into the rear connector socket **112B**, so that the display side **110A** of the display module is readily visible to the user. The display side **110A** of the display module in the presently preferred embodiment, also includes sensing circuitry for detection of the position of a stylus. (The primary position for stylus use is as shown in FIG. **1B**, but it is preferred (although not strictly necessary) that the stylus can also be used in the position of FIG. 1A. Thus, this position permits stylus input and keyboard input to be combined.)

The keyboard **120** is mounted atop the chassis **100**, accessible for touch-typing, and mates with signal pins **122** on the top surface of the chassis. In this Figure, note that the bay **102** is visible. This is simply a cavity in the bottom of the chassis **100**, which is sized to fit the keyboard **120**. Note that, in the presently preferred embodiment, this cavity extends all the way through the chassis **100** from side to side. This permits the keyboard to be as wide as possible, consistent with the overall width of the chassis **100**.

FIG. 1B shows the system of the first embodiment configured for stylus input. In the stylus-entry position shown, the display module **110** is plugged into the front connector socket **112A**, and folded back against the chassis **100**, so that the display side **110A** of the display module is readily visible to the user and readily accessible to the user's stylus **140**. (Comparison of FIGS. 1A and 1B shows how the hinged portion **114** of the display module is movable.) The keyboard **120** is mounted in bay **102** of chassis **100**, for safe storage. (A mechanical clip, not shown, prevents the keyboard from falling out.) The stylus **140**, in the presently preferred embodiment, is an active stylus which is wired to its own connector in the chassis, and which docks in a hole in the chassis. (Note that the display module **110** of this embodiment preferably includes a slope in one edge which is complementary to the slope of the inner edge of berm **101**.)

FIG. 1C shows the system of the first embodiment in a closed position for storage or transport. In this position, as in the position of FIG. 1A, the hinged portion **114** of display **110** is inserted into the rear connector **112B**. However, in this position the display module **110** is folded forward, against the top surface of chassis **100**, to protect the display side **110A**. The exposed back side **110B** is simply blank plastic.

Second Class of Embodiments

FIGS. 2A-2C show a second embodiment of the computer system of the presently preferred embodiment. Note that, in this embodiment, the chassis **200** includes a recess **203** in its topside into which keyboard **220**, or display module **210**, can be docked. The recess includes a lateral connector **212A**, which can mate either to the keyboard or to the display module. (Thus, the keyboard **220** of this embodiment has a different connector geometry from the keyboard **120** of the first embodiment.)

FIG. 2A shows the system of the second embodiment configured for keyboard input. Keyboard **220** is housed in recess **203** (where it is accessible for touch-typing), and is docked to connector **212A**. In the typing position shown, the display module **210** is plugged into the rear connector socket **212B**, so that the display side **210A** of the display module **210** is readily visible to the user. (The display module **210** of this embodiment differs from display module **110** of the first embodiment in being shaped to fit within the recess **203**, and in not having a sloped edge.) The display side **210A** of the display module **210**, in the presently preferred embodiment, also includes sensing circuitry for detection of the position of a stylus.

FIG. 2B shows the system of the second embodiment configured for stylus input. In this position, keyboard **220** is stored in bay **102**. Display module **210** lies flat in recess **203** (where it is readily visible to the user, and is accessible for stylus entry), and is connected, through its movable portion **114**, to connector **212A**.

FIG. 2C shows the system of the second embodiment configured in a closed position for storage or transport. In this position, as in the position of FIG. 2B, the hinged portion **114** of display **110** is inserted into the lateral front connector **212A**; but in this position the connection is simply for physical security, and to protect the connector. In this position the display module **210** is face down against the top surface of chassis **100**, protecting the display side **210A**. The exposed back side **210B** is simply blank plastic.

Third Class of Embodiments

FIGS. 3A-3C show a third embodiment of the computer system of the presently preferred embodiment. In this embodiment, the chassis includes a raised front lip **307** with a lateral contact **312A** on the inner edge thereof.

