

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

FORD MOTOR COMPANY
Petitioner,

v.

VERSATA SOFTWARE, INC.
Patent Owner.

U.S. Patent No. 7,882,057

IPR Case No.: 2016-01013

**DECLARATION OF DR. PHILIP GREENSPUN IN SUPPORT
OF *INTER PARTES* REVIEW UNDER 35 U.S.C. § 311 *ET SEQ.*
AND 37 C.F.R. § 42.100 *ET SEQ.* (CLAIMS 17, 30, AND 44-46 OF
U.S. PATENT NO. 7,882,057)**

TABLE OF CONTENTS

List of Exhibits.....3

I. Qualifications and Professional Experience.....7

II. Relevant Legal Standards13

III. Level of Ordinary Skill in the Art15

IV. The '057 Patent.....16

V. The '057 Patent Prosecution History.....26

VI. Challenged Claims of the '057 Patent Viewed in their Broadest Reasonable Interpretation27

VII. Scope and Content of the Prior Art (Summary)27

VIII. Prior Art: Loomans, U.S. Patent 7,873,503.....42

IX. Prior Art: “A Customization Approach for Structure Products in Electronic Shops” (“Stahl”).....44

X. Grounds for Challenge.....47

 A. Ground 1 – Claims 17, 30, 44 And 45-46 Obvious Based On Loomans In View Of Stahl And The General Knowledge Of A Person Having Ordinary Skill In The Art47

 1. Analysis of Claims 17, 30, and 44.....47

 a. Claim 17.....48

 b. Obvious to Combine Loomans with Stahl.....83

 c. Claim 30.....87

 d. Claim 44.....102

 2. Analysis of Claims 45 and 46.....108

 a. Claim 45.....108

 b. Claim 46.....150

XI. Conclusion.....153

List of Exhibits

Exhibit No.	Description	Date	Identifier
1101	U.S. Patent No. 7,882,057	Feb. 1, 2011	'057 Patent
1102	Expert Declaration of Dr. Philip Greenspun	n/a	Greenspun Decl.
1103	Curriculum Vitae of Dr. Philip Greenspun	n/a	Greenspun CV
1104	U.S. Patent No. 7,882,057 File History	n/a	'057 Patent File History
1105	U.S. Patent No. 7,873,503 to Loomans et al.	Jan. 18, 2011	Loomans
1106	A. Stahl, R. Bergmann, S. Schmitt, <u>A Customization Approach for Structured Products in Electronic Shops</u> , <i>Electronic Commerce: The End of the Beginning, 13th International Bled Electronic Commerce Conference</i> (June 19-21, 2000)	Jun. 2000	Stahl
1107	Alexander Kott, Gerald Agin, David Fawcett, <u>Configuration Tree Solver: A Technology for Automated Design and Configuration</u> , <i>ASME Journal of Mechanical Design</i> 114(1): 187-195 (1992)	1992	Kott
1108	L. Anselma, D. Magro, and P. Torasso, <u>Automatically Decomposing Configuration Problems</u> , <i>AI*IA 2003: Advances in Artificial Intelligence</i> , Lecture Notes in Computer Science, Volume 2829, pp. 39-52 (2003)	2003	Anselma

Exhibit No.	Description	Date	Identifier
1109	D. Magro and P. Torasso, <u>Decomposing and Distributing Configuration Problems</u> , <i>Artificial Intelligence: Methodology, Systems, and Applications</i> , Lecture Notes in Computer Science, Volume 2443, pp. 81-90 (2002)	2002	Magro
1110	Judith Bachant, John McDermott, <u>R1 Revisited: Four Years in the Trenches</u> , <i>AI Magazine</i> Volume 5, Number 3 (1984)	1984	Bachant
1111	John McDermott, <u>R1: A Rule-Based Configurer of Computer Systems</u> , <i>Artificial Intelligence</i> (1982)	1982	McDermott
1112	Bryan M. Kramer, <u>Knowledge-Based Configuration of Computer Systems Using Hierarchical Partial Choice</u> , <i>IEEE</i> (1991)	1991	Kramer
1113	Bei Yu and Hans Jorgen Skovgaard, <u>A Configuration Tool to Increase Product Competitiveness</u> , <i>IEEE Intelligent Systems</i> , 34-41 (July/August 1998)	1998	Yu
1114	U.S. Patent Application Publication No. 2003/0187950 to Rising	Oct. 2, 2003	Rising
1115	Martin R. Wagner, <u>Understanding the ICAD System</u> , ICAD, Inc., 1990	1990	ICAD
1116	Oracle Configurator Developer, User's Guide, Release 11i for Windows 95/98/2000 and Windows NT 4.0	April 2002	Oracle
1117	Stefan Schulz, <u>CBR-Works A State-of-the-Art Shell for Case-Based Application Building</u> , TECINNO GmbH, 1999	1999	CBR

Exhibit No.	Description	Date	Identifier
1118	Richard M. Stallman and Gerald Jay Sussman, <u>Forward Reasoning and Dependency-Directed Backtracking In a System for Computer-Aided Circuit Analysis</u> , MIT Artificial Intelligence Laboratory, Memo No. 380, Sept. 1976	Sept. 1976	Stallman

I, Philip Greenspun, hereby declare as follows:

1. I am making this declaration at the request of Ford Motor Company in the matter of *Inter Partes* Review of U.S. Patent No. 7,882,057 (“the ’057 Patent”) to Little.

2. I am a salaried non-owner employee of Fifth Chance Media LLC, which is being compensated for my work in this matter at a rate of \$475/hour. My compensation in no way depends on the outcome of this proceeding.

3. In preparation of this declaration, I have studied the exhibits as listed in the Exhibit List shown above. Each of these exhibits is a true and accurate copy.

4. In forming the opinions expressed below, I have considered:

(a) The documents listed above as well as additional patents and documents referenced herein;

(b) The relevant legal standards, including the standard for obviousness provided in *KSR International Co. v. Teleflex, Inc.*, 550 U.S. 398 (2007), and any additional legal standards set forth in the body of this declaration; and

(c) My knowledge and experience based upon my work and study in this area as described below.

I. Qualifications and Professional Experience

5. I have provided my full background in the curriculum vitae that is attached as Exhibit 1103.

6. I earned a Ph.D. in Electrical Engineering and Computer Science from Massachusetts Institute of Technology in 1999. I also obtained a Bachelor of Science Degree in Mathematics from Massachusetts Institute of Technology in 1982 and a Master of Science Degree in Electrical Engineering and Computer Science from Massachusetts Institute of Technology in 1993.

7. My Ph.D. dissertation concerned the engineering of large online Internet communities with a Web browser front-end and a relational database management system (RDBMS) containing site content and user data.

8. I have authored five computer science textbooks in total, including Database Backed Websites (Macmillan), Software Engineering for Internet Applications, and an SQL language tutorial.

9. I have served as an independent member of various advisory and corporate boards, mostly for technology companies. For example, I joined the corporate board of an MIT materials science spin-off in late 2005 during a \$550,000 seed capital phase. I stepped down when the company secured \$10 million in venture capital in mid-2007.

10. I began working full-time as a computer programmer in 1978,

developing a database management system for the Pioneer Venus Orbiter at the National Aeronautics and Space Administration's Goddard Space Flight Center.

11. In the early 1980s, I developed computer-aided design software for electronic systems, specifically to assist digital hardware engineers designing processors at Hewlett-Packard and Symbolics. The integrated circuit design software that I built at Symbolics included the capability to automatically configure various kinds of circuits.

12. I co-developed a computer program for computer-aided design of mechanical systems in the mid-1980s. This was called the ICAD System. The ICAD System enabled engineers to decompose a mechanical design into a hierarchy of subassemblies and establish configuration rules at each level of subassembly. The end-result was a system in which it was possible to go from customer specifications to a finished design without human intervention. The first applications for the ICAD System involved large structures built from steel, such as house-sized air-cooled heat-exchangers used in commercial buildings and industrial plants.

13. ICAD went public as "Concentra" in the 1990s and was acquired by Oracle Corporation in 2002. The product's mechanical design capabilities were deemphasized and its configuration capabilities were improved for use as a general-purpose sales configuration system. The product survives today as Oracle

Configurator, part of the Oracle Applications suite of business software. “Understanding the ICAD System” is a 1990 marketing brochure that contains an explanation of some of the basic capabilities. Excerpts from this brochure are reproduced below:

A design instance of an ICAD product model is organized into a tree structure called the *product structure tree*.

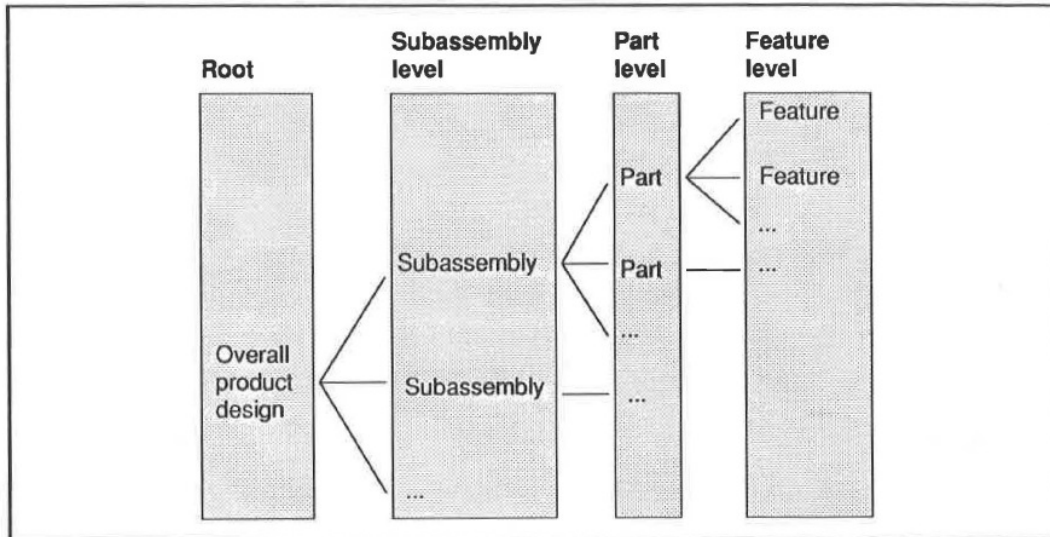
A tree organization naturally reflects the way that design and manufacturing engineers think about product design. It provides an unambiguous way of addressing every rule in the ICAD product model, which allows complex relationships to be set up between rules (see the section "Rule Dependencies in the ICAD Product Model", page 4-45). Tree organizations also provide a natural modularity to the product design. Complex assemblies are divided into subassemblies. Each subassembly can be designed separately, and each is further divided into components that can be designed separately.

A product structure tree links the components of a product design together. The *root* of the product structure tree is a design instance of the defpart that represents the entire product structure. The root branches out into more detailed levels which are called *children* or *parts* of the root. For example, a branch might represent a subassembly of the product assembly. Each child can have additional children. The terminal components of the tree are called *leaves*; these are parts that have no children.

It can be helpful to think of the different levels of the product structure tree as roughly representing a number of conceptual levels in a mechanical assembly. The root represents the entire design instance, which is divided into the design of the part. This overall design decomposes into subassemblies. Subassemblies are typically built from component parts (e.g., screws, holes, etc). Parts are often divided into their component design features. The lowest level (leaves) of the product structure tree usually represents the actual geometric structure of the mechanical part (see the section "Modeling with Simple Geometric Primitives", page 4-56).

Creating the Product Structure

This is shown in the following diagram. Note that product structure tree of your ICAD product model may not follow this diagram exactly. Sometimes there are many levels of subassemblies before the component part level, and in some cases component parts are not divided into design features.



Different levels in a product structure tree.

(Ex. 1115 [ICAD] at 4-29 – 4-31, pages 80-82.)

14. In the second half of the 1980s, I was the principal developer of a computer program for computer-aided design and control of civil engineering projects, specifically earthmoving. This work was the foundation of my master's thesis at MIT and also of U.S. Patent Nos. 5,150,310 and 5,964,298, on which I am a named inventor.

15. I developed my first program using a relational database management system in 1994. It was a Web interface to the Children's Hospital Oracle RDBMS, Version 6. This application enabled doctors at the hospital to view patient clinical

data using any computer equipped with a Web browser.

16. In 1995, I led an effort by Hearst Corporation to set up an infrastructure for Internet applications across all of their newspaper, magazine, radio, and television properties. This infrastructure included software for managing users, shopping carts, electronic commerce, advertising, and user tracking. One capability of the system was using private data, regarding a user's history, to determine the (publicly available) advertisements to be shown on pages viewed by that user, including pages that included order summaries and other private data.

17. Between 1995 and 1997, I significantly expanded the photo.net online community that I had started in 1993 to help people teach each other to become better photographers. I began distributing the source code behind photo.net to other programmers as a free open-source toolkit, called "ArsDigita Community System." One version of this system was an add-on to AOLserver, a Web server with an API.

18. In May 1997, Macmillan published my first textbook on Internet Application development, Database Backed Websites. A September 1998 update to this book was published as Philip and Alex's Guide to Web Publishing (hardcopy version published in April 1999).

19. In 1997, I started a company, ArsDigita, to provide support and

service for the ArsDigita Community System. Between 1997 and the middle of 2000, I managed the growth of ArsDigita to 80 people, almost all programmers, and \$20 million per year in annual revenue. This involved supervising dozens of software development projects, nearly all of which were Internet Applications with a Web front-end and an Oracle RDBMS back-end.

20. In 1999, I supervised the packaging up of much of our ecommerce-related code into the “ecommerce” module of the ArsDigita Community System. As the founder, CEO, and chief technical employee of the company, I personally developed functional specifications, SQL data models (Structured Query Language, or “SQL”, the standard programming language for relational database management systems), and Web page flows that determined the user experience.

21. Between 2000 and the present, I have managed software development projects for philip.greenspun.com and photo.net. Both online services are implemented as relational database management applications. For photo.net, in particular, I evaluated various Web-based comparative shopping tools that would allow readers to find the best delivered prices, given a zip code, for camera equipment. In addition, I am currently developing a Facebook application that allows parents to create electronic baby books.

22. Separately from this commercial and public work, I have been involved as a part-time teacher within the MIT Department of Electrical

Engineering and Computer Science, educating students in how to develop Internet Applications with an RDBMS back-end. In the spring of 1999, I taught 6.916, Software Engineering of Innovative Web Services, with Professors Hal Abelson and Michael Dertouzos. In the spring of 2002, this course was adopted into the standard MIT curriculum as 6.171. I wrote 15 chapters of a new textbook for this class, Software Engineering for Internet Applications. This book was published on the Web at <http://philip.greenspun.com/seia/> starting in 2002 and 2003 and also in hardcopy from MIT Press in 2006. I am the sole author of a supplementary textbook for the class, SQL for Web Nerds, a succinct SQL programming language tutorial available only on the Web at <http://philip.greenspun.com/sql/>. I use this book when I teach an intensive course in database programming at MIT, as I did most recently in January 2015.

23. Based at least on my education and experience, I consider myself to be an expert in software engineering, including the development of configuration systems such as the system described in the '057 Patent.

II. Relevant Legal Standards

24. I have been asked to provide opinions regarding the validity of claims of the '057 Patent in light of the prior art.

25. It is my understanding that a claimed invention is unpatentable under 35 USC § 102 if a prior art reference teaches every element of the claim. This is

sometimes referred to as “anticipation.”

26. It is my understanding that a claimed invention is unpatentable under 35 U.S.C. § 103 if the differences between the invention and the prior art are such that the subject matter as a whole would have been obvious at the time the alleged invention was made to a person having ordinary skill in the art to which the subject matter pertains. This is sometimes described as “obviousness.” I understand that an obviousness analysis takes into account the level of ordinary skill in the art, the scope and content of the prior art, and the differences between the prior art and the claimed subject matter.

27. It is my understanding that the Supreme Court, in *KSR Int’l Co. v. Teleflex Inc.*, 550 U.S. 398 (2007) and other cases, has recognized several rationales for combining references or modifying a reference to show obviousness of the claimed subject matter. Some of these rationales include the following: combining prior art elements according to known methods to yield predictable results; simple substitution of one known element for another to obtain predictable results; a predictable use of prior art elements according to their established functions; applying a known technique to a known device to yield predictable results; choosing from a finite number of identified, predictable solutions, with a reasonable expectation of success; and some teaching, suggestion, or motivation in the prior art that would have led one of ordinary skill to modify the prior art

reference or to combine prior art reference teachings to arrive at the claimed invention.

III. Level of Ordinary Skill in the Art

28. I have reviewed the '057 Patent, as well as the pertinent prior art documents discussed below. Based on this review and my knowledge of the configuration system field, including my work on ICAD system in the 1980s, it is my opinion that a person having ordinary skill in the art would have the following: (1) a bachelor's degree in computer science, electrical engineering, computer engineering, or similar technical field, or (2) equivalent experience in the design or implementation of configuration systems. The relevant field of art is product configuration software.

29. I understand that this determination is made at the time of the invention, which I understand that the patentee states as being the October 4, 2004 filing of U.S. Application No. 10/957,919, which ultimately issued as the '057 Patent.

30. As I also discussed in my "Qualifications and Professional Experience" section above, I am familiar with the level of knowledge and the abilities of a person having ordinary skill in the art at the time of the claimed invention based on my education and work experience.

IV. The '057 Patent

31. The '057 Patent discloses a configuration system and method for “processing complex configuration problems using configuration sub-models.” (Ex. 1101 ['057 Patent] at 1:8-10.)

32. In the Background of the Invention, the '057 Patent discloses a conventional product configuration process known in the prior art:

In one embodiment of a conventional inference procedure, configuration query 102 is formulated based on user configuration input, a configuration engine performs the configuration query 102 using a configuration model 104, and the configuration engine provides an answer 106 to the configuration query 102 based on the configuration query 102 and the contents of the configuration model 104. The answer 106 represents a particular response to the configuration query 102.

(Ex. 1101 ['057 Patent] at 1:16-25.)