FIG. 3A shows the system of the third embodiment configured for keyboard input. (The display module **310** of this embodiment, unlike display module **210** of the second embodiment, does not have to fit within recess **203**.) The display side **310A** of the display module **310**, in the presently preferred version of this embodiment, also includes sensing circuitry for detection of the position of a user stylus **140**.

FIG. 3B shows the system of the third embodiment configured for stylus input. In this position, keyboard **220** is stored in bay **102**. Display module **310** lies flat on the top side of chassis **300** (where it is readily visible to the user, and is accessible for stylus entry), and is connected, through its movable portion **114**, to lateral connector **312A**.

FIG. 3C shows the system of the third embodiment configured in a closed position for storage or transport. In this position, as in the position of FIG. 3A, the hinged portion **114** of display **310** is inserted into the rear connector **312B**. However, in this position the display module **310** is folded forward, against the top surface of chassis **300**, to protect the display side **310A**. The exposed back side **310B** of display module **310** is simply blank plastic.

Example of Internal Hardware

FIG. **4** shows a block diagram of the electronic organization of a sample pen- and keyboard-operated computer. (Of course, other organizations can be used instead; this organization is provided merely as one example of a context for use of the claimed inventions.)

In this example, a processor chipset **402** and **406**, similar to the Intel 386SL chipset, is used. (Of course, a very wide variety of other chipsets can be used instead.) Chip **402** includes the 386SX processor core, and also includes ISA bus control logic (connected to ISA bus **407**). Chip **402** also includes memory controller logic (connected to main memory **403** by 20 address lines, 16 data lines, and RAS and CAS strobe signals). Chip **402** also includes cache control tag RAMs (connected to cache RAM **403A**). Chip **402** also includes math coprocessor interface logic (connected to optional math coprocessor **409**). Chip **402** also includes CPU power management logic. Chip **402** receives a clock input (16 MHz in this example) from clock/timing generator **404**.

Chip **406** is connected to receive clock line SYSCLK from chip **402**. Chip **406** also includes timer logic (approximately equivalent to two 8254s), which is connected to receive a 14.31818 MHz clock signal from clock generator **404**. Chip **406** also includes interrupt controller logic (approximately equivalent to two 8259As). Chip **406** also includes DMA controller logic (approximately equivalent to two 8237s). Chip **406** also includes bus interface logic which is connected to the bus **407**. Chip **406** also includes power management logic, and a real-time clock (approximately equivalent to a 146818), which is connected to receive a 32.768 KHz signal from clock generator **404**. Chip **406** also includes serial port control logic (approximately equivalent to two 16450s), which is connected to receive a 1.8432 MHz signal from clock generator **404**.

Flash EPROM **401** contains code for BIOS and for the operating system (e.g. DOS). This chip, in the presently preferred embodiment, is a 256K×8 memory.[5]

A microcontroller **410** (an 80C51SL, in the presently preferred embodiment) monitors user inputs to the soft power switch **411**, and also receives keyboard inputs (through photodiode **414** and associated control logic **412**). Associated with this microcontroller are SRAM **412**A (32K×8 in this sample embodiment), EPROM **412**B (8K×8 in this sample embodiment), and resonant crystal **413** (16 MHz in this sample embodiment). This microcontroller is also connected to control speaker **415**.

Microcontroller **410** is also connected to a medium-speed general-purpose two-way wireless interface, implemented by logic **460**. This logic includes an array of several (e.g. ten) LEDs **462** for transmission, and an array of two infrared PIN diodes **464** for reception. This interface provides the capability for a highly flexible portable wireless interface, which can provide (in effect) the capability of a self-connecting and self-configuring LAN, with the proper software. These diodes are driven by an IR communications driver **466** (which, in the presently preferred embodiment, is a Photonics chip), and interface to bus **407** through control chip **468**.

Microcontroller **410** is also connected to LCD contrast control logic **444**. This logic, together with video signals from the VGA controller **440** (which is a Cirrus **6410** in the presently preferred embodiment), controls the LCD display **442**. VGA controller **440** is also connected to local DRAM **441**, and a programmable pixel clock **443**.