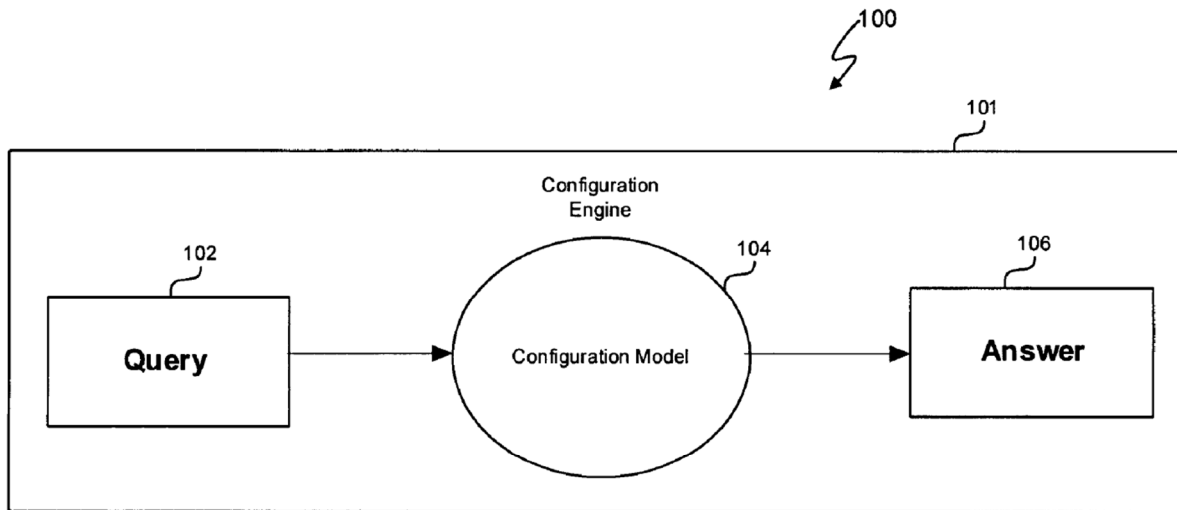


Figure 1 (prior art)

(Ex. 1101 ['057 Patent] at Figure 1.)

33. The '057 Patent further discloses the complex relationships and queries that make-up a configuration model of a product:

A configuration model 104 uses, for example, data, rules, and/or constraints (collectively referred to as "data") to define compatibility relationships between parts (also commonly referred to as "features") contained in a specific type of product. A part represents a single component or attribute from a larger, more complex system. Parts may be combined in different ways in accordance with rules and/or constraints to define different instances of the more complex system. For example, "V6 engine" or the exterior color "red" can be parts on a vehicle, and a specific hard disk drive can be a part on a computer. A part group, also called a group, represents a collection of related parts. For example, an "Engines" group might contain the parts "V6 engine" and "4 cylinder engine". A

product configuration is a set of parts that define a product. For example, a vehicle configuration containing the parts "V6 engine" and "red" represents a physical vehicle that has a red exterior and a V6 engine. A product can be a physical product such as a vehicle, computer, or any other product that consists of a number of configurable features such as an insurance product. Additionally, a product can also represent a service. **A configuration query (also referred to as a "query") is essentially a question that is asked about the parts and relationships in a configuration model. The answer returned from a configuration query will depend on the data in the configuration model, the approach used for answering the question, and the specifics of the question itself.** For example, one possible configuration query, translated to an English sentence, is the following: For the given configuration model, are the parts "red" and "V6 engine" compatible with each other.

(Ex. 1101 ['057 Patent] at 1:26-54, emphasis added.)

34. The '057 Patent next acknowledges that the achievable complexity of configuration models has been limited because of computer processing limitations:

Solving configuration problems using computer assisted technology often requires a significant amount of data processing capabilities. Consequently, configuration technologies have attempted to exploit increased data processing capabilities, memory capacities, and network data transfer throughput rates by increasing the capabilities of the configuration engines and/or enhancing the complexity of configuration models and configuration queries. The

complexity of a configuration model can be defined in any number of ways, such as by the diversity of parts, part groups, rules, and constraints supported by the configuration model, by the number of parts, rules, and constraints, and by the complexity of part and part group relationships defined by configuration rules and constraints. **In any event, the practical complexity achievable for configuration models has been limited by the ability of computer systems to process data within a given period of time, T, and/or limited by other processing constraints, such as a lack of memory. The time period, T, represents an amount of time considered reasonable to perform a configuration task.** Time T can vary depending upon the application and expectation of configuration system users.

(Ex. 1101 [‘057 Patent] at 2:37-57, emphasis added.)

35. Figure 3 of the ‘057 Patent illustrates limitations on configuration models/configuration queries because of limited data processing capabilities known in the prior art. As complexity goes up, shown from left-to-right on the x-axis of the graphic in Figure 3 (and specifically depicted in line 302), the maximum data processing capability is reached (depicted by dashed line 304). Thus, the graphic in Figure 3 indicates that there is sufficient data processing capability to process the configuration model represented as “A” (below dashed line 304), but insufficient data processing capability to process the configuration model represented as “B” (above dashed line 304).

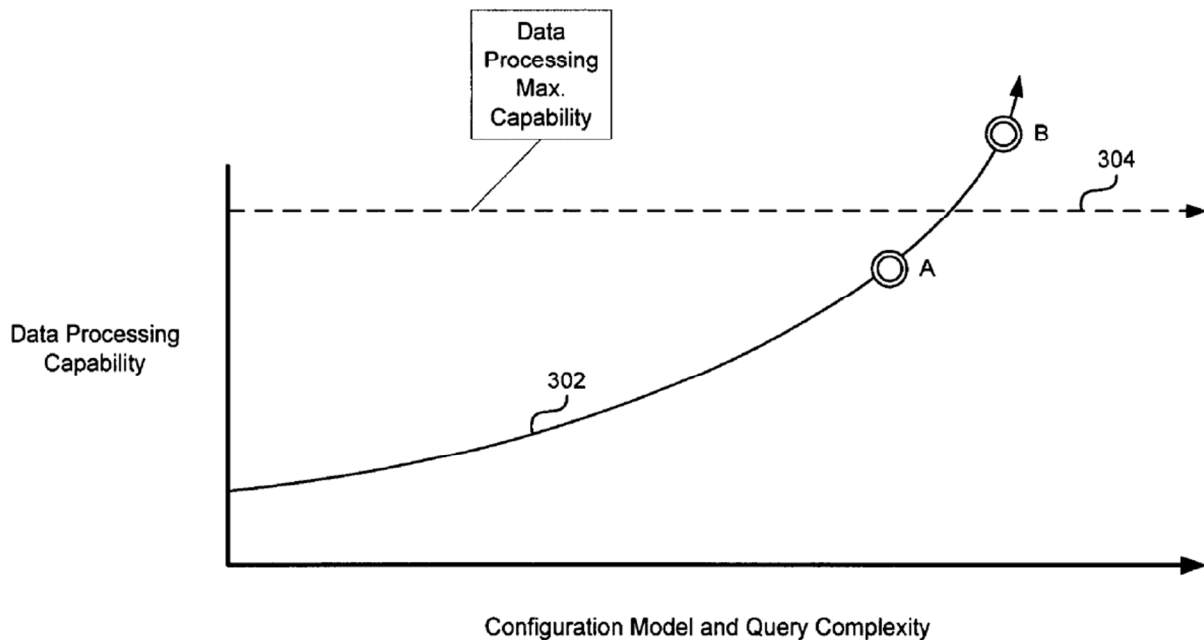


Figure 3 (prior art)

(Ex. 1101 [‘057 Patent] at Fig. 3.)

36. To overcome the limitations on data processing capabilities known in the prior art, the ‘057 Patent discloses:

A configuration model dividing and configuration sub-model inference processing system and procedure addresses the issue of configuration model and query complexity by breaking a configuration problem down into a set of smaller problems, solving them individually and recombining the results into a single result that is equivalent to a conventional inference procedure. In one embodiment, a configuration model is divided into configuration sub-models that can respectively be processed using existing data processing resources. The sub-model inference procedure does not change the exponential nature of configuration

model and query complexity but instead generates configuration sub-models on the side of the achievable performance curve. Accordingly, a sub-model inference procedure provides a way to scale queries to larger and more complicated configuration models. Embodiments of the configuration model dividing and configuration sub-model processing system and inference procedure allows processing by a data processing system of configuration models and queries whose collective complexity exceeds the complexity of otherwise unprocessable conventional, consolidated configuration models and queries.

(Ex. 1101 ['057 Patent] at 4:18-40, emphasis added.)

37. The '057 Patent discusses an embodiment where a consolidated configuration model is divided into several sub-models.

FIG. 4 depicts the configuration model dividing and configuration sub-model inference processing system 400 (referred to herein as "sub-model processing system 400") that performs configuration model dividing and configuration sub-model inference procedure 402 (referred to herein as "sub-model inference procedure 402"). The sub-model inference procedure 402 includes operations 404, 406, 408, and 410. The sub-model processing system 400 can include software code that is executable by a processor of a computer system, such as a server computer system. In a network environment, the sub-model processing system 400 can be accessed by and communicates with any number client systems 401(1) through 401(n).

Operation 404 receives, as an input, a conventional, consolidated configuration model 412 and divides the consolidated configuration model 412 into a set of configuration sub-models CM1 through CMn, where n is an integer representing the number of configuration sub-models. The configuration sub-models are an input to this process. In one embodiment, the configuration sub-models meet the following criteria: a. Each configuration sub-model should represent a portion of the source configuration model 412; b. The data collectively contained in the configuration sub-models should be sufficient to provide an answer for each of the sub-queries Q1 through Qn or query being processed; and c. The configuration sub-models should be divided in such a way that the results of the sub-queries or query can be recombined to provide an answer to the input configuration query 414.

(Ex. 1101 ['057 Patent] at 4:40-5:4.)

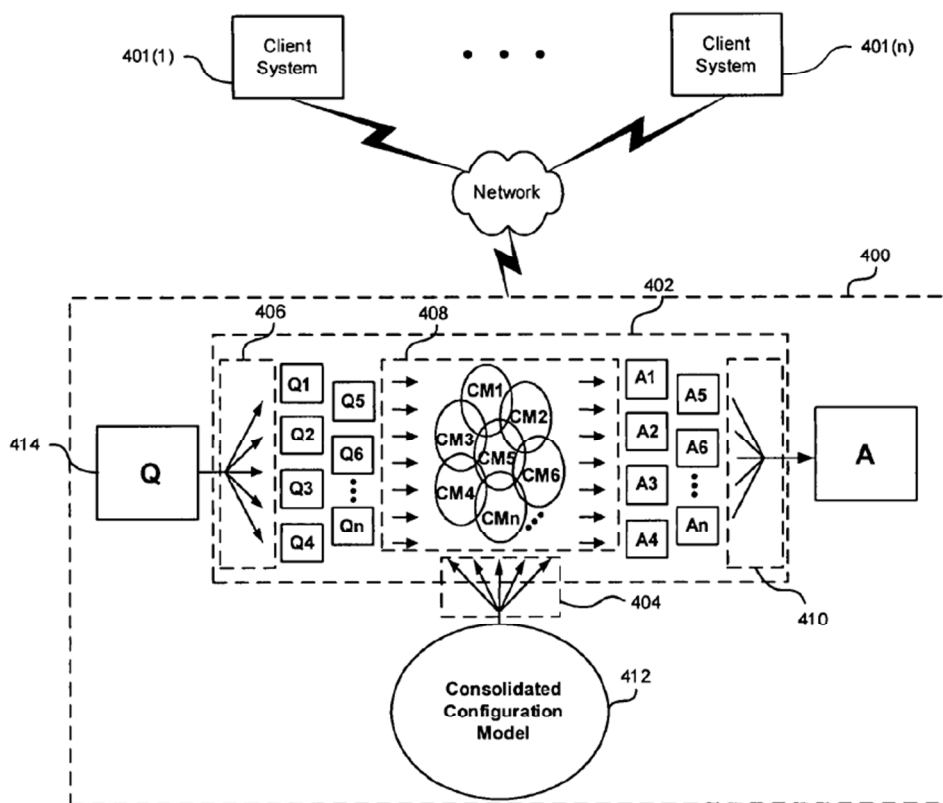


Figure 4

(Ex. 1101 ['057 Patent] at Fig. 4.)

38. Figure 6 of the '057 Patent illustrates an example of a consolidated configuration model (602), which has been divided into configuration sub-models CM_1 , CM_2 and CM_3 .

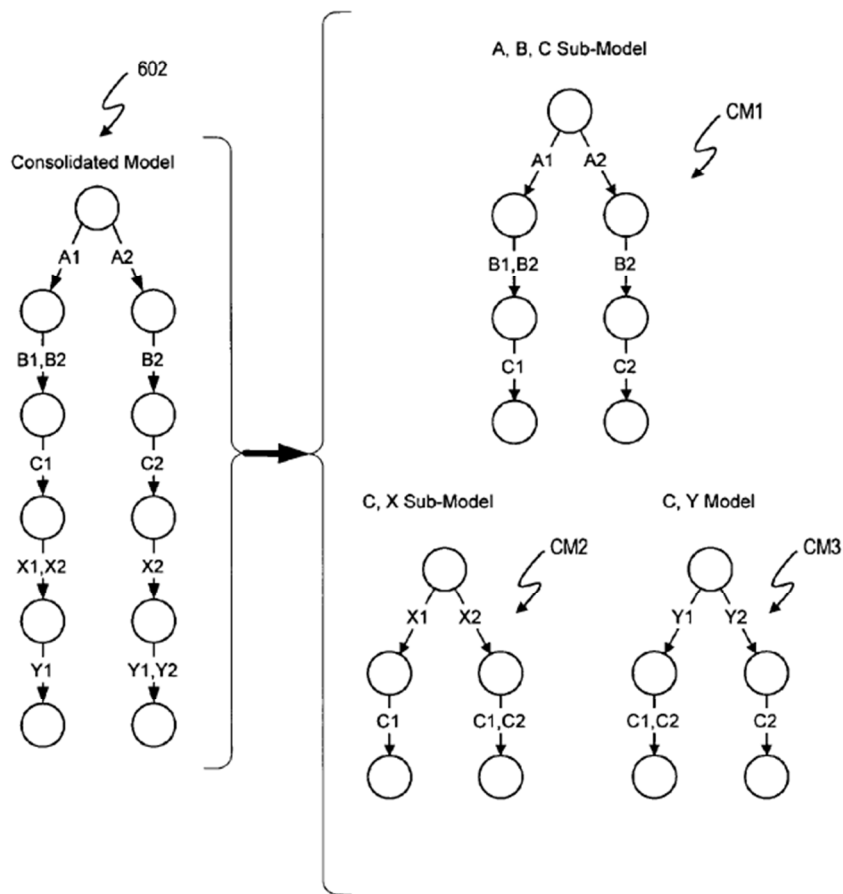


Figure 6

(Ex. 1101 [‘057 Patent] at Fig. 6.)

39. Figure 5 of the ‘057 Patent illustrates the data processing capability of a computer system when processing a consolidated configuration model (412) compared to the data processing capability of a computer system when processing sub-models CM_1 , CM_2 and CM_n are divided out of the consolidated configuration model (412). “In general, the consolidated configuration model 412 is divided sufficiently so that the complexity of each configuration sub-model CM_1 , CM_2 , through CM_n is low enough to allow processing using available data processing

capabilities while still representing the relationships included in the consolidated configuration model 412, which, in this embodiment, would otherwise not be cable (*sic*) of being processed by the computer system.” (Ex. 1101 [‘057 Patent] at 5:11-18.)

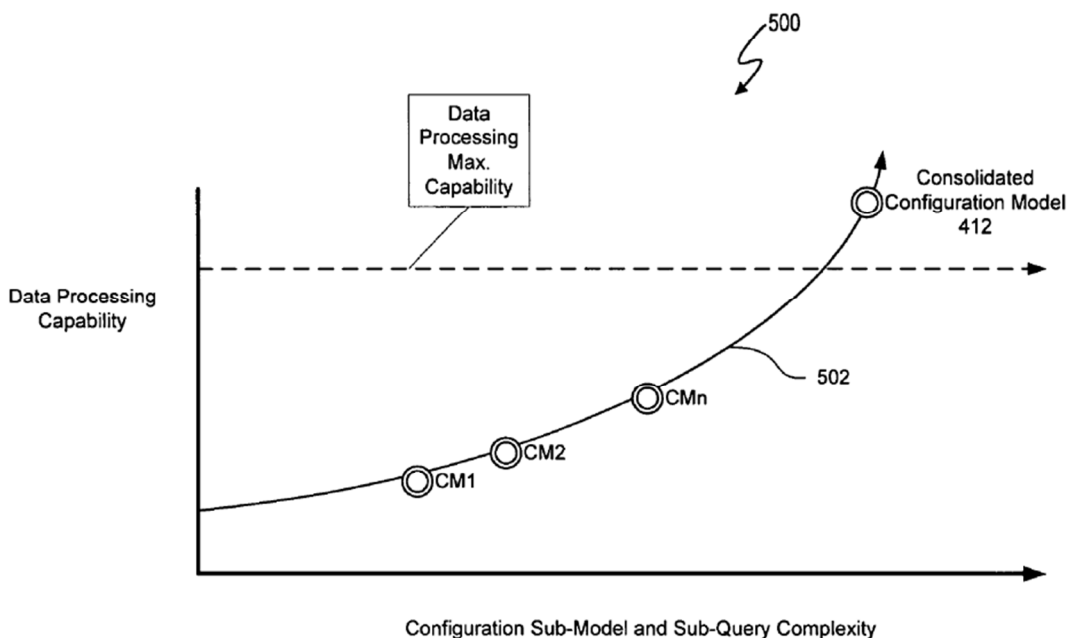


Figure 5

(Ex. 1101 [‘057 Patent] at Fig. 5.)

40. As I discuss in the “Scope and Content of the Prior Art” section and the Ground for Challenge below, it is my opinion that this methodology for processing configuration models was well-known and practiced in the prior art before 2004. It is my opinion that by 2004 a person of ordinary skill would have considered this to be an obvious method for processing data from configuration models.

V. The '057 Patent Prosecution History

41. I have reviewed the prosecution history of the '057 Patent. In the Reasons for Allowance, the only item that the Examiner's identified as missing from the prior art references was the following:

...dividing one or more configuration queries into multiple configuration sub-queries, wherein the multiple configuration sub-queries represent the one or more configuration queries; processing each sub-query using at least one configuration sub-model per sub-query, wherein each configuration sub-model collectively models the configurable product and each configuration sub-models includes...the processing of each sub-query using at least one configuration sub-model per sub-query...

(Ex. 1104 ['057 Patent File History] at 428-429, Notice of Allowability p. 2-3.)

42. The prior art considered most directly during examination was Rising, Patent Application Publication 2003/0187950. (U.S. Patent Application Publication No. 2003/0187950 to Rising; attached as Exhibit 1114.) This is not a configuration or design tool, but a tool for finding digital content by querying into a database of MPEG-7 descriptions. Following a request for continuing examination, the Examiner rejected the claims based on Henson, U.S. Patent 6,167,383, a Dell Computer system that assisted customers with ordering PCs. (Ex. 1104 at 174-195.) The Applicant responded that Henson failed to teach the division of configuration rules for a PC, and therefore the processing of those rules in response to queries, into sub-models. (Ex. 1104 at 227-229.)