Also connected to bus **407** is an IDE interface **470**, which is connectible to the internal hard disk drive (or to a semiconductor mass-memory drive emulation, if a diskless configuration is chosen). Also connected to bus **407** is optional interface logic **472**, which can be connected to a modem or a LAN interface card.

Also connected to bus **407** is a digitizer chip set **420**, which is connected to detect and measure the movements of the computer's stylus.

Power supply **450** is driven by battery **452**, and also be powered by in input socket **454**. This power supply, in the presently preferred embodiment, provides output voltages of ±5V, ±12V, and ±18.6V. However, of course, lower logic supply voltages may be used in future embodiments.

An "I/O Slice" **430** plugs into the side of the computer's chassis **100** using a special 86-pin connector. This connector expansion unit provides industry-standard connectors for serial ports (DB-9), parallel ports (DB-25), SCSI interface (DB-25), and for optional docking to external display (DB-15VGA) and/or keyboard (DIN 5-pin) units.

Further Modifications and Variations

It will be recognized by those skilled in the art that the innovative concepts disclosed in the present application can be applied in a wide variety of contexts. Moreover, the preferred implementation can be modified in a tremendous variety of ways. Accordingly, it should be understood that the modifications and variations suggested below and above are merely illustrative. These examples may help to show some of the scope of the inventive concepts, but these examples do not nearly exhaust the full scope of variations in the disclosed novel concepts.

For example, the keyboard can be cabled to the chassis, or linked by an infrared transceiver, in place of the connector configurations shown. Similarly, the stylus may have a cabled link to the chassis, or may have a wireless data link, or may even be a purely passive stylus (in which case the position-sensing electronics are located in the display module).

As will be recognized by those skilled in the art, the innovative concepts described in the present application can be modified and varied over a tremendous range of applications, and accordingly the scope of patented subject matter is not limited by any of the specific exemplary teachings given.

What is claimed is:

1. A computer system, comprising:

a keyboard dimensioned for touch-typing;

a primary system chassis which contains a CPU, a power supply, a program memory, and at least one data communications interface;

said primary system chassis also being shaped to define a bay, in the exterior thereof, which is dimensioned to hold said keyboard removably but securely;

said primary system chassis having a top surface and first and second long edges in proximity to said top surface, and containing therein a first connector in proximity to said first long edge and a second connector in proximity to said second long edge, said CPU being operatively connected to send display signals to first connector and to said second connector;

a display module having therein a third connector which is complementary to said first connector and is also substantially complementary to said second connection, and also including display electronics which provide a visible display corresponding to signals written in through said connector, said display and said chassis alternatively positionable in a first alternative position in which said third connector is mated to said second connector, said display module is folded backward to exposed said visible display and said keyboard is mounted on said top surface and in a second alternative position in which said third connector is mated to said first connector and said keyboard is stored in said storage bay; and

stylus position-sensing electronics which are connected to sense the position of a user-operated stylus which may be in proximity to said visible display, and to provide electronic output to said CPU accordingly.

[5] See commonly-owned U.S. patent application Ser. No. 706,750, filed May 29, 1991, which is hereby incorporated by reference. This application discloses a computer system in which the basic system software can be electrically rewritten. This system provides some significant safeguards against data corruption.

2. The system of claim 1, wherein said stylus position-sensing electronics comprise a stylus which includes active electronic circuitry.

3. The system of claim 1, wherein said stylus position-sensing electronics comprise a stylus which includes active electronic circuitry and wireless transceiver circuitry.

4. The system of claim 1, wherein said chassis comprises a fourth connector, and said stylus position-sensing electronics comprise a stylus connected by a cable to said fourth connector.

5. The system of claim 1, wherein said display and said chassis are alternatively positionable, in a third alternative position, in which said display module is mounted against said top surface of said chassis and said keyboard is stored in said storage bay.

6. A computer system, comprising:

a keyboard having more than 77 keys and dimensioned for touch-typing;

a primary system chassis which contains a CPU, a power supply, a program memory, and at least one data communications interface;

said primary system chassis also being shaped to define a bay, in the exterior thereof, which is dimensioned to hold said keyboard removably but securely with said keys inward;

said primary system chassis having a top surface and first and second long edges in proximity to said top surface, and containing therein a first connector in proximity to said first long edge and a second connector in proximity to said second long edge, said CPU being operatively connected to read and write to first connector and to said second connector;

a display module having therein a third connector which is complementary to said first connector and is also substantially complementary to said second connector, and also including display electronics which provide a visible display corresponding to signals written in through said connector, said display and said chassis alternatively positionable in a first alternative position in which said third connector is mated to said second connector, said display module is folded backward to expose said visible display and said keyboard is mounted on said top surface and in a second alternative position in which said third connector is mated to said first connector and said keyboard is stored in said storage bay; and

stylus position-sensing electronics which are connected to sense the position of a user-operated stylus which may be in proximity to said visible display, and to provide electronic output to said CPU accordingly.

7. The system of claim 6, wherein said first connector provides insertion substantially normal to said top surface of said chassis, and said second connector also provides insertion substantially normal to said top surface of said chassis.

8. The system of claim 6, further comprising a stylus removably mounted to said chassis.

9. The system of claim 6, wherein said top surface of said chassis is shaped to include a recess therein, and said first connector is located inside said recess and provides insertion substantially parallel to said top surface of said chassis, and said second connector provides insertion substantially normal to said top surface of said chassis.

10. The system of claim 6, wherein said top surface of said chassis is shaped to include a lip near said first edge thereof, and said first connector is located inside said lip and provides insertion substantially parallel to said top surface of said chassis, and said second connector provides insertion substantially normal to said top surface of said chassis.

11. The system of claim 6, wherein said stylus position-sensing electronics comprise a stylus which includes active electronic circuitry.

12. The system of claim 6, wherein said stylus position-sensing electronics comprise a stylus which includes active electronic circuitry and wireless transceiver circuitry.

13. The system of claim 6, wherein said chassis comprises a fourth connector, and said stylus position-sensing electronics comprise a stylus connected by a cable to said fourth connector.

14. The system of claim 6, wherein said display and said chassis are alternatively positionable in a third alternative position in which said display module is mounted against said top surface of said chassis and said keyboard is stored in said storage bay.

15. A portable computer system chassis having front, back, top and bottom sides and a recess formed in said bottom side, comprising:

first and second I/O display module connectors located on said top side proximate said front and back sides, respectively, said I/O display module connectors adapted to alternatively communicate with a I/O display module depending upon a desired computer configuration; and

a first keyboard connector located on said top side proximate said front side, said keyboard connector adapted to communicate with a keyboard when said I/O display module communicates with said second I/O display module connector, said keyboard stowed within said recess when said I/O display module communicates with said first I/O display module connector to thereby allow said chassis to be alternatively configured to be part of a keyboard-entry computer or a display-entry computer.

16. The chassis as recited in claim 15 wherein said chassis can be alternatively configured to be part of a combined keyboard-entry/display-entry computer.

* * * * *

APPENDIX K

IEEE 100
The Authoritative Dictionary of IEEE Standards Terms

Seventh Edition

IEEE
Published by
Standards Information Network
IEEE Press

Trademarks and disclaimers

IEEE believes the information in this publication is accurate as of its publication date; such information is subject to change without notice. IEEE is not responsible for any inadvertent errors.

Other tradenames and trademarks in this document are those of their respective owners.

The Institute of Electrical and Electronics Engineering, Inc.
3 Park Avenue, New York, NY, 10016-5997, USA

To order IEEE Press publications, call 1-800-678-IEEE.