VI. Challenged Claims of the '057 Patent Viewed in their Broadest Reasonable Interpretation

43. I understand that in an *inter partes* review at the Patent Office, claims are to be given their broadest reasonable interpretation in light of the specification as would be read by a person of ordinary skill in the relevant art.

44. In applying the claims at issue to the prior art, I have given all of the claim terms their broadest reasonable interpretation in light of the specification, as would be commonly understood by those of ordinary skill in the art at the time the patent was filed.

VII. Scope and Content of the Prior Art (Summary)

45. The '057 Patent acknowledges that prior art systems would perform configuration queries.

Computer assisted product configuration continues to offer substantial benefits to a wide range of users and industries. **FIG. 1 depicts a conventional product configuration process 100 performed by a configuration engine 101.** The configuration process 100 represents one embodiment of an inference procedure. In one embodiment of a conventional inference procedure, configuration query 102 is formulated based on user configuration input, a configuration engine performs the configuration query 102 using a configuration model 104, and the configuration engine provides an answer 106 to the configuration query 102 based on the configuration query 102 and the

contents of the configuration model 104. The answer 106 represents a particular response to the configuration query 102.

(Ex. 1101 ['057 Patent] at 1:12-25, emphasis added.)

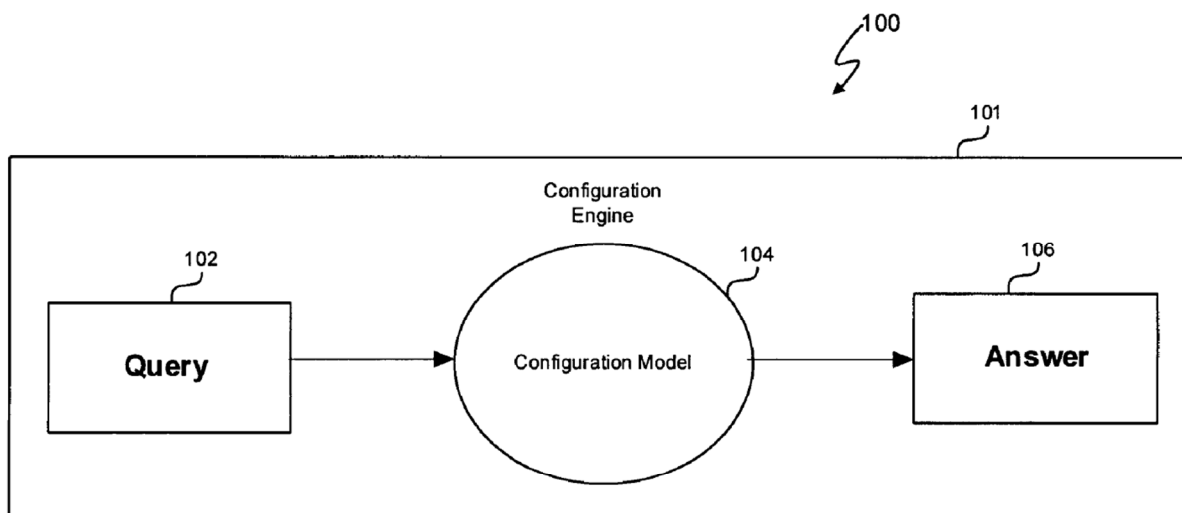


Figure 1 (prior art)

(Ex. 1104 ['057 Patent] at Fig. 1.)

46. A person of ordinary skill in configuration systems would understand “prior art” to refer to well-known computer-assisted configuration systems such as Digital Equipment’s XCON/R1 (went into production in 1980). The system is described in “R1: Revisited: Four Years in the Trenches”¹ (“Bachant”) (Bachant and McDermott, *The AI Magazine*, Fall 1984; attached as Exhibit 1110):

¹ Ex. 1110 is a true and accurate copy of: Judith Bachant, John McDermott, “R1 Revisited: Four Years in the Trenches,” *AI Magazine* Volume 5, Number 3 (1984).

where (McDermott,1980) and (McDermott, 1982). Briefly, given a customer's purchase order, R1 determines what, if any, substitutions and additions have to be made to the order to make it consistent, complete, and produce a number of diagrams showing the spatial and logical relationships among the 50 to 150 components that typically constitute a system. The program has been used on a regular basis by Digital Equipment Corporation's manufacturing organization since January, 1980. R1 has sufficient knowledge of the configuration domain and of the peculiarities of the various configuration constraints that at each step in a configuration task it is usually able to recognize just what to do; thus it ordinarily does not need to backtrack when configuring a computer system.

(Ex. 1110 [Bachant] at 1.))

47. XCON supported grouping rules into categories.

A substantial change to R1 in July of 1982 modified it to deal with a different categorization scheme for components. The component descriptions had been developed exclusively for R1 and were tailored to the configuration task. As Digital developed other knowledge-based systems for other purposes, it became desirable to have a common data base, where the components were categorized in a less ad hoc fashion. Before R1 could use the new descriptions, nearly all of its rules (about 2000 at the time) had to be changed, and for several hundred of these rules, the task of reformulation took considerable thought.

(Ex. 1110 [Bachant] at 6.) A 1982 article, "R1: A Rule-based Configurer of Computer Systems"² ("McDermott") (McDermott; *Artificial Intelligence* 19;

² Ex. 1111 is a true and accurate copy of: John McDermott, "R1: A Rule-Based Configurer of Computer Systems," *Artificial Intelligence* (1982).

Attached as Exhibit 1111) describes the process of configuring a computer system as a “task” and that “The configuration task can be viewed as a hierarchy of subtasks...” (Ex. 1111 [McDermott] at 41 and 49.) The minicomputer’s 420 components are broken down into 15 classes, such as “cabinet” and “unibus device” (*Id.* at 46.)

48. As discussed above in the section on my personal background, the ICAD system was developed in 1985 (partially by me) and was able to configure mechanical systems based on a hierarchical set of rules.

49. “Knowledge-based Configuration of Computer Systems Using Hierarchical Partial Choice,”³ (“Kramer”) (Kramer; *Proceedings of the 1991 IEEE International Conference on Tools for AI* (San Jose, California); Attached as Exhibit 1112) describes a system in which a computer system is broken down into subcomponents:

³ Ex. 1112 is a true and accurate copy of: Bryan M. Kramer, “Knowledge-based Configuration of Computer Systems Using Hierarchical Partial Choice,” IEEE (1991).

Below are two partial descriptions which illustrate the component representation

Workstation (abstract) IS-A Computer subcomponents

display: WorkstationDisplay
number: [1, 2]
keyboard: WorkstationKeyboard
disk: WorkStationDisk
memory: Memory

requires

networkconnection:
NetworkConnection

Ventura_Publisher² (individual) IS-A PublishingPackage

requires

printer: LaserPrintingResource
constraint:
printer.postscript-capability
= true

disk: ExtStorResource
level: 657
consumes: bytesConsumed

memory: MemoryResource

constraint:

memory.supplied-by =
workstation

workstation: WorkStation

properties

price: 2000

(Ex. 1112 [Kramer] at 3.)

50. Users of this 1991 system were able to select their desired system attributes in a graphical user interface:

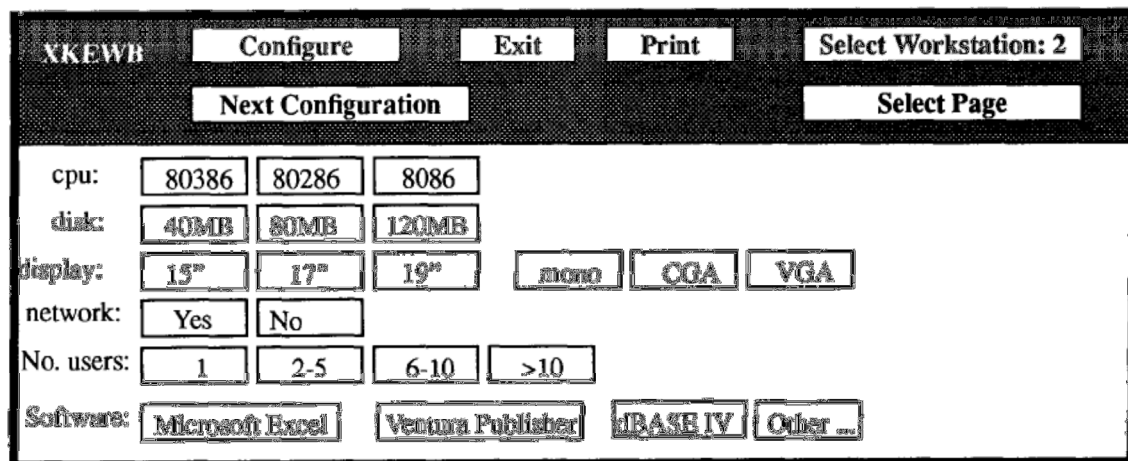


Figure 2: The XKEWB User Interface

(Ex. 1112 [Kramer] at 6.)

51. As noted above, the ICAD system became a commercial product for sales configuration and was ultimately acquired by Oracle Corporation. An Oracle competitor in the business application software market, Baan, offered their own “SalesPLUS” system that is described in “A Configuration Tool to Increase Product Competitiveness”⁴ (“Yu”) (Yu and Skovgaard 1998; *IEEE Intelligent Systems*; attached as Exhibit1113). The authors describe that “salesPLUS is based on the concept of mass customization—that is, product configuration generates customized solutions based on a standard product or product model. It adopts the computer-support-assistant philosophy: it is an assistant interacting with the user.”

⁴ Ex. 1113 is a true and accurate copy of: Bei Yu and Hans Jorgen Skovgaard, “A Configuration Tool to Increase Product Competitiveness,” *IEEE Intelligent Systems* (July/August 1998) pp. 34-41.

(Ex. 1113 [Yu] at 1.)

52. salesPLUS supported submodels:

Product modeling. A product model incorporates all the information that represents products or services. This information is encapsulated in *objects*, which involve resources and constraints. A model can consist of several submodels; for example, a train consists of several cars. Objects might vary from physical parts (such as a screw), to sub-assemblies (for example, a speaker), to whole products (perhaps a car radio system).

Product modulization. This methodology has recently become a hot topic in research and practice. As a configuration-support tool, salesPLUS supports modulization of products into models and submodels. Engineers can work separately on the submodels. These submodels can then be linked into a whole product model. This is important for an integrated development environment; modulizing a large product into manageable modules can reduce product complexity.

(Ex. 1113 [Yu] at 2-3.)

53. One of the examples for salesPLUS was configuring automobiles:

Car configuration with salesPLUS

Now let's look at a specific application of salesPLUS. The configuration problem is from the 900 series models of 1992 Saab automobiles. We'll consider the objects

(Ex. 1113 [Yu] at 5.)

54. Users ended up with conventional menu-based configuration screen,

e.g.,

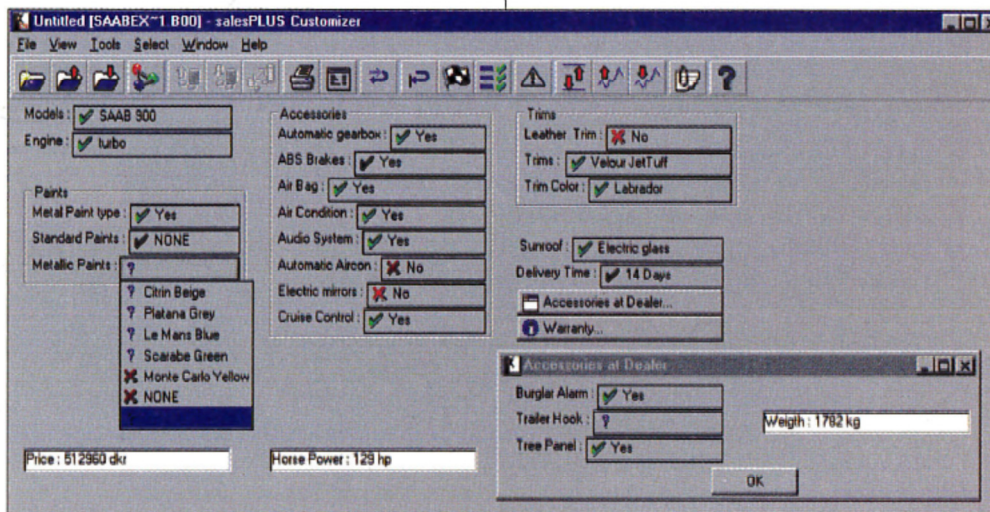


Figure 6. Saab configuration, using the salesPLUS Customizer.

(Ex. 1113 [Yu] at 6.) The reference states that “Wittenborg has used salesPLUS since May 1996,” suggesting that the software was fully functional no later than May 1996. (*Id.* at 8.)

55. In addition to well-known computer-assisted configuration systems like those described above, a person of ordinary skill in the art would also be familiar with the highly-relevant literature, which specifically relates to decomposition of configuration models and queries. Such literature provides details regarding the processes, methods, and systems designed and used for the decomposition of complex configuration problems into tractable configuration subparts. A person of ordinary skill would have appreciated the benefits associated with these types of strategies, which have long been a focus of research both in academia and the industry.

56. An example of the prior art literature surrounding the decomposition

of complex configuration problems is the 1992 article by Alexander Kott, Gerald Agin, and David Fawcett published in the *Journal of Mechanical Design*.⁵ (A. Kott, G. Agin, D. Fawcett, “Configuration Tree Solver: A Technology for Automated Design and Configuration,” *ASME Journal of Mechanical Design* 114(1): 187-195 (1992); “Kott,” attached as Exhibit 1107.) The article provides a particular approach to decomposable configuration problems for the purpose of limiting computational requirements.

In a decomposable configuration problem, all possible configurations of the configuration artifact are implicitly known beforehand. However, the space of all possible configurations is usually very large and to find a configuration that satisfies a particular set of configuration requirements is a computationally explosive problem.

(Ex. 1107 [Kott] at 2-3.) The article further describes techniques for addressing the problems associated with configuration complexity.

Configuration is a process of generating a definitive description of a product that satisfies a set of specified requirements and known constraints. Knowledge-based technology is an important factor in automation of configuration tasks found in mechanical design. In this paper, we describe a configuration technique that is well suited for

⁵ Ex. 1107 is a true and accurate copy of: A. Kott, G. Agin, D. Fawcett, “Configuration Tree Solver: A Technology for Automated Design and Configuration,” *ASME Journal of Mechanical Design* 114(1): 187-195 (1992).

configuring "decomposable" artifacts with reasonably well defined structure and constraints. This technique may be classified as a member of a general class of decompositional approaches to configuration. The domain knowledge is structured as a general model of the artifact, an and-or hierarchy of the artifact's elements, features, and characteristics. The model includes constraints and local specialists which are attached to the elements of the and-or-tree. Given the specific configuration requirements, the problem solving engine searches for a solution, a subtree, that satisfies the requirements and the applicable constraints. We describe an application of this approach that performs configuration and design of an automotive component.

(Ex. 1107 [Kott] at 1.)

57. Additional examples of the type of prior art literature with which a person of skill in the art would have been familiar include the set of papers by L. Anselma, D. Magro, and P. Torasso. These papers include, among others: (1) L. Anselma, D. Magro, and P. Torasso, "Automatically Decomposing Configuration Problems," *AI*IA 2003: Advances in Artificial Intelligence*, Lecture Notes in Computer Science, Volume 2829, pp. 39-52 (2003);⁶ "Anselma," attached as

⁶ Ex. 1108 is a true and accurate copy of: L. Anselma, D. Magro, and P. Torasso, "Automatically Decomposing Configuration Problems," *AI*IA 2003: Advances in*

Exhibit 1108); and (2) D. Magro and P. Torasso, “Decomposing and Distributing Configuration Problems,” *Artificial Intelligence: Methodology, Systems, and Applications*, Lecture Notes in Computer Science, Volume 2443, pp. 81-90 (2002);⁷ (“Magro,” attached as Exhibit 1109). These papers provide detailed analysis regarding particular methodologies that can be used to decompose various configuration problems.

The present paper addresses the issue of decomposing a configuration problem into simpler subproblems by exploiting as much as possible the implicit decomposition provided by the partonomic relations of complex components. The adoption of a structured framework for modeling the configuration domains as well as for expressing the configuration problems plays a major role since the criterion for singling out the classes of bound constraints is based on an analysis of the partonomic slots mentioned in the constraints. The problem decomposition is induced by this partitioning of the constraints into classes.

Artificial Intelligence, Lecture Notes in Computer Science, Volume 2829, pp. 39-52 (2003).

⁷ Ex. 1109 is a true and accurate copy of: D. Magro and P. Torasso, “Decomposing and Distributing Configuration Problems,” *Artificial Intelligence: Methodology, Systems, and Applications*, Lecture Notes in Computer Science, Volume 2443, pp. 81-90 (2002).

(Ex. 1109 [Magro] at 9.)

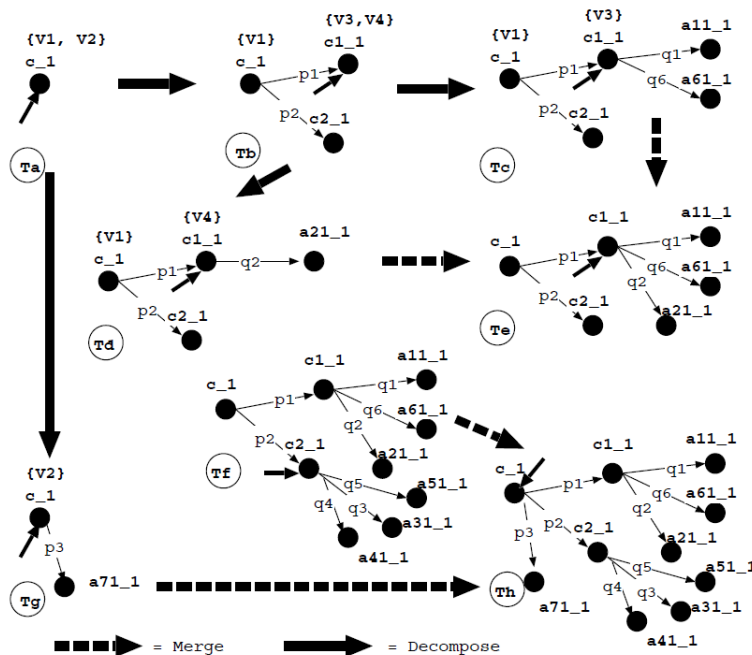


Fig. 3. A configuration example

(Ex. 1109 [Magro] at 8.)