Print: ISBN 0-7381-2601-2 *SP1122*

See other standards and standards-related product listings at: http://standards.ieee.org/

The publisher believes that the information and guidance given in this work serve as an enhancement to users, all parties must rely upon their own skill and judgement when making use of it. The publisher does not assume any liability to anyone for any loss or damage caused by any error or omission in the work, whether such error or omission is the result of negligence or any other cause. Any and all such liability is disclaimed.

This work is published with the understanding that the IEEE is supplying information through this publication, not attempting to render engineering or other professional services. If such services are required, the assistance of an appropriate professional should be sought. The IEEE is not responsible for the statements and opinions advanced in this publication.

Library of Congress Cataloging-in-Publication Data

IEEE 100 : the authoritative dictionary of IEEE standards terms.—7th ed.
p. cm.
ISBN 0-7381-2601-2 (paperback : alk. paper)
1. Electric engineering—Dictionaries. 2. Electronics—Dictionaries. 3. Computer engineering—Dictionaries. 4. Electric engineering—Acronyms. 5. Electronics—Acronyms. 6. Computer engineering—Acronyms. I. Institute of Electrical and Electronics Engineers.

TK9 .I28 2000 00-050601
621.3′03—dc21

types of air gap surge arresters to exhibit a low resistance and then to revert to a high resistance state as a result of an external influence. (SPD/PE) C62.32-1981s, [8]

clearing circuit A circuit used for the operation of a signal in advance of an approaching train. (EEC/PE) [119]

clearing-out drop (cord circuit or trunk circuit) A drop signal that is operated by ringing current to attract the attention of the operator. (EEC/PE) [119]

clearing source (low-voltage air-gap surge-protective devices) (low voltage surge protective devices) A defined electrical source which is intentionally applied as a clearing stimulus to an air gap surge protective device under laboratory test conditions. This stimulus is intended to simulate conditions encountered during normal usage. (SPD/PE) C62.32-1981s, [8]

clearing time (fuse) The time elapsing from the beginning of an overcurrent to the final circuit interruption. *Note:* The clearing time is equal to the sum of melting time and arcing time. *Synonym:* total clearing time. (SWG/PE) [56]
(2) **(A)** (mechanical switching device). The interval between the time the actuating quantity in the main circuit reaches the value causing actuation of the release and the instant of final arc extinction on all poles of the primary arcing contacts. *Note:* Clearing time is numerically equal to the sum of contact parting time and arcing time. **(B)** (total clearing time of a fuse). The time elapsing from the beginning of a specified overcurrent to the final circuit interruption, at rated maximum voltage. *Note:* The clearing time is equal to the sum of melting time and the arcing time.
(SWG/PE/NP) C37.100-1992, 308-1980, C37.40-1993

clearly discernable Capable of being noticed easily and without close inspection. (SUB/PE) C37.123-1996

clear packet A packet used during initialization to empty line buffers and initialize the line. CSR state is unaffected; e.g., the node's address is unchanged by a "clear." Clear may be sent by any node that has lost synchronization in order to trigger reinitialization. (C/MM) 1596-1992

clear sky (illuminating engineering) A sky that has less than 30% cloud cover. (EEC/IE) [126]

cleartext Intelligible data, the semantic content of which is available. (LM/C) 802.10-1992

cleat An assembly of two pieces of insulating material provided with grooves for holding one or more conductors at a definite spacing from the surface wired over and from each other, and with screw holes for fastening in position. *See also:* raceway. (EEC/PE) [119]

clerestory (illuminating engineering) That part of a building which rises clear of the roofs or other parts and whose walls contain windows for lighting the interior. (EEC/IE) [126]

CLI *See:* cumulative leakage index.

click (1) A disturbance of a duration less than a specified value as measured under specified conditions. *See also:* electromagnetic compatibility. (EMC/IM) [53], C63.4-1991, [76]
(2) The act of pressing and releasing a mouse button without moving the mouse pointer. (C) 1295-1993w

client (1) (MULTIBUS) An agent that requests services of a server. *See also:* server. (C/MM) 1296-1987s
(2) Software that uses the interface.
(C/PA) 1351-1994w, 1224-1993w, 1327-1993w, 1328-1993w
(3) In networking, a station or program requesting a service. *Contrast:* server. (C) 610.7-1995
(4) Software that uses an interface. (C/PA) 1224.1-1993w
(5) Refers to the software component on one device that uses the services provided by a server on another device.
(C/MM) 1284.4-2000
(6) *See also:* batch client.
®MULTIBUS is a registered trademark of Intel Corporation.

client application An application program that makes use of Media Management System (MMS) services to manage its media. Examples of client applications include a backup program, a hierarchical storage manager, and an application that allows individual users to mount their own tapes. (C/SS) 1244.1-2000

Client Cached Block Cookie A particular Block Cookie associated with a Base Client. (IM/ST) 1451.1-1999

client execution environment The machine state that exists when a client program begins execution. (C/BA) 1275-1994

client interface A set of data and procedures giving a client program access to client interface services. (C/BA) 1275-1994

client instance A manifestation of the client that shares the input and output queues of the client with other instances. (C/PA) 1224.1-1993w

client interface handler A mechanism by which control and data are transferred from a client program to the firmware, and subsequently returned, for the purpose of providing client interface services. (C/BA) 1275-1994

client layer In the OSI model, refers to the data link and physical layers. *See also:* transport layer; network layer; presentation layer; physical layer; sublayer; data link layer; entity layer; application layer; session layer; logical link control sublayer; medium access control sublayer. (C) 610.7-1995

client interface services Those services that Open Firmware provides to client programs, including device tree access, memory allocation, mapping, console I/O, mass storage, and network I/O. (C/BA) 1275-1994

Client object Any object that invokes operations on other objects. (JM/ST) 1451.1-1999

Client Port An instance of the class `IEEE1451_ClientPort` or of a subclass thereof. (IM/ST) 1451.1-1999

client program A software program that is loaded and executed by Open Firmware (or a secondary boot program). (The client program may use services provided by the Open Firmware client interface.). (C/BA) 1275-1994

client role The location where the software is actually executed or used (as opposed to the target where it is actually installed). The configuration of software is performed by this role. (C/PA) 1387.2-1995

client-server In a communications network, the client is the requesting device and the server is the supplying device. For example, the user interface could reside in the client workstation while the storage and retrieval functions could reside in the server database. (C) 610.7-1995

client-server communication A communication pattern, where a specific object, the client, communicates in a one-to-one fashion with a specific server object, the server. (IM/ST) 1451.1-1999

climber in training A worker who is in training to become a qualified climber. (T&D/PE) 1307-1996

climbing The vertical movement (ascending and descending) and horizontal movement to access or depart the worksite. (NESC/T&D/PE) C2-1997, 1307-1996

climbing space The vertical space reserved along the side of a pole or structure to permit ready access for linemen to equipment and conductors located on the pole structure. (T&D/PE) 196-1951w, [10], C2.2-1960

clinometer (navigation aids) An instrument for indicating the degree of slope of the angle of roll or pitch of a vehicle, according to the plane in which it is mounted. (AES/GCS) 172-1983w

clip (1) (charged-particle detectors) (x-ray energy spectrometers) (radiation detectors) A limiting operation, such as the use of a high-pass filter or a nonlinear operation such as diode limiting of pulse amplitude. *Synonym:* clipping. *See also:* fuse clips; differentiated; contact clip. (NPS/NID) 325-1971w, 759-1984r, 301-1976s
(2) (charged-particle detectors) A limiting operation, such as the use of a high-pass filter (differentiator) or a nonlinear operation to limit the amplitude of a pulse. The first usage is archaic. *Synonym:* clipping. (NPS) 300-1988r
(3) *See also:* cable clamp. (PE/T&D) 524-1992r

clipboard
object th
object th
clipper (d
the insta
imum v
which t
side of
clipper li
input lie
inputs a
tude gat
clipping (
tial or fi
ation of
(2) (con
which d
made in
view ar
area be
two-din
dow; in
ume.
scissori

Bef

(3) *Se*
(4) *Se*

clipping-
of sag
suspen
pensio

clipping
distanc
to a po
sion cl
clippir
forces
clips *See*
CLIST
ment.
CLK A
signal.
clock (1)
synchi
time.
clock;
chang

(2) A
high l
stored
an op
(3) A
value
possil

(4) (
signal
opera

APPENDIX L

WEBSTER'S NEW WORLD™

COMPUTER DICTIONARY

NINTH EDITION

by Bryan Pfaffenberger, Ph.D.