Once a problem has been decomposed into a set of independent subproblems, these could be solved concurrently and with a certain degree of parallelism, potentially reducing the overall computational time. However, also a sequential configuration process can take advantage of the decomposition. In fact, if two subproblems are recognized to be independent, the configurator is aware that no choice made during the configuration process of the first one needs to be revised if it enters a failure path while solving the second one.

(Ex. 1109 [Magro] at 5.)

58. As noted above, Oracle Configurator is the final commercial evolution

of the ICAD system that I co-developed in 1985. It is sold as part of Oracle Applications, a popular Enterprise Resource Planning system that supports nearly every aspect of a company's business. Oracle Applications competes with the SAP system. Both of these systems grew out of the world of computerized accounting and should not be confused with general-purpose RDBMS systems such as Oracle Database. The Oracle Database lets a programmer create tables with columns named and supplied with data types as appropriate. Oracle Applications comes with predefined tables to represent information that most need to store, e.g., invoices that have been sent to customers, the names, addresses, and salaries of all employees, etc.

59. Oracle Configurator was marketed as a tool for supporting sales by businesses that run Oracle Applications.

60. A true and accurate copy Oracle Configurator Developer: User's Guide, April 2002, is attached as Exhibit 1116. The cover page explains that "This document describes how to build and deploy configuration models using Oracle Configurator Developer." Whereas in the 1980s a rule-developer would be typing text in a machine-readable language, much like the work of a computer programmer, the Oracle system circa 2002 offered a graphic user interface with menus. Pages 1-1 to 1-2 explain the overall concept:

Oracle Configurator Developer is the development tool in the Oracle Configurator family of products. It provides a convenient drag-and-drop interface that enables you to rapidly develop a configurator. A **configurator** is a tool for configuring products and services. The configuration process can include assessing customer needs, selecting product and service components, and viewing configurations.

A configurator enables end users to access the parts that make up your product and the rules that govern how those parts fit together. With a configurator, end users can generate any custom product configuration that the rules allow. A configurator brings the expertise of your enterprise to the point of sale, dramatically changing and improving the way you sell products and services.

With Oracle Configurator Developer, you build a Model, configuration rules, and User Interface structure that reflect your enterprise and your end users' requirements. The Model, all configuration rules, and User Interface structure are stored in the Oracle Configurator schema in the Oracle Applications database. The Oracle Configurator schema is part of the Oracle Applications database. There is generally one Oracle Configurator schema per Oracle Applications database instance.

The compiled configuration rules and Model structure exist as the **Active Model** in the Oracle Configurator schema. The Active Model enforces valid configurations based on end user selections. The User Interface definitions of the configuration model function as the **User Interface**. The User Interface also interprets the data in the Oracle Configurator schema and keeps the UI state current as the end user works. In other words, when the end user works in the Oracle runtime Configurator, the Oracle Configurator schema, the Active Model, and the User

Interface determine what is available for selection, what results from selections, and how it is displayed.

(Ex. 1116 [Oracle] at 27-28.) Page 1-4 gives some examples of the types of rules and knowledge that can be embodied in the system:

You must consider what rules you need to build into your configuration model. The design step may include writing a functional specification and other design documents.

Ask yourself questions such as:

- What components must be included in a valid configuration?
- What components are optional?
- What sub-components are compatible with each other?
- What selections affect another selection?
- What are valid default initial selections?
- What rules define the configuration of product families?
- What rules define the relations among product families?

(Ex. 1116 [Oracle] at 30.)

61. As suggested by the highlights above, the history of computer-supported configuration started with top academic researchers tackling what was originally a research challenge. Work on the XCON system began in 1978 at Digital Equipment Corporation, then one of the world's leading vendors of computer systems, and Carnegie-Mellon University, then (as now) one of the world's leading centers of Artificial Intelligence research, and was a fit subject for a person (e.g., John McDermott) with a PhD in Computer Science and a faculty position. Through the 1980s commercial software engineers, such as myself with the ICAD system, introduced rule-based expert systems to the industrial market. When it transpired in the 1990s that most enterprises did not want to put resources into developing a complete body of rules to characterize a three-dimensional

product that could take on a near-infinite number of different configurations, a simpler generation of feature-based configuration systems were developed and offered commercially. These simpler systems also had the advantage that they could potentially run fast enough to give real-time feedback to a shopper on a Web site. They may have lacked the power of the ICAD system, for example, in being able to design a steel structure with thousands of parts, but they also didn't require multiple person-years of rule development or multiple minutes, if not hours, of runtime.

62. Thus did the exciting research result of 1980 become by 2002 an off-the-shelf product available to the tens of thousands of enterprises relying on Oracle Applications.

VIII. Prior Art: Loomans, U.S. Patent 7,873,503

63. Loomans⁸ is a U.S. Patent that was filed on November 18, 2002, and issued on January 18, 2011. It is also my understanding that Loomans is prior art as it was filed before the filing date of the '057 Patent. Loomans is assigned to Siebel Systems, company founded by a former Oracle executive in 1993 and later acquired by Oracle. Siebel's original specialty was "sales force automation," i.e., helping salespeople.

⁸ Ex. 1105 is a true and accurate copy of U.S. Patent No. 7,873,503, which was filed on November 18, 2001 and issued on January 18, 2011.

64. Loomans discloses a system that is directed at the same market as the Oracle Configurator. As with prior art systems, Loomans supports a hierarchical product model:

The invention provides techniques to configure complicated entities using sub-configuration, which effectively partitions the configuration of a complicated top-level entity into a set of configurations of smaller sub-level entities (i.e., components of the top-level entity) in combination with a configuration of a "simplified" top-level entity. The top-level entity may be represented by and configured via a parent model, and each sub-configurable component may be represented by and configured via a child sub-model.

(Ex. 1105 [Loomans] at 1:57-65, emphasis added.)

65. Presumably for run-time speed and simplicity, Loomans compiles down all of its rules into a “configuration table” that is illustrated in FIG. 2B and explained in 4:17-30. Thus validating a configuration at run-time does not require an artificial intelligence technique of evaluating rules from a rules database, but only looking up to see if a configuration is present in a table of valid configurations. This has the disadvantage that it is hard to the system to explain why a configuration is invalid other than “Could not find configuration in table.” Loomans thus adds an “exception table” in which explanations for “why the combinations are incorrect” may be found. (Ex. 1105 [Loomans] at 11:39-55)

IX. Prior Art: “A Customization Approach for Structure Products in Electronic Shops” (“Stahl”)

66. Stahl is a printed publication titled “A Customization Approach for Structured Products in Electronic Shops,”⁹ which was published during the 13th International Bled Electronic Commerce Conference in Bled, Slovenia in June 2000. It is also my understanding that Stahl is prior art as it was published before the filing date of the ‘057 Patent.

67. Stahl was authored by three members of the “Artificial Intelligence—Knowledge-Based Systems Group” within the Department of Computer Science at the University of Kaiserslauten (Germany) and describes a system that is a more direct descendant of academic approaches to the configuration problem. The system described in Stahl is built on top of an existing case-based reasoning (“CBR”) tool for handling sets of rules: “This prototype consists of an extension of the commercial CBR tool CBR-Works which has been developed jointly by the University of Kaiserslauten and Tecinno GmbH.” (Ex. 1106 [Stahl] at 8.)

68. Stahl references a 1999 paper “CBR-Works, A State-of-the-Art Shell

⁹ Ex. 1106 is a true and accurate copy of: Armin Stahl, Ralph Bergmann, Sascha Schmitt, “A Customization Approach for Structure Products in Electronic Shops,” 13th International Bled Electronic Commerce Conference June 19-21, 2000. Available at (<https://domino.fov.uni-mb.si/ecomframes.nsf/pages/bled2000>.)

for Case-Based Application Building,”¹⁰ by Stefan Schulz (“CBR”). I have attached this prior art reference as Exhibit 1117.

69. CBR-Works is capable of modeling features in more detail than Loomans, e.g., the overall acceptability of a configuration will be lower as the price rises, but not simply “Above X is unacceptable; Below X is acceptable.” Also, customers may have a problem with very low prices: “fig 5 regarding a customers [sic] ‘feeling of an acceptable price’ being different in a retrieved case to a specified value in the query. A higher price only is accepted up to specific limit quickly dropping the higher it is. The situation is similar offering products with lower prices, as a customer usually thinks of lower quality by a lower price once the negative limit is passed.” (Ex. 1117 [CBR] at 6.) In Loomans, all attributes are of equal weight. The lack of a match in any attribute can cause the system to respond “Not found in configuration table.” In Stahl, however, attributes can have different weights and be classified as either mandatory or not: “For retrieval purposes, attributes have three additional, functional properties: one for defining its weight, i.e., its importance in respect to the other attributes of the concept, a property for defining whether an attribute is discriminant for retrieval or will be ignored, and another property defining if an attribute is mandatory for a

¹⁰ Ex. 1117 is a true and accurate copy of: Stefan Schulz, “CBR-Works A State-of-the-Art Shell for Case-Based Application Building,” (1999).

case to be valid.” (Ex. 1117 [CBR] at 2-3.)

70. The additional richness of the rules environment used as a starting point for Stahl gives Stahl the ability to handle gracefully cases in which an exact match cannot be found for a consumer’s preferences. This is explained on Page 6 of Stahl:

At the beginning of the adaptation cycle, it has to be determined if the base product includes parts that do not fulfill the requirements of the query. This can be done by an examination of the local similarities between the different product parts and the related part-queries. Product parts with a low similarity to the respective part-query are called weak parts. The determination of these parts is the task of the part selection process. If the product includes no weak parts at all, the adaptation process is finished and the given product is presented to the customer as suggested customization result. Generally, we can distinguish between two causes for weak parts. The first one occurs if the retrieved standard product does not include components for required product parts. For example, if a customer wants a PC with a modem, but the base-product includes no modem at all. Here, we can speak of a missing component. The second cause is given if installed components do not fulfill the technical requirements of the customer, e.g., if the PC still contains a modem but this modem provides only an insufficient data transmission rate. Then, we can speak of a weak component. During the adaptation cycle, a selected weak part leads to a new similarity-based retrieval, called component retrieval. In this process, the part-query corresponding to the selected weak part is used to retrieve a collection of alternative components from the component case-base. The elements of this collection are arranged by their similarity to the query in a decreasing order.

(Ex. 1106 [Stahl] at 6.)

71. Stahl’s evaluation process also allows for the system to work in such a way that intermediate solutions will violate various rules, but with an eventual goal of producing a consistent output. See page 7:

Preservation of Consistency. Up to now, we have assumed that a product modification during one iteration of the adaptation cycle must even lead to a consistent product. This means, only adaptation orders that preserve the consistency of the product are allowed.

Temporary Loss of Consistency. In contrast to the previously described approach it is also possible to allow the temporary loss of consistency during the adaptation of a base product. That means, if the product validation process determines the violation of constraints after the component exchange process, this must not necessarily lead to an immediately retraction procedure. It is rather possible only to notice the loss of consistency and to continue the whole adaptation process without the retraction of the last modification. But, to find a final solution that can be presented to the customer it is of course necessary to re-establish the consistency during the further adaptation.

(Ex. 1106 [Stahl] at 7.)

72. This is not to say that Stahl is better than Loomans. They represent different points in a large space of academic and commercial approaches to computer-assisted configuration. Stahl may work better in situations where there is some flexibility in what can work for a customer. Loomans may be more appropriate in situations where there are rigid constraints on what can be bought and used and/or on what can be manufactured (example: FAA-certified aircraft). Loomans also should offer faster runtime performance.

X. Grounds for Challenge

A. Ground 1 – Claims 17, 30, 44 And 45-46 Obvious Based On Loomans In View Of Stahl And The General Knowledge Of A Person Having Ordinary Skill In The Art

1. Analysis of Claims 17, 30, and 44

a. Claim 17

... [17.0] A method for using a computer system, wherein the computer system includes computer assisted configuration technology to respond to one or more configuration queries using configuration sub-models, the method comprising:

73. It is my opinion that Loomans in view of Stahl discloses a method of using a computer system to respond to one or more configuration queries using configuration sub-models.

74. For example, Loomans discloses configuring a parent configuration model by configuring and validating child sub-models created by portioning the parent configuration model.

Techniques to performing sub-configuration of components of an entity. **In one method, the entity is configured via a parent model and each sub-configurable component is configured via one of a number of sub-models.** Initially a selection to configure a particular sub-configurable component of the entity is received, and a sub-model for the selected component is identified. One or more values for one or more features of the selected component are received (e.g., from the parent model or via the sub-model) and form a configuration for the component, **which is then validated based on the associated sub-model and the received values.** Configuration of the entity is also validated based on the parent model and the validated configuration for the selected component. Feedbacks may be provided for each configuration of the parent model and sub-models. The data

for the parent model and sub-models may be localized or globalized.

(Ex. 1105 [Loomans] at Abstract, emphasis added.)

The invention provides techniques to configure complicated entities using sub-configuration, which effectively partitions the configuration of a complicated top-level entity into a set of configurations of smaller sub-level entities (i.e., components of the top-level entity) in combination with a configuration of a "simplified" top-level entity. The top-level entity may be represented by and configured via a parent model, and each sub-configurable component may be represented by and configured via a child sub-model.

(Ex. 1105 [Loomans] at 1:57-65, emphasis added.)

As discussed above, a PHOSITA would have understood that the teachings of configuring a sub-model or validating a sub-model described in Loomans corresponds to the “query” nomenclature of the ‘057 Patent.

75. Loomans further discloses that one or more configuration queries are responded to via the disclosed sub-models.

With sub-configuration, a complicated entity may be more efficiently modeled, configured, and validated. Each sub-configurable option of the entity may be configured and validated via a respective child sub-model. Generally, sufficient **information** is made available to the child sub-model such that the associated option can be validly configured. **The required information may be**

provided from the parent level, queried and entered at the sub-level via the child sub-model, and/or provided via some other mechanisms.

(Ex. 1105 [Loomans] at 4:59-67, emphasis added.)

76. Loomans describes a computer system, which is capable of implementing the disclosed configuration technology.

FIG. 8 is a simplified diagram of an embodiment of a configuration system 800 that may be capable of implementing various aspects and embodiments of the invention. In this embodiment, configuration system 800 is implemented on a set of one or more host servers 810 that couple to and interact with one or more client **computers** 830 via direct connections, a computer network (e.g., the Internet), and/or some other means. Host server(s) 810 further couples to a database server 820 that stores data (typically in a "raw" form) used by the system.

(Ex. 1105 [Loomans] at 16:26-35, emphasis added.)

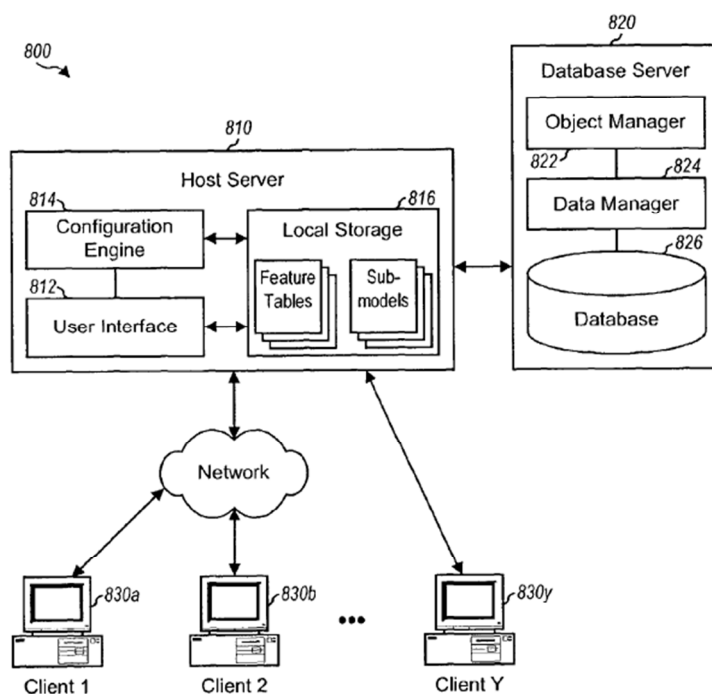


FIG. 8

(Ex. 1105 [Loomans] at Fig. 8.)

77. Loomans further describes an example in Figure 9 of a computer used as host computer (810) or the client computers from Figure 8.

FIG. 9 is a block diagram of an embodiment of a computer system 900 that may be used to implement host server 810 or client computers 820. System 900 includes a bus 908 that interconnects major subsystems such as one or more processors 910, a memory subsystem 912, a data storage subsystem 914, an input device interface 916, an output device interface 918, and a network interface 920. Processor(s) 910 perform many of the processing functions for system 900 and communicate with a number of peripheral devices via bus 908.

(Ex. 1105 [Loomans] at 17:42-50, emphasis added.)

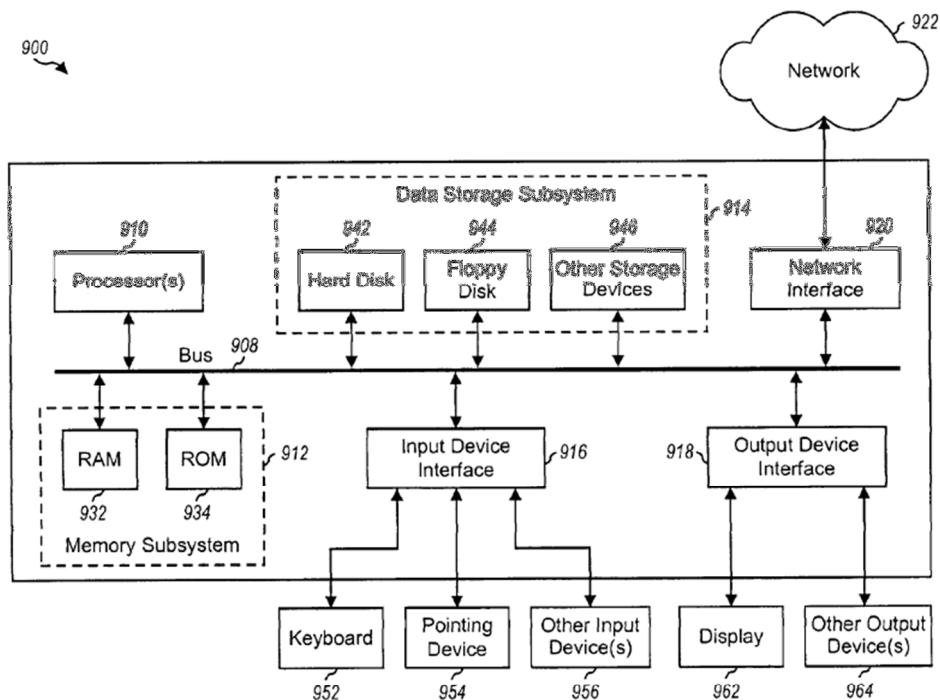


FIG. 9

(Ex. 1105 [Loomans] at Fig. 9.)

78. Stahl discloses partitioning a query into sub-problems (i.e., sub-queries), which are then solved and combined back together to form the overall solution of the configuration problem (i.e., configuration query)

Query. The starting point of the configuration process is the query represented by an incomplete instantiation of the compositional structure. When looking at the example query shown in Fig. 3, we can interpret the root note as our actual problem, i.e., we are searching a PC with a set of special properties. To reach this goal it is necessary to select appropriate components that fulfill the properties of the respective part-queries. In our example, one part-

query states that the PC should have a hard-disk with 12GB of capacity. To fulfill this demand we can, e.g., integrate the concrete hard-disk "Maxtor91303D6" for the hard-disk part in the PC. **Thus, we can interpret the different leaf nodes of the query as sub-problems that must be solved to solve the overall problem, i.e., the configuration of the required PC.** If we have found suitable sub-solutions, i.e. suitable components, for every part-query, **we have to combine these sub-solutions to a final solution for the overall configuration problem.**

(Ex. 1106 [Stahl] at 4-5, emphasis added.)

79. A person having ordinary skill in the art would have understood that the sub-problems disclosed in Stahl represent a form of sub-query, which Stahl discloses are solved with "suitable sub-solutions" (i.e., answers) by determining the suitable parts for a particular "part-query". (Ex. 1106 [Stahl] at 5.) A person having ordinary skill in the art would have further understood that the sub-models described in Loomans could be partitioned in such a way to provide the solutions (i.e., answers) to a particular part-query raised in a sub-problem disclosed in Stahl. A person having ordinary skill in the art would have further appreciated that the models and sub-models described in Loomans could have been used to process solutions and sub-solutions to the Query and sub-problems in Stahl. Indeed, Loomans discloses that "each sub-configurable component is configured via one of a number of sub-models." (Ex. 1105 [Loomans] at Abstract.) Thus, a person

having ordinary skill in the art would have understood that the sub-problems described in Stahl could be answered (i.e., solved) using the sub-models described in Loomans. Such models (and the rules contained therein) are commonly used during the resolving stage of a configuration problem in a configuration system.

80. Finally, Stahl discloses “[i]f we have found suitable sub-solutions, i.e. suitable components, for every part-query, **we have to combine these sub-solutions to a final solution for the overall configuration problem.**” (Ex. 1106 [Stahl] at 5, emphasis added.) Thus, Stahl discloses recombining the sub-solutions to the sub-problems to provide a response to the configuration problem in the form of a combined solution.

81. Accordingly, it is my opinion that Loomans in view of Stahl discloses *a method for using a computer system, wherein the computer system includes computer assisted configuration technology to respond to one or more configuration queries using configuration sub-models.*

... [17.1] dividing a consolidated configuration model into multiple configuration sub-models; and

82. Loomans discloses dividing a consolidated parent model into multiple child sub-models.

The invention provides techniques to configure complicated entities using sub-configuration, which provides numerous benefits. **Sub-configuration effectively partitions the overall configuration of a**

complicated top-level entity (e.g., computer system 100) into a set of configurations of smaller sub-level entities (e.g., Drive Bays 1, 2, and 3) in combination with a less complicated configuration of a "simplified" top-level entity. The top-level entity may be represented by and configured via a parent model, and each sub-configurable component (or sub-level entity) may be represented by and configured via a child sub-model.

(Ex. 1105 [Loomans] at 4:48-58, emphasis added.)

The invention provides techniques to configure complicated entities using sub-configuration, **which effectively partitions the configuration of a complicated top-level entity into a set of configurations of smaller sub-level entities (i.e., components of the top-level entity) in combination with a configuration of a "simplified" top-level entity. The top-level entity may be represented by and configured via a parent model, and each sub-configurable component may be represented by and configured via a child sub-model.**

(*Id.* at 1:57-65, emphasis added.)

83. Accordingly, it is my opinion that Loomans in view of Stahl discloses *dividing a consolidated configuration model into multiple configuration sub-models.*

... [17.2] performing with the computer system:

84. Loomans describes a computer system, which is capable of

implementing the disclosed configuration technology.

FIG. 8 is a simplified diagram of an embodiment of a configuration system 800 that may be capable of implementing various aspects and embodiments of the invention. In this embodiment, configuration system 800 is implemented on a set of one or more host servers 810 that couple to and interact with one or more client computers 830 via direct connections, a computer network (e.g., the Internet), and/or some other means. Host server(s) 810 further couples to a database server 820 that stores data (typically in a "raw" form) used by the system.

(Ex. 1105 [Loomans] at 16:26-35, emphasis added.)

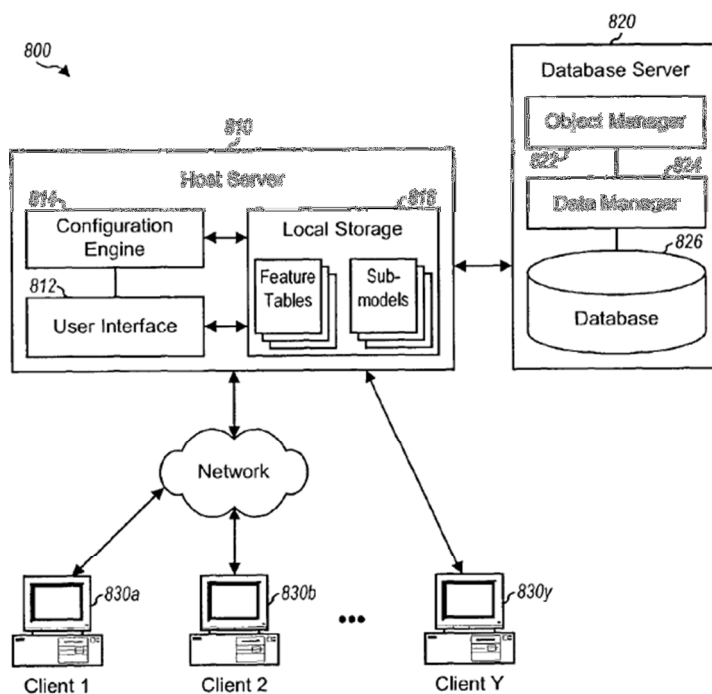


FIG. 8

(Ex. 1105 [Loomans] at Fig. 8.)

85. Loomans further describes an example in Figure 9 of a computer used

as host computer (810) or the client computers from Figure 8.

FIG. 9 is a block diagram of an embodiment of a computer system 900 that may be used to implement host server 810 or client computers 820. System 900 includes a bus 908 that interconnects major subsystems such as one or more processors 910, a memory subsystem 912, a data storage subsystem 914, an input device interface 916, an output device interface 918, and a network interface 920. Processor(s) 910 perform many of the processing functions for system 900 and communicate with a number of peripheral devices via bus 908.

(Ex. 1105 [Loomans] at 17:42-50, emphasis added.)

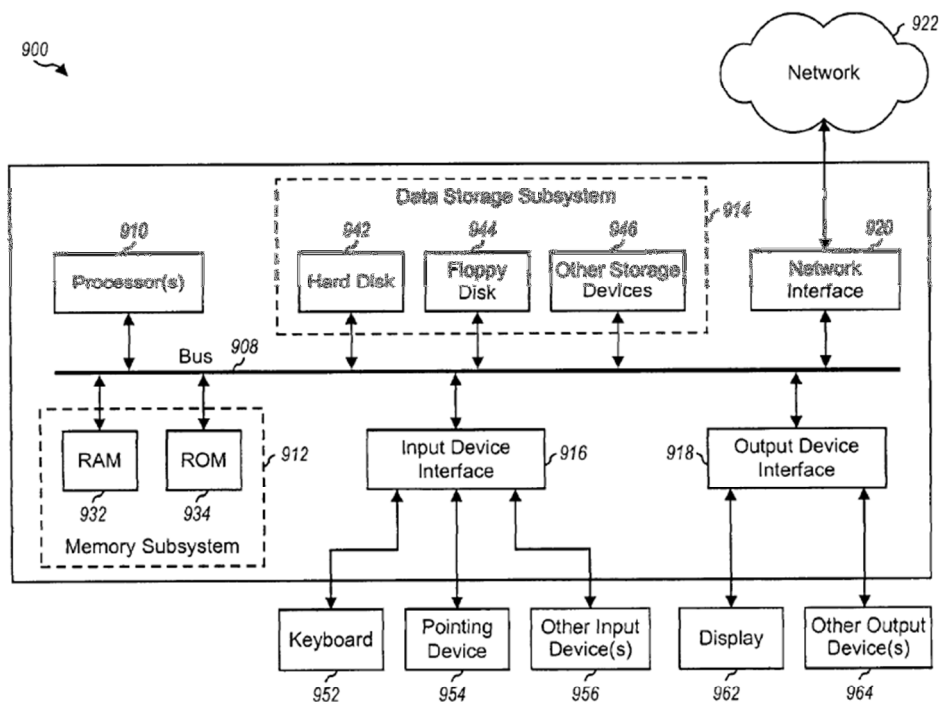


FIG. 9

(Ex. 1105 [Loomans] at Fig. 9.)

86. Likewise, Stahl discloses implementing a demo of its disclosed

configuration methods over the world wide web.

To evaluate the functionality of our configuration approach we have implemented a generic prototype for the described configuration process. This prototype consists of an extension of the commercial CBR tool CBR-Works which has been developed jointly by the University of Kaiserslautern and Tecinno GmbH. **To be accessible over the World Wide Web the prototype also provides two respective interfaces for the demand acquisition (see Fig. 5) and the result presentation.**

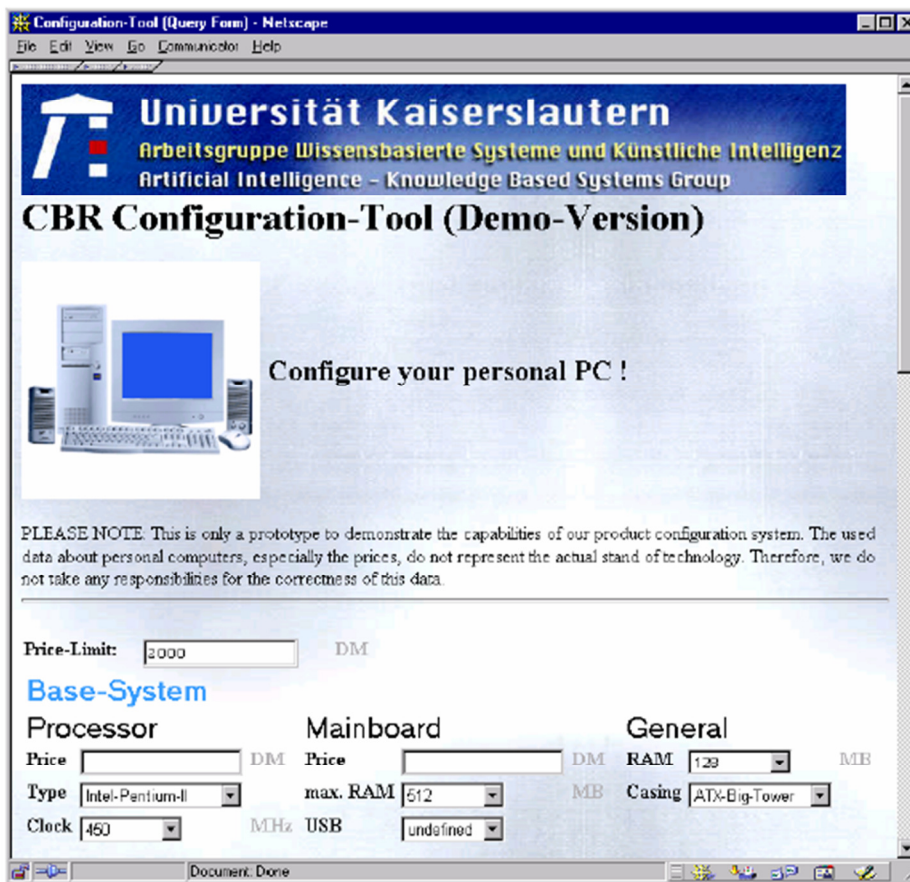


Fig. 5. Implemented demo version.

(Ex. 1106 [Stahl] at Fig. 5.)

87. A person having ordinary skill in the art would have understood that

the demo would need to be implemented on a computer system to be functional over the World Wide Web.

88. Accordingly, a person having ordinary skill in the art would have understood that Loomans in view of Stahl discloses performing steps [17.3]-[17.9] (below) with the *computer system*.

... [17.3] responding to the one or more configuration queries representing questions involving configuration of a configurable product, wherein responding to the one or more configuration queries comprises:

89. Loomans discloses configuring and validating one or more features of a selected component using an associated sub-model.

A specific embodiment of the invention provides a method for performing sub-configuration of components of an entity. In the method, **the entity is configured via a parent model and each sub-configurable component is configured via one of a number of sub-models**. Initially a selection to configure a particular sub-configurable component of the entity is received, and a sub-model for the selected component is identified. One or more values for one or more features of the selected component are received (e.g., from the parent model or via the sub-model) and form a configuration for the component, which is then validated based on the associated sub-model and the received values. **Configuration of the entity is also validated based on the parent model and the validated configuration for the selected component.**

(Ex. 1105 [Loomans] at 2:25-38, emphasis added.)

90. Thus, Loomans discloses configuring each sub-configurable component via sub-models, and then configuring/validating the overall entity via the parent model. As discussed above, the act of selecting a component, and then validating the configuration, includes a configuration query. Indeed, the query is used in the configuration of the product. Thus, a person having ordinary skill in the art would have understood that Loomans' disclosure of selecting and validating a particular component based on an associated sub-model would effectively be responding to one or more configuration queries.

91. Loomans further discloses querying for information via the child sub-model.

With sub-configuration, a complicated entity may be more efficiently modeled, configured, and validated. **Each sub-configurable option of the entity may be configured and validated via a respective child sub-model.** Generally, sufficient information is made available to the child sub-model such that the associated option can be validly configured. **The required information may be provided from the parent level, queried and entered at the sub-level via the child sub-model,** and/or provided via some other mechanisms.

(Ex. 1105 [Loomans] at 4:59-67, emphasis added.)

92. Thus, Loomans discloses that the required information for configuration is queried at the sub-level via the child sub-model. A PHOSITA

would have understood that this statement from Loomans teaches that the sub-level queries are processed via child sub-models. Indeed, Loomans discloses querying for the information from the child sub-model so that the information can then be entered. The query is processed and answered by the child sub-model.

93. Further, Stahl discloses interpreting a query as a series of “sub-problems”, which can be solved with “sub-solutions.”

Thus, **we can interpret the different leaf nodes of the query as sub-problems** that must be solved to solve the overall problem, i.e., the configuration of the required PC. **If we have found suitable sub-solutions, i.e. suitable components, for every part-query**, we have to combine these sub-solutions to a final solution for the overall configuration problem.

(Ex. 1106 [Stahl] at 5, emphasis added.)

94. A person having ordinary skill in the art reading Stahl in view of Loomans would have understood that each sub-problem (part-query) could have been solved via one or more sub-models, which address the parts at issue for the particular sub-problem. Indeed, as discussed above, Loomans discloses using child sub-models to process (i.e., answer) queries, and models are commonly used during the configuration and validation process.

95. Finally, Stahl discloses that “[i]f we have found suitable sub-solutions, i.e. suitable components, for every part-query, **we have to combine**

these sub-solutions to a final solution for the overall configuration problem.”

(Ex. 1106 [Stahl] at 5, emphasis added.) Thus, Stahl discloses that the sub-solutions to the sub-problems (i.e., sub-queries), which as described above could be determined using the child sub-models described in Loomans, are then combined to solve the configuration problem – i.e., the top level configuration query.

96. Finally, both Loomans and Stahl disclose using the described configuration method to configure a PC. (Ex. 1105 [Loomans] at 3:21-4:67; Ex. 1106 [Stahl] at 4-5.) Thus, both Loomans and Stahl disclose responding to the one or more configuration queries representing questions **involving configuration of a configurable product**.

97. Accordingly, it is my opinion that Loomans in view of Stahl discloses *responding to the one or more configuration queries representing questions involving configuration of a configurable product*.

... [17.4] dividing one or more configuration queries into multiple configuration sub-queries, wherein the multiple configuration sub-queries represent the one or more configuration queries;

98. [Intentionally left blank].

99. Stahl discloses partitioning a query into sub-problems (i.e., sub-queries), which are then solved and combined back together to form the overall

solution to the configuration problem (i.e., configuration query)

Query. The starting point of the configuration process is the query represented by an incomplete instantiation of the compositional structure. When looking at the example query shown in Fig. 3, we can interpret the root node as our actual problem, i.e., we are searching a PC with a set of special properties. To reach this goal it is necessary to select appropriate components that fulfill the properties of the respective part-queries. In our example, one part-query states that the PC should have a hard-disk with 12GB of capacity. To fulfill this demand we can, e.g., integrate the concrete hard-disk "Maxtor91303D6" for the hard-disk part in the PC. **Thus, we can interpret the different leaf nodes of the query as sub-problems that must be solved to solve the overall problem, i.e., the configuration of the required PC.** If we have found suitable sub-solutions, i.e. suitable components, for every part-query, we have to combine these sub-solutions to a final solution for the overall configuration problem.

(Ex. 1106 [Stahl] at 4-5, emphasis added.)

100. A person having ordinary skill in the art would have understood that the sub-problems disclosed in Stahl when combined represent the original query. Indeed, Stahl discloses that "we can interpret **the different leaf nodes of the query as sub-problems** that must be solved to solve the overall problem." (Ex. 1106 [Stahl] at 5, emphasis added.) Thus, the sub-problems (leaf nodes) ultimately make up the complete query, as displayed in the enhancement of Figure 3 below.