Webster's New World™ Computer Dictionary, Ninth Edition

Hungry Minds, Inc.
909 Third Ave.
New York, NY 10022
www.hungryminds.com

For general information on Hungry Minds' products and services please contact our Customer Care department; within the U.S. at 800-762-2974, outside the U.S. at 317-572-3993 or fax 317-572-4002. For sales inquiries and reseller information, including discounts, bulk sales, customized editions, and premium sales, please contact our Customer Care department at 800-434-3422.

A Webster's New World™ Book

Library of Congress Control Number: 2001091950

ISBN 0-7645-6325-4

Cataloging-in-publication information available upon request.

Manufactured in the United States of America

5 4 3 2

temporarily solved by means of the CIDR addressing protocol on Internet backbone networks, and it will be permanently solved by IPv6, the next-generation IP protocol, which will introduce a 128-bit address space. See *CIDR, Class A network, Class C network, IP address, IPv6.*

Class C network On the Internet, a participating network that is allocated up to 256 distinct Internet addresses (called IP addresses). Current Internet addressing limitations define a maximum of 2,097,152 Class C networks. See *Class A network, Class B network, IP address.*

clean management In a Y2K readiness program, a management program in which any new system components (including hardware peripherals, programs, or network components) are tested forY2K compliance before being added to a Y2K-compliant system. See *Y2K,Y2K-compliant.*

clear To remove data from a document, cell, or field. In the Microsoft Windows 95/98 and Macintosh environments, the Clear command (Edit menu) completely wipes out the selection, as opposed to Cut, which removes the selection to the Clipboard (from which one can retrieve the selection, if he or she later discovers that he or she deleted it by mistake).

cleartext In cryptography, a message that is transmitted without any encryption so that it can be easily intercepted and read while it is en route. A major security drawback of the Internet is that, with most password authentication schemes, passwords are transmitted in cleartext. See *ciphertext.*

Clear to Send/Ready to Send See *CTS/RTS.*

CLEC Acronym for competitive local exchange carrier. In U.S. telephony, a local exchange carrier (LEC) that is now permitted (thanks to the U.S. 1996 Telecommunications Act) to compete in local telephone markets with the incumbent local exchange carrier (ILEC), the company that possessed a monopoly in that market prior to the passage of the 1996 reforms. See *ILEC, LEC.*

click To press and quickly release a mouse button. When no button is specified, the left button is assumed. One frequently sees this term in instructions such as "Click the Bold check box in the Fonts dialog box." For users of IBM-compatible PCs, this instruction means, "Move the mouse pointer so that its tip touches the Bold check box, and then click the left mouse button." See *double-click, Shift+click.*

click and mortar In electronic commerce (e-commerce), a retail strategy in which a Web retail site is paired with a chain of local retail stores. Customers prefer this strategy because they can return or exchange unwanted goods more easily. See *e-commerce.*

client **1.** In an Internet service, a program that can communicate with a server located on the Internet to exchange data of a certain type, such as a Web document or an e-mail message. A Web browser is a client for accessing information available on Web servers. **2.** In a client/server network, a program that is designed to request information from a server. See *client/server, heavy client, light client.* **3.** In Object Linking and Embedding (OLE), an application that includes data in another application, called the server application. See *client application.*

client application In Object Linking and Embedding (OLE), an application in which one can create a linked object or embed an object. See *server application.*

client/server A design model for applications running on a network, in which the bulk of the back-end processing, such as performing a physical search of a database,