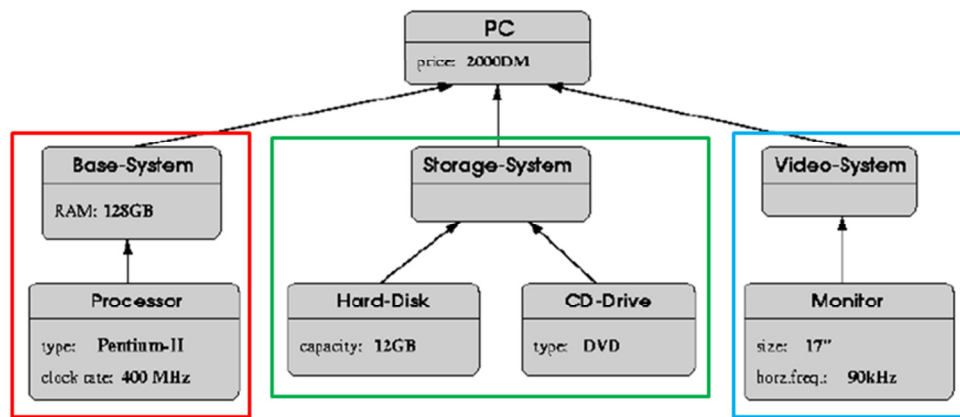


Fig. 3. An Example Query.

Base-System sub-problem + Storage System sub-problem + Video-System sub-problem = PC Problem (Query)

(Ex. 1106 [Stahl] at Fig. 3)

101. A person having ordinary skill in the art would have understood that Loomans discloses receiving one or more configuration queries. Indeed, the use of configuration queries is consistent with conventional processing of a configuration model, which would include the parent model described in Loomans.

102. Loomans discloses dividing a top-level entity, such as a computer system, represented via a parent model into sub-configurable components represented by child sub-models.

The invention provides techniques to configure complicated entities using sub-configuration, which provides numerous benefits. Sub-configuration effectively **partitions** the overall configuration of a complicated top-level entity (e.g., computer system 100) into a set of configurations of smaller sub-level entities (e.g., Drive Bays 1, 2, and 3) in combination with a less complicated configuration of a

"simplified" top-level entity. **The top-level entity may be represented by and configured via a parent model, and each sub-configurable component (or sub-level entity) may be represented by and configured via a child sub-model.**

(Ex. 1105 [Loomans] at 4:48-58, emphasis added.)

103. Loomans next discloses validating sub-configurable options with a child sub-model, a process which includes querying each child sub-model.

With sub-configuration, a complicated entity may be more efficiently modeled, configured, and validated. **Each sub-configurable option of the entity may be configured and validated via a respective child sub-model.** Generally, sufficient information is made available to the child sub-model such that the associated option can be validly configured. **The required information may be provided from the parent level, queried and entered at the sub-level via the child sub-model,** and/or provided via some other mechanisms.

(Ex. 1105 [Loomans] at 4:59-67, emphasis added.)

104. A person having ordinary skill in the art would have understood that the one or more configuration queries received in regards to the top entity (parent) model in Loomans would have needed to be divided into sub-queries when being applied to each child sub-model. Otherwise, the system would be unable to determine the appropriate sub-model to apply to the portion of the query being configured.

105. Accordingly, it is my opinion that Loomans in view of Stahl discloses

dividing one or more configuration queries into multiple configuration sub-queries, wherein the multiple configuration sub-queries represent the one or more configuration queries.

... [17.5] processing each sub-query using at least one configuration sub-model per sub-query,

106. Loomans discloses dividing a top-level entity, such as a computer system, represented by a parent model, into sub-configurable components represented by child sub-models.

The invention provides techniques to configure complicated entities using sub-configuration, which provides numerous benefits. Sub-configuration effectively **partitions** the overall configuration of a complicated top-level entity (e.g., computer system 100) into a set of configurations of smaller sub-level entities (e.g., Drive Bays 1, 2, and 3) in combination with a less complicated configuration of a "simplified" top-level entity. **The top-level entity may be represented by and configured via a parent model, and each sub-configurable component (or sub-level entity) may be represented by and configured via a child sub-model.**

(Ex. 1105 [Loomans] at 4:48-58, emphasis added.)

107. Loomans next discloses querying for information via the child sub-model.

With sub-configuration, a complicated entity may be more efficiently modeled, configured, and validated. **Each sub-configurable option**

of the entity may be configured and validated via a respective child sub-model. Generally, sufficient information is made available to the child sub-model such that the associated option can be validly configured. **The required information may be provided from the parent level, queried and entered at the sub-level via the child sub-model,** and/or provided via some other mechanisms.

(Ex. 1105 [Loomans] at 4:59-67, emphasis added.)

108. Loomans further discloses configuring and validating one or more features of a selected component using an associated sub-model.

A specific embodiment of the invention provides a method for performing sub-configuration of components of an entity. In the method, **the entity is configured via a parent model and each sub-configurable component is configured via one of a number of sub-models. Initially a selection to configure a particular sub-configurable component of the entity is received, and a sub-model for the selected component is identified. One or more values for one or more features of the selected component are received (e.g., from the parent model or via the sub-model) and form a configuration for the component, which is then validated based on the associated sub-model and the received values.** Configuration of the entity is also validated based on the parent model and the validated configuration for the selected component.

(Ex. 1105 [Loomans] at 2:25-38, emphasis added.)

109. As discussed above, the act of selecting a component, and then

validating the configuration, includes a configuration query. Thus, a PHOSITA would have understood that Loomans' disclosure of selecting and validating a particular component (represented by a sub-model) would include querying that sub-model .

110. Further, Stahl discloses interpreting a query as a series of “sub-problems”, which can be solved with “sub-solutions.”

Thus, **we can interpret the different leaf nodes of the query as sub-problems** that must be solved to solve the overall problem, i.e., the configuration of the required PC. **If we have found suitable sub-solutions, i.e. suitable components, for every part-query**, we have to combine these sub-solutions to a final solution for the overall configuration problem.

(Ex. 1106 [Stahl] at 5, emphasis added.)

111. As described above for element [17.0], a person having ordinary skill in the art would have understood that the sub-problems disclosed in Stahl represent a form of sub-query, which Stahl discloses are solved with “suitable sub-solutions” (i.e., answers) by determining the suitable parts for a particular “part-query”. (Ex. 1106 [Stahl] at 5.) A person having ordinary skill in the art would have further understood that the sub-models described in Loomans could be partitioned in such a way to provide the solutions (i.e., answers) to a particular part-query raised in a sub-problem disclosed in Stahl. Indeed, Loomans discloses that “each sub-

configurable component is configured via one of a number of sub-models.” (Ex. 1105 [Loomans] at Abstract.) And Loomans further discloses that “[t]he required information [for configuration] may be provided from the parent level, **queried and entered at the sub-level via the child sub-model.**” (*Id.* at 4:64-66, emphasis added.) Thus, a person having ordinary skill in the art would have understood that the sub-problems described in Stahl could be processed (i.e., solved/answered) using the sub-models described in Loomans. The feature-specific sub-models taught in Loomans would have informed a PHOSITA how to process and solve the sub-problems taught in Stahl for specific features.

112. Accordingly, it is my opinion that Loomans in view of Stahl discloses *processing each sub-query using at least one configuration sub-model per sub-query.*

... [17.6] wherein each configuration sub-model collectively models the configurable product and each configuration sub-model includes data to define compatibility relationships between parts included in the configuration sub-model,

113. Loomans discloses a top level entity that is represented by a parent model, which is then partitioned into sub-level entities that are represented as child sub-models and include the components of the top-level entity.

The invention provides techniques to configure complicated entities using sub-configuration, which **effectively partitions the**

configuration of a complicated top-level entity into a set of configurations of smaller sub-level entities (i.e., components of the top-level entity) in combination with a configuration of a "simplified" top-level entity. The top-level entity may be represented by and configured via a parent model, and **each sub-configurable component may be represented by and configured via a child sub-model.**

(Ex. 1105 [Loomans] at 1:57-65, emphasis added.)

114. Thus, Loomans discloses dividing the top-level entity, which is represented as the parent model, into sub-level entities that are represented as child sub-models. A person having ordinary skill in the art would have understood that the partitioned sub-models taught by Loomans would collectively include all of the information from the top-entity parent model to ensure that the full configurable product remains intact for configuration. This is consistent with the purpose of the invention of Loomans, which is to “configure complicated entities.” (*Id.* at 1:57-58.)

115. Loomans further discloses that each sub-configurable option may be configured based on its own set of part options/choices.

Each top-level sub-configurable option may be configured based on its associated sets of sub-level options. For example, each of Drive Bays 1, 2, and 3 may be used to install a disk drive, a CD-drive, or a hard disk (i.e., three possible options), or nothing at all (which may be a default). The Disk Drive, CD-Drive, and Hard Disk options

may each be further associated with one or more sets of choices that are specific for that option.

(Ex. 1105 [Loomans] at 3:41-49, emphasis added.)

116. Loomans discloses that the sub-models include data maps, which defines the available parts in a sub-model for a particular option.

As shown in FIG. 3A, structure 300 includes a sub-model mapping set 310 that includes a number of (N) elements, one element for each sub-configurable option at the top level. For example, **sub-model mapping set 310 may include four elements for options C, D, E, and F (i.e., Drive Bays 1, 2, and 3 and Storage) of computer system 100.** Each element of mapping set 310 is associated with a respective sub-model map 320. For example, **option F (Storage) may be associated with a sub-model map that includes the two available types of storage devices ("ABC" and "XYZ").** Similarly, **option C (Drive Bay 1) may be associated with a sub-model map that includes the three available drive types (Disk Drive, CD-Drive, and Hard Disk).** Each sub-model map 320 may further be associated with a set of (M) sub-models 330, where M may be one or greater. For example, the sub-model map for option F (Storage) may include a first sub-model for the ABC storage device and a second sub-model for the XYZ storage device.

(Ex. 1105 [Loomans] at 6:30-46, emphasis added.)

117. Thus, Loomans discloses that each child sub-model includes information, in the form of sub-model mapping, which defines the part choices

(e.g., types of storage) for particular options. A person having ordinary skill in the art would have understood that this necessarily would include defining compatibility relationships between the part choices to ensure that the configured product is buildable.

118. Accordingly, it is my opinion that Loomans in view of Stahl discloses that *each configuration sub-model collectively models the configurable product and each configuration sub-model includes data to define compatibility relationships between parts included in the configuration sub-model.*

... [17.7] and each configuration sub-model (i) represents a portion of a configuration model of the configurable product and (ii) allows answers from each configuration sub-model to be combined to provide a consolidated answer to the one or more configuration queries,

119. Loomans discloses a top-level entity that is represented by a parent model, which is then partitioned into sub-level entities that are represented as child sub-models and include the components of the top-level entity.

The invention provides techniques to configure complicated entities using sub-configuration, which **effectively partitions the configuration of a complicated top-level entity into a set of configurations of smaller sub-level entities (i.e., components of the top-level entity)** in combination with a configuration of a "simplified" top-level entity. The top-level entity may be represented by and

configured via a parent model, and **each sub-configurable component may be represented by and configured via a child sub-model.**

(Ex. 1105 [Loomans] at 1:57-65, emphasis added.)

120. Thus, Loomans discloses dividing the top-level entity, which is represented as the parent model, into sub-level entities that are represented as child sub-models. A person having ordinary skill in the art would have understood that the partitioned sub-models each represent a portion of the parent model for the top-level entity (i.e., configurable product).

121. Loomans further discloses that information from each sub-model is made available to the parent model so that the parent model can be validated and configured in view of any new configurations made at the sub-level – i.e., the sub-model level.

Once a sub-configurable option has been configured and validated, the parent model may be returned to, **and any information from the sub-model that may be needed by the parent model is made available to the parent.** Upon a return from the sub-level, or whenever directed, the parent model can be run (i.e., executed) in the context of all options that have been selected or configured. This allows the parent model to be validated with any new configuration made at the sub-level.

(Ex. 1105 [Loomans] at 5:10-18, emphasis added.)

122. For example, Loomans discloses validating the parent model based on the configuration and validation of sub-features using sub-models

In one implementation for sub-configuration, which uses mapped features, parameter values needed for configuration and validation at the parent model and child sub-models are passed between these two levels. Each sub-model is provided with sufficient information needed to configure and validate the option represented by that sub-model. Some of the required information may be obtained at the parent model and passed to the sub-model as mapped features. Other information may be collected in the sub-model. **Upon returning from the child sub-model to the parent model, values for the mapped features are returned from the child sub-model to the parent model.**

* * *

Sub-configuration thus allows a complicated entity to be configured in smaller portions and incrementally, one component at a time. **As shown in FIG. 5, a particular option may be configured using sub-configuration and validated. The parent model is then validated based on the current set of features, which includes those for the sub-configured option.** Another option may then be configured using sub-configuration and validated. The parent model is then validated based on the new current set of features.

(Ex. 1105 [Loomans] at 6:11-22 and 11:19-27, emphasis added.)

A person having ordinary skill in the art would have understood that based on this teaching Loomans discloses the feature-based sub-models are used for the configuration and validation of the parent model.

123. Stahl discloses combining sub-solutions to each of the sub-problems (i.e., sub queries) to form a final solution to the overall configuration problem.

Thus, we can interpret the different leaf nodes of the query as sub-problems that must be solved to solve the overall problem, i.e., the configuration of the required PC. If we have found suitable sub-solutions, i.e. suitable components, for every part-query, **we have to combine these sub-solutions to a final solution for the overall configuration problem.**

(Ex. 1106 [Stahl] at 5, emphasis added.)

124. Thus, Stahl discloses answering the sub-problems and then combining the answers (i.e., solutions) into a consolidated answer for the overall configuration problem (i.e., the configuration query). As described above, it would have been obvious to a person having ordinary skill in the art to utilize the sub-models in Loomans to answer (i.e., provide the sub-solutions to) the sub-problems (i.e., sub-queries) disclosed in Stahl. A PHOSITA would have recognized that the sub-models, which contain the configuration rules for various features of the product, would be queried to solve the sub-problems and the query. A PHOSITA would have recognized that the models (and sub-models) contain all of the required

information to configure the final product.

125. Thus, a person having ordinary skill in the art would have understood that the sub-models from Loomans (i.e., sub-solutions) could be combined to form the final solution to the configuration problem (i.e., configuration query) described in Stahl.

126. Accordingly, it is my opinion that Loomans in view of Stahl discloses that *each configuration sub-model (i) represents a portion of a configuration model of the configurable product and (ii) allows answers from each configuration sub-model to be combined to provide a consolidated answer to the one or more configuration queries.*

... [17.8] generating a response to the one or more configuration queries based upon the processing of each sub-query using at least one configuration sub-model per sub-query; and

127. Loomans discloses a top-level entity that is represented by a parent model, which is then partitioned into sub-level entities that are represented as child sub-models and include the components of the top-level entity.

The invention provides techniques to configure complicated entities using sub-configuration, which **effectively partitions the configuration of a complicated top-level entity into a set of configurations of smaller sub-level entities (i.e., components of the top-level entity)** in combination with a configuration of a "simplified" top-level entity. The top-level entity may be represented by and

configured via a parent model, and **each sub-configurable component may be represented by and configured via a child sub-model.**

(Ex. 1105 [Loomans] at 1:57-65, emphasis added.)

128. Thus, Loomans discloses dividing the top-level entity, which is represented as the parent model, into sub-level entities that are represented as child sub-models. A person having ordinary skill in the art would have understood that the partitioned sub-models each represent a portion of the parent model for the top-level entity (i.e., configurable product).

129. Loomans next discloses configuring an entire entity using a parent model and configuring each sub-configurable component with a sub-model.

A specific embodiment of the invention provides a method for performing sub-configuration of components of an entity. In the method, **the entity is configured via a parent model and each sub-configurable component is configured via one of a number of sub-models. Initially a selection to configure a particular sub-configurable component of the entity is received, and a sub-model for the selected component is identified.** One or more values for one or more features of the selected component are received (e.g., from the parent model or via the sub-model) and form a configuration for the component, which is then validated based on the associated sub-model and the received values. Configuration of the entity is also validated based on the parent model and the validated configuration for the selected component.

(Ex. 1105 [Loomans] at 2:25-38, emphasis added.)

Loomans further discloses that information from each sub-model is made available to the parent model so that the parent model can be validated and configured in view of any new configurations made at the sub-level – i.e., the sub-model level.

Once a sub-configurable option has been configured and validated, the parent model may be returned to, **and any information from the sub-model that may be needed by the parent model is made available to the parent.** Upon a return from the sub-level, or whenever directed, the parent model can be run (i.e., executed) in the context of all options that have been selected or configured. This allows the parent model to be validated with any new configuration made at the sub-level.

(Ex. 1105 [Loomans] at 5:10-18, emphasis added.)

130. In one implementation, Loomans discloses configuring and validating features at the sub-model level and then returning the configuration values back to the parent model.

In one implementation for sub-configuration, which uses mapped features, parameter values needed for configuration and validation at the parent model and child sub-models are passed between these two levels. Each sub-model is provided with sufficient information needed to configure and validate the option represented by that sub-model. Some of the required information may be obtained at the parent model and passed to the sub-model as mapped features. Other information may be collected in the sub-model. **Upon returning from the child**

sub-model to the parent model, values for the mapped features are returned from the child sub-model to the parent model.

* * *

Sub-configuration thus allows a complicated entity to be configured in smaller portions and incrementally, one component at a time. **As shown in FIG. 5, a particular option may be configured using sub-configuration and validated. The parent model is then validated based on the current set of features, which includes those for the sub-configured option.** Another option may then be configured using sub-configuration and validated. The parent model is then validated based on the new current set of features.

(Ex. 1105 [Loomans] at 6:11-22 and 11:19-27, emphasis added.)

131. Stahl discloses interpreting a query as a series of “sub-problems,” which can be solved with “sub-solutions,”

Thus, **we can interpret the different leaf nodes of the query as sub-problems** that must be solved to solve the overall problem, i.e., the configuration of the required PC. **If we have found suitable sub-solutions, i.e. suitable components, for every part-query,** we have to combine these sub-solutions to a final solution for the overall configuration problem.

(Ex. 1106 [Stahl] at 5, emphasis added.)

132. A person having ordinary skill in the art would have understood that models and sub-models like those described in Loomans could have been used to process each sub-problem described in Stahl. Moreover, a person having ordinary

skill in the art would have recognized that the models, which contain the configuration rules for the product, would be queried to solve the sub-problems and the ultimate query. Indeed, a PHOSITA would recognize that the models (and sub-models) contain all of the required information to configure the final product. (*Id.*) As Stahl discloses, the processed sub-solutions could then be combined to provide a final solution for the overall configuration problem (i.e., a response to the configuration query). (*Id.*)

133. Thus, a person having ordinary skill in the art would have understood that the sub-models from Loomans (i.e., sub-solutions) could be combined to form the final solution to the configuration problem (i.e., configuration query) described in Stahl.

134. Accordingly, it is my opinion that Loomans in view of Stahl discloses *generating a response to the one or more configuration queries based upon the processing of each sub-query using at least one configuration sub-model per sub-query.*

... [17.9] providing the response to the one or more configuration queries as data for display by a display device.

135. Loomans discloses a user interface that is used to provide information to an administrator and/or user of the configuration system:

In an embodiment, configuration system 800 includes a number of modules such as a **user interface module 812** and a configuration engine 814. Additional, fewer, and/or different modules may also be included in configuration system 800, and this is within the scope of the invention. **User interface module 812 provides the interface (e.g., screens such as those shown in FIGS. 7A through 7C) used to present information to an administrator and/or a user of the configuration system.** User interface module 812 further receives and interprets user commands and data, which may be provided via mouse clicks, mouse movements, keyboard inputs, and other means. **User interface module 812 then provides the received data and commands to other modules, which then perform the appropriate responsive action.**

(Ex. 1105 [Loomans] at 16:36-49, emphasis added.)

136. Loomans discloses an example of the user interface in Figure 7C: **FIG. 7C shows portion of a screen 790 that may be displayed to show the selections for various options.** In this example screen, the selected feature values for each sub-configurable option and each selectable option are shown on the screen.

(Ex. 1105 [Loomans] at 16:18-22, emphasis added.)

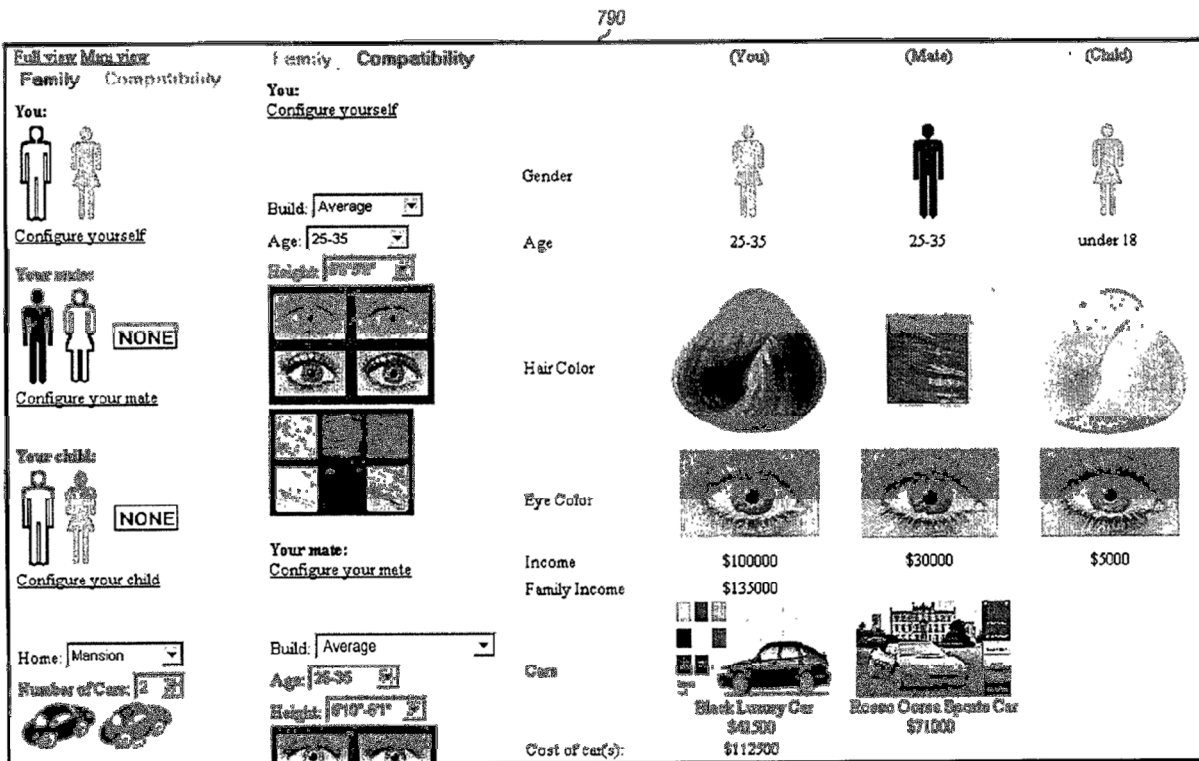


FIG. 7C

(Id. at Fig. 7C)

137. Likewise, Stahl discloses a using a display to implementing a demo of its disclosed configuration methods over the world wide web.

To evaluate the functionality of our configuration approach we have implemented a generic prototype for the described configuration process. This prototype consists of an extension of the commercial CBR tool CBR-Works which has been developed jointly by the University of Kaiserslautern and Tecinno GmbH. **To be accessible over the World Wide Web the prototype also provides two respective interfaces for the demand acquisition (see Fig. 5) and the result presentation.**

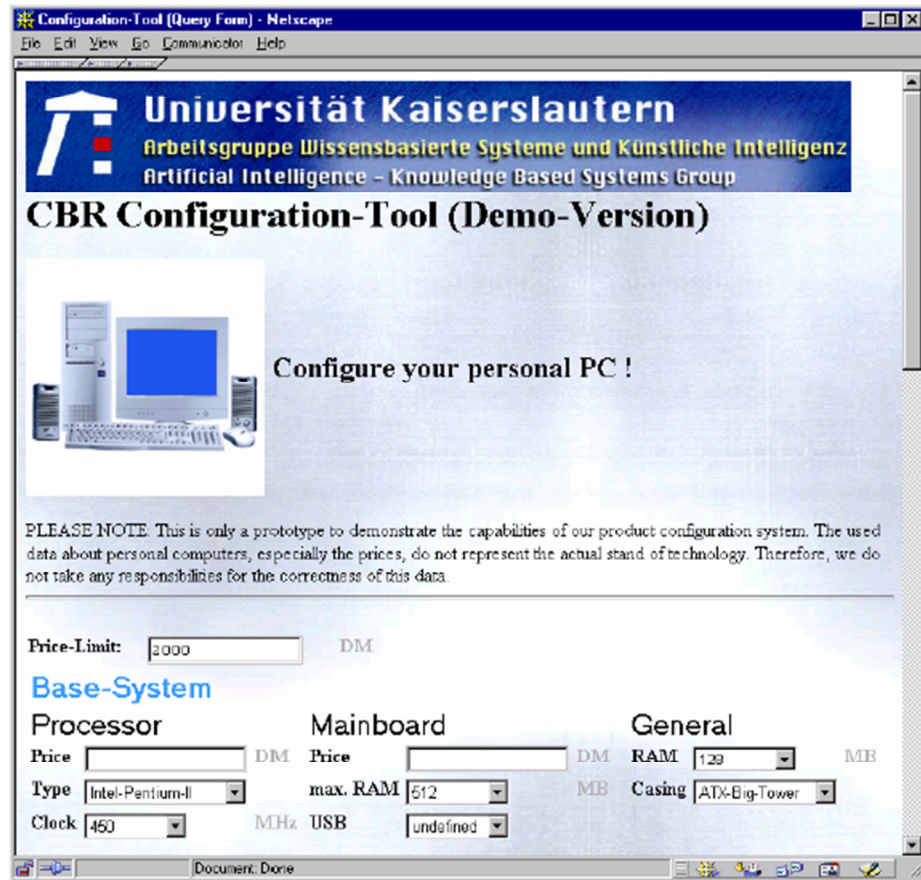


Fig. 5. Implemented demo version.

(Ex. 1106 [Stahl] at 8, emphasis added.)

138. Accordingly, it is my opinion that Loomans in view of Stahl discloses *providing the response to the one or more configuration queries as data for display by a display device.*

b. Obvious to Combine Loomans with Stahl

139. A person having ordinary skill in the art would have found it obvious to combine the teachings from Loomans with the teachings from Stahl. Both Loomans and Stahl describe configuration systems that are based on the concept of eliminating configuration complexity by dividing a configuration problem into

simpler sub-problems. Indeed, Loomans expressly describes partitioning a top-entity parent model into sub-configurable sub-models so that the sub-models can be configured and validated. (Ex. 1105 [Loomans] at 1:57-65.)

140. Similarly, Stahl discloses dividing configuration queries into sub-problems, which can then be solved via sub-solutions, and the sub-solutions can then be combined to create a final answer to the query. (Ex. 1106 [Stahl] at 4-5.) Thus, both Loomans and Stahl describe using decomposition to evaluate rules in a configuration system. Both Loomans and Stahl use the configuration of a personal computer as an example. At the time of the alleged invention, a software engineer interested in building a high-performance configuration system, e.g., one that can respond in real-time to customers trying to order products on a web site, would have had good reason to draw guidance from the configuration approaches described in Loomans as well as Stahl. A configuration system with the rule-evaluation mechanism of Loomans (checking a configuration table or a sub-configuration table) would likely run faster. A configuration with the rule-evaluation mechanism of Stahl would work better in situations where the vendor did not expect to be able to find an exact match for a customer's requirements.

141. Although both Loomans and Stahl are addressing a common problem in the prior art, the emphasis of Loomans and Stahl are somewhat different. Loomans provides more details about how to ensure that answers to configuration

queries are processed in a roughly constant amount of time. This is based on the fact that any set of values from a customer-specified configuration can be looked up in the configuration and sub-configuration tables of Loomans in a roughly constant amount of time. (Ex. 1105 [Loomans] at 4:26-29, 11:39-46,). Stahl provides more details about how to deal with situations in which the customer's requirements are over-specified to the point that no configuration can satisfy every requirement. ("the most suitable component that is available within the component case-base," (Ex. 1106 [Stahl] at 6); "If it is impossible to determine a weak part whose adaption could perhaps improve the product, the complete configuration process is succeeded and the final product can be presented to the customer," (*Id.* at 7), making it clear that the system produces the best result that it can). Taken together, these references comfortably render obvious the subject matter of the challenged claims. Thus, to the extent that Stahl does not expressly disclose dividing configuration models into sub-models to process sub-problems, or equivalents thereof, it would have been obvious to a person having ordinary skill in the art to use the rule database system described in Loomans with the rule evaluation system described in Stahl.

142. First, using the case-based reasoning of Stahl, at least for some of the sub-models, would improve the functionality of the configuration system described in Loomans. As noted above, Loomans gives quick and certain answers but, unless

an invalid configuration has been expected by the developers and placed into an “exception table,” it is difficult to give the user a full explanation as to why a configuration is invalid. (Ex. 1105 [Loomans] at 11:39-46.) The case-based reasoning of Stahl would help in situations where Loomans was applied to products in which it was unlikely that all of a customer’s preferences could be satisfied simultaneously, e.g., the customer wants a car with sport functionality that also seats 8.

143. Second, for similar reasons, a person having ordinary skill in the art would have concluded that it was obvious to try using the rule evaluation system of Stahl with at least some of the sub-models of Loomans. During the rule-preparation process of Loomans it might have been discovered that the exception table was growing to an extremely large size. Recall that Loomans uses a black-and-white system for evaluating configurations. If a configuration is found in a configuration table it is valid; if not found in the configuration table it is not valid. (Ex. 1105 [Loomans] at 4:26-29, 11:39-46.). A customer whose requested configuration comes back as “invalid” cannot get any explanation of the problem unless the requested configuration was anticipated by the programmers and maintainers of the system and recorded in the exception table. Customers calling up to ask “Why can’t I place an order?” would have motivated a person of ordinary skill to use the more flexible rule evaluation system of Stahl at least in whichever

sub-model was proving difficult to satisfy.

144. Further, because the sub-models in Loomans are designed for use in configuring sub-configurable options, and Stahl teaches that the described sub-problems represent configuration problems for parts (i.e., sub-configurable options), using the rule evaluation system of Stahl to evaluate one or more of the sub-models taught in Loomans would have yielded a predictable solution to configuring the overall product taught in Loomans with increased efficiency than previous configuration methods. A person of ordinary skill in the art would have known that modifying the teachings of Loomans configuration to use Stahl's method of rule evaluation on at least some sub-models would have been a simple software modification, largely accomplished by installing the CBR-Works package; and a person having ordinary skill in the art would have been capable and knowledgeable to make such a software change.

c. Claim 30

... [30.0] A computer system to implement an inference procedure for responding to one or more configuration queries using configuration sub-models, the system comprising:

145. For the same reasons discussed above for claim 17, it is my opinion that a person having ordinary skill in the art would have concluded that it was obvious to combine the teachings of Loomans with the teachings of Stahl. For

brevity, I incorporate my opinions and analysis regarding that combinability of Loomans and Stahl above for claims 17 for claim 30.

146. Loomans describes a computer system, which is capable of implementing the disclosed configuration technology.

FIG. 8 is a simplified diagram of an embodiment of a configuration system 800 that may be capable of implementing various aspects and embodiments of the invention. In this embodiment, configuration system 800 is implemented on a set of one or more host servers 810 that couple to and interact with one or more client computers 830 via direct connections, a computer network (e.g., the Internet), and/or some other means. Host server(s) 810 further couples to a database server 820 that stores data (typically in a "raw" form) used by the system.

(Ex. 1105 [Loomans] at 16:26-35, emphasis added.)

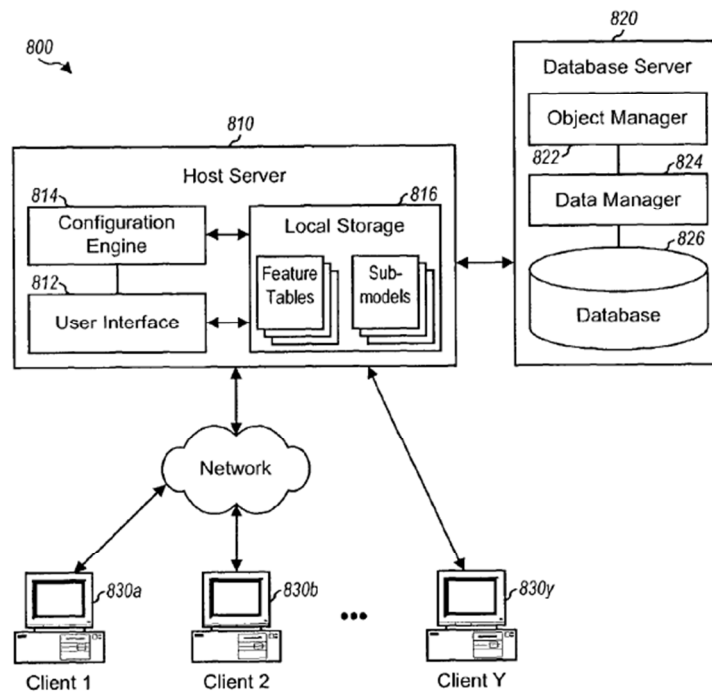


FIG. 8

(Ex. 1105 [Loomans] at Fig. 8.)

147. Loomans further describes an example in Figure 9 of a computer used as host computer (810) or the client computers from Figure 8.

FIG. 9 is a block diagram of an embodiment of a computer system 900 that may be used to implement host server 810 or client computers 820. System 900 includes a bus 908 that interconnects major subsystems such as one or more processors 910, a memory subsystem 912, a data storage subsystem 914, an input device interface 916, an output device interface 918, and a network interface 920. Processor(s) 910 perform many of the processing functions for system 900 and communicate with a number of peripheral devices via bus 908.

(Ex. 1105 [Loomans] at 17:42-50, emphasis added.)

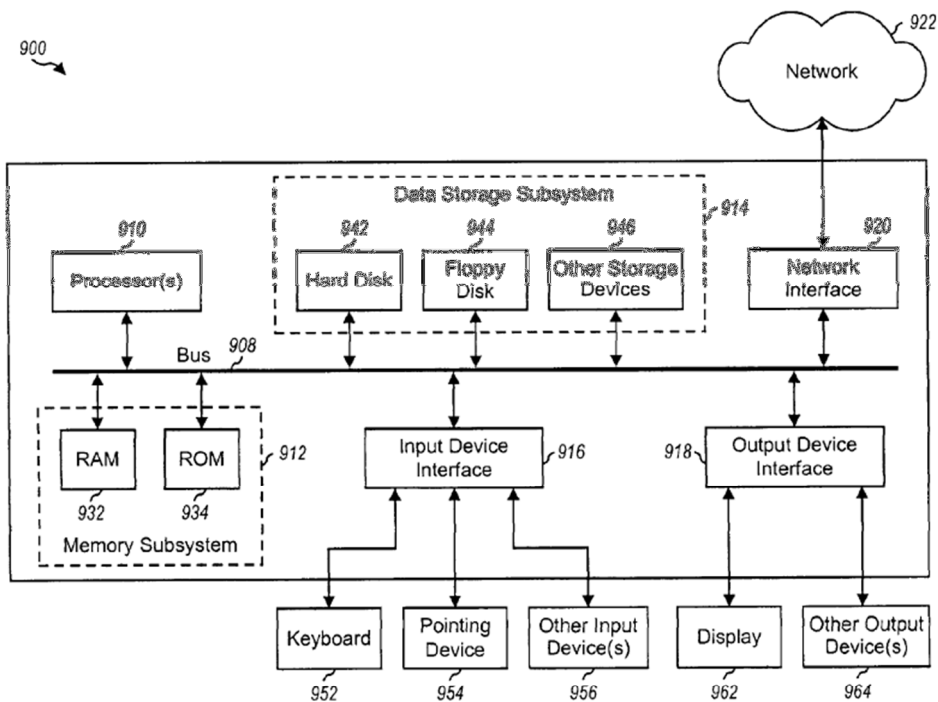


FIG. 9

(Ex. 1105 [Loomans] at Fig. 9.)

148. Likewise, Stahl discloses implementing a demo of its disclosed configuration methods over the world wide web.

To evaluate the functionality of our configuration approach we have implemented a generic prototype for the described configuration process. This prototype consists of an extension of the commercial CBR tool CBR-Works which has been developed jointly by the University of Kaiserslautern and Tecinno GmbH. **To be accessible over the World Wide Web the prototype also provides two respective interfaces for the demand acquisition (see Fig. 5) and the result presentation.**

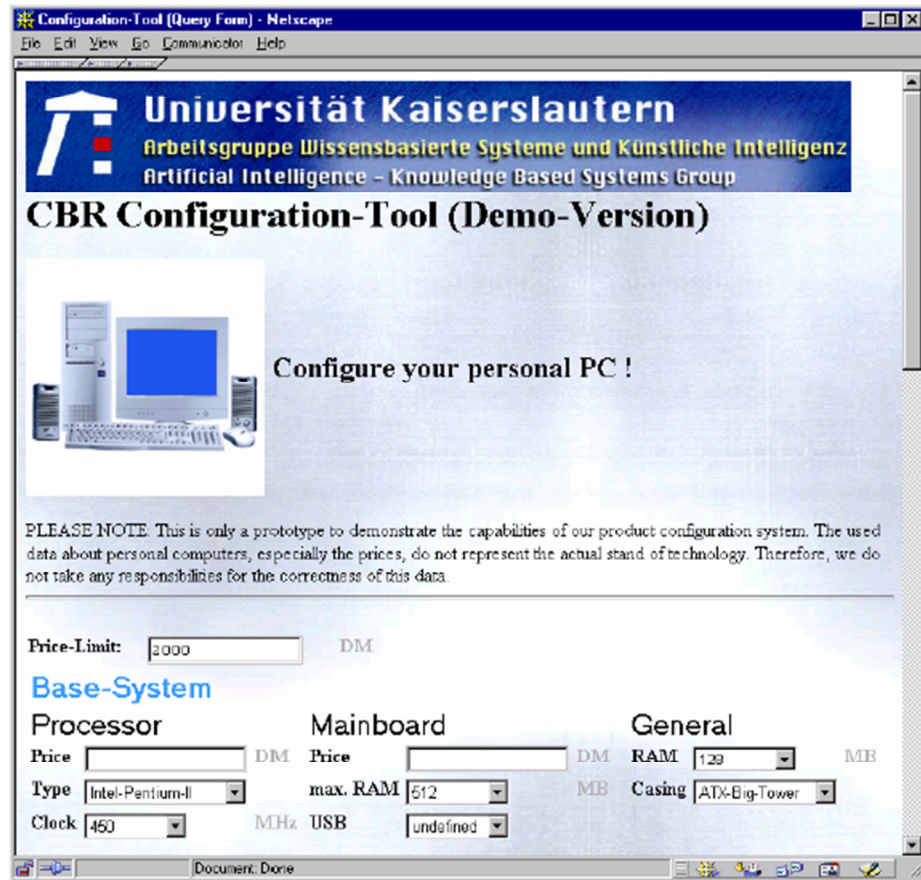


Fig. 5. Implemented demo version.

(Ex. 1106 [Stahl] at Fig. 5.)

A person having ordinary skill in the art would have understood that the demo would need to be implemented on a computer system to be functional over the World Wide Web.

149. Loomans further discloses configuring a parent configuration model by configuring and validating child sub-models created by portioning the parent configuration model.

Techniques to performing sub-configuration of components of an entity. **In one method, the entity is configured via a parent model**

and each sub-configurable component is configured via one of a number of sub-models. Initially a selection to configure a particular sub-configurable component of the entity is received, and a sub-model for the selected component is identified. One or more values for one or more features of the selected component are received (e.g., from the parent model or via the sub-model) and form a configuration for the component, **which is then validated based on the associated sub-model and the received values.** Configuration of the entity is also validated based on the parent model and the validated configuration for the selected component. Feedbacks may be provided for each configuration of the parent model and sub-models. The data for the parent model and sub-models may be localized or globalized.

(Ex. 1105 [Loomans] at Abstract, emphasis added.)

The invention provides techniques to configure complicated entities using sub-configuration, which effectively partitions the configuration of a complicated top-level entity into a set of configurations of smaller sub-level entities (i.e., components of the top-level entity) in combination with a configuration of a "simplified" top-level entity. The top-level entity may be represented by and configured via a parent model, and each sub-configurable component may be represented by and configured via a child sub-model.

(Ex. 1105 [Loomans] at 1:57-65, emphasis added.)

150. Loomans further discloses that one or more configuration queries are responded to via the disclosed sub-models.

With sub-configuration, a complicated entity may be more efficiently modeled, configured, and validated. Each sub-configurable option of the entity may be configured and validated via a respective child sub-model. Generally, sufficient information is made available to the child sub-model such that the associated option can be validly configured. **The required information may be provided from the parent level, queried and entered at the sub-level via the child sub-model,** and/or provided via some other mechanisms.

(Ex. 1105 [Loomans] at 4:59-67, emphasis added.)

151. Stahl discloses partitioning a query into sub-problems (i.e., sub-queries), which are then solved and combined back together to form the overall solution of the configuration problem (i.e., configuration query)

Query. The starting point of the configuration process is the query represented by an incomplete instantiation of the compositional structure. When looking at the example query shown in Fig. 3, **we can interpret the root node as our actual problem, i.e., we are searching a PC with a set of special properties.** To reach this goal it is necessary to select appropriate components that fulfill the properties of the respective part-queries. In our example, one part-query states that the PC should have a hard-disk with 12GB of capacity. To fulfill this demand we can, e.g., integrate the concrete hard-disk "Maxtor91303D6" for the hard-disk part in the PC. **Thus, we can interpret the different leaf nodes of the query as sub-problems that must be solved to solve the overall problem, i.e., the**

configuration of the required PC. If we have found suitable sub-solutions, i.e. suitable components, for every part-query, **we have to combine these sub-solutions to a final solution for the overall configuration problem.**

(Ex. 1106 [Stahl] at 4-5, emphasis added.)

152. A person having ordinary skill in the art would have understood that the sub-problems disclosed in Stahl represent a form of sub-query, which Stahl discloses are solved with “suitable sub-solutions” (i.e., answers) by determining the suitable parts for a particular “part-query”. (Ex. 1106 [Stahl] at 5.) A person having ordinary skill in the art would have further understood that the sub-models described in Loomans could be partitioned in such a way to provide the solutions (i.e., answers) to a particular part-query raised in a sub-problem disclosed in Stahl. A PHOSITA would have further appreciated that the models and sub-models described in Loomans could have been used to process solutions and sub-solutions to the Query and sub-problems in Stahl. Indeed, Loomans discloses that “each sub-configurable component is configured via one of a number of sub-models.” (Ex. 1105 [Loomans] at Abstract.) Thus, a person having ordinary skill in the art would have understood that the sub-problems described in Stahl could be answered (i.e., solved) using the sub-models described in Loomans. Such models (and the rules contained therein) are commonly used during the resolving stage of a configuration problem in a configuration system.

153. Finally, Stahl discloses “[i]f we have found suitable sub-solutions, i.e. suitable components, for every part-query, **we have to combine these sub-solutions to a final solution for the overall configuration problem.**” (Ex. 1106 [Stahl] at 5, emphasis added.) Thus, Stahl discloses recombining the sub-solutions to the sub-problems to provide a response to the configuration problem in the form of a combined solution.

154. Accordingly, it is my opinion that Loomans in view of Stahl discloses *a computer system to implement an inference procedure for responding to one or more configuration queries using configuration sub-models, the system comprising.*

... [30.1] a processor; and a storage medium having data encoded therein, the data comprising processor executable code for:

155. Loomans describes a system, which is capable of implementing the disclosed configuration technology: “FIG. 8 is a simplified diagram of an embodiment of a configuration system 800 that may be capable of implementing various aspects and embodiments of the invention.” (Ex. 1105 at 16:26-28.) Loomans discloses an example in Figure 9 of a computer used to implement the described configuration process. As bolded below, the exemplar computer includes a processor (depicted in yellow in Figure 9 below), a data storage subsystem (depicted in blue in Figure 9 below), and a memory subsystem (depicted in orange

in Figure 9 below).

FIG. 9 is a block diagram of an embodiment of a computer system 900 that may be used to implement host server 810 or client computers 820. System 900 includes a bus 908 that interconnects major subsystems such as **one or more processors 910, a memory subsystem 912, a data storage subsystem 914**, an input device interface 916, an output device interface 918, and a network interface 920. Processor(s) 910 perform many of the processing functions for system 900 and communicate with a number of peripheral devices via bus 908.

(Ex. 1105 [Loomans] at 17:42-50, emphasis added.)

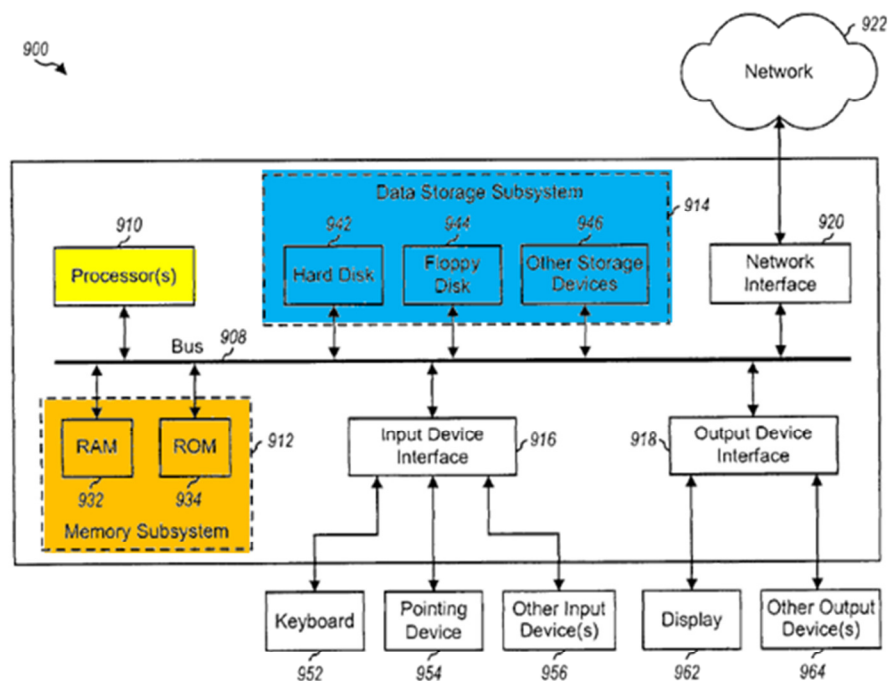


FIG. 9

(Ex. 1105 [Loomans] at Fig. 9.)

Loomans further discloses “[a] computer program product for performing sub-

configuration of components of an entity, comprising a computer-usable storage medium having embodied therein computer-readable program codes executable by a **processor**. (Ex. 1105 at 18:35-39, emphasis added..) Thus, Loomans discloses a general purpose microprocessor for performing sub-configuration.

156. Loomans further discloses that memory subsystem “may include a RAM 932 and a ROM 934 used to store codes and data that implement various aspects of the invention.” (Ex. 1105 [Loomans] at 17:51-53.) Loomans also discloses that the data storage subsystem “provides non-volatile storage for program codes and data” (*Id.* at 17:56-57.)

157. A person of ordinary skill in the art would recognize that configuration methods disclosed in Loomans and Stahl would be implemented using one or more computer processors like those disclosed in Loomans. Such processor use ‘executable code’ to provide instructions to the processor and the ‘storage medium’ to store and execute instructions for the configuration system.

158. Accordingly, a person having ordinary skill in the art would have understood that Loomans in view of Stahl discloses *a processor and a storage medium having data encoded therein* for performing steps [30.2]-[30.9] (below).

... [30.2] dividing a consolidated configuration model into multiple configuration sub-models;

159. This limitation is identical to claim element [17.2]. Accordingly, this

limitation is met by the prior art as described in my analysis of claim element [17.2] (above). Thus, I incorporate my analysis for element [17.2] for this claim element.

160. Accordingly, a person having ordinary skill in the art would have understood that Loomans in view of Stahl discloses *dividing a consolidated configuration model into multiple configuration sub-models*.

... [30.3] responding to the one or more configuration queries representing questions involving configuration of a configurable product, wherein responding to the one or more configuration queries comprises:

161. This limitation is identical to claim element [17.3]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [17.3] (above). Thus, I incorporate my analysis for element [17.3] for this claim element.

162. Accordingly, it is my opinion that Loomans in view of Stahl discloses *responding to the one or more configuration queries representing questions involving configuration of a configurable product*.

... [30.4] dividing one or more configuration queries into multiple configuration sub-queries, wherein the multiple configuration sub-queries represent the one or more configuration queries;

163. This limitation is identical to claim element [17.4]. Accordingly, this

limitation is met by the prior art as described in my analysis of claim element [17.4] (above). Thus, I incorporate my analysis for element [17.4] for this claim element.

164. Accordingly, it is my opinion that Loomans in view of Stahl discloses *dividing one or more configuration queries into multiple configuration sub-queries, wherein the multiple configuration sub-queries represent the one or more configuration queries.*

... [30.5] processing each sub-query using at least one configuration sub-model per sub-query,

165. This limitation is identical to claim element [17.5]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [17.5] (above). Thus, I incorporate my analysis for element [17.5] for this claim element.

166. Accordingly, it is my opinion that Loomans in view of Stahl discloses *processing each sub-query using at least one configuration sub-model per sub-query.*

... [30.6] wherein each configuration sub-model collectively models the configurable product and each configuration sub-model includes data to define compatibility relationships between parts included in the configuration sub-model,

167. This limitation is identical to claim element [17.6]. Accordingly, this

limitation is met by the prior art as described in my analysis of claim element [17.6] (above). Thus, I incorporate my analysis for element [17.6] for this claim element.

168. Accordingly, it is my opinion that Loomans in view of Stahl discloses that *each configuration sub-model collectively models the configurable product and each configuration sub-model includes data to define compatibility relationships between parts included in the configuration sub-model.*

... [30.7] and each configuration sub-model (i) represents a portion of a configuration model of the configurable product and (ii) allows answers from each configuration sub-model to be combined to provide a consolidated answer to the one or more configuration queries,

169. This limitation is identical to claim element [17.7]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [17.7] (above). Thus, I incorporate my analysis for element [17.7] for this claim element.

170. Accordingly, it is my opinion that Loomans in view of Stahl discloses that *each configuration sub-model (i) represents a portion of a configuration model of the configurable product and (ii) allows answers from each configuration sub-model to be combined to provide a consolidated answer to the one or more configuration queries.*

... [30.8] generating a response to the one or more configuration queries based upon the processing of each sub-query using at least one configuration sub-model per sub-query; and

171. This limitation is identical to claim element [17.8]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [17.8] (above). Thus, I incorporate my analysis for element [17.8] for this claim element.

172. Accordingly, it is my opinion that Loomans in view of Stahl discloses *generating a response to the one or more configuration queries based upon the processing of each sub-query using at least one configuration sub-model per sub-query.*

... [30.9] providing the response to the one or more configuration queries as data for display by a display device.

173. This limitation is identical to claim element [17.9]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [17.9] (above). Thus, I incorporate my analysis for element [17.9] for this claim element.

174. Accordingly, it is my opinion that Loomans in view of Stahl discloses *providing the response to the one or more configuration queries as data for display by a display device.*

d. Claim 44

... [44.0] A computer storage medium comprising data embedded therein to cause a computer system to respond to one or more configuration queries using configuration sub-models, wherein the data comprises code for:

175. For the same reasons discussed above for claim 17, it is my opinion that a person having ordinary skill in the art would have concluded that it was obvious to combine the teachings of Loomans with the teachings of Stahl. For brevity, I incorporate my opinions and analysis regarding that combinability of Loomans and Stahl above for claims 17 for claim 44.

176. Loomans discloses an example in Figure 9 of a computer used to implement the described configuration process. As bolded below, the exemplar computer includes a processor (depicted in yellow in Figure 9 below), a data storage subsystem (depicted in blue in Figure 9 below), and a memory subsystem (depicted in orange in Figure 9 below).

FIG. 9 is a block diagram of an embodiment of a computer system 900 that may be used to implement host server 810 or client computers 820. System 900 includes a bus 908 that interconnects major subsystems such as **one or more processors 910, a memory subsystem 912, a data storage subsystem 914**, an input device interface 916, an output device interface 918, and a network interface 920. Processor(s) 910 perform many of the processing functions for

system 900 and communicate with a number of peripheral devices via bus 908.

(Ex. 1105 [Loomans] at 17:42-50, emphasis added.)

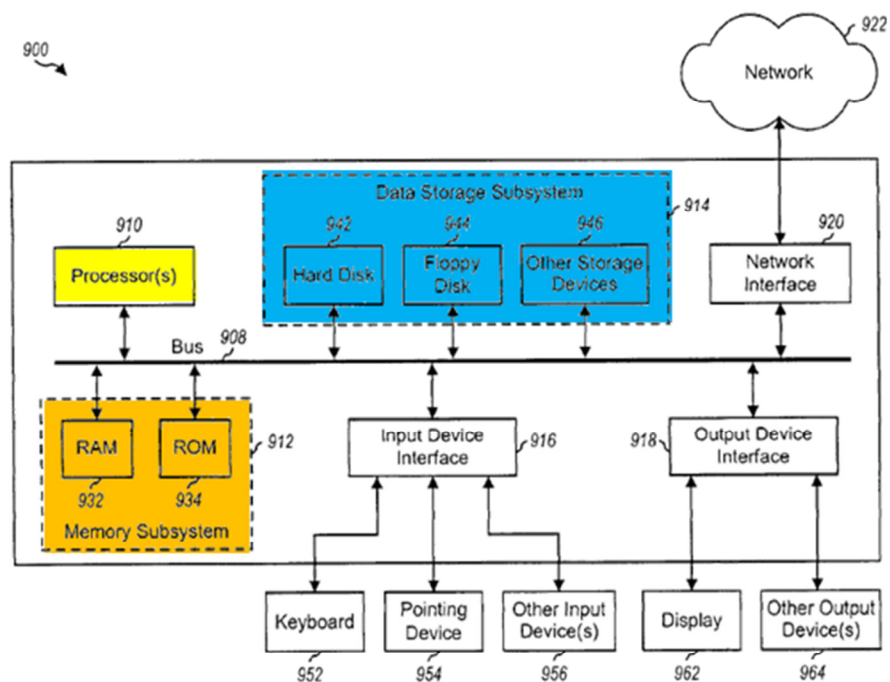


FIG. 9

(Ex. 1105 [Loomans] at Fig. 9.)

177. Loomans further discloses that memory subsystem “may include a RAM 932 and a ROM 934 used to store codes and data that implement various aspects of the invention.” (Ex. 1105 [Loomans] at 17:51-53.) Loomans also discloses that the data storage subsystem “provides non-volatile storage for program codes and data” (*Id.* at 17:56-57.)

178. Accordingly, it is my opinion that Loomans in view of Stahl discloses *a computer storage medium comprising data embedded therein to cause a*

computer system to respond to one or more configuration queries using configuration sub-models.

... [44.1] dividing a consolidated configuration model into multiple configuration sub-models;

179. This limitation is identical to claim element [17.2]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [17.2] (above). Thus, I incorporate my analysis for element [17.2] for this claim element.

180. Accordingly, a person having ordinary skill in the art would have understood that Loomans in view of Stahl discloses *dividing a consolidated configuration model into multiple configuration sub-models.*

... [44.2] responding to the one or more configuration queries representing questions involving configuration of a configurable product, wherein responding to the one or more configuration queries comprises:

181. This limitation is identical to claim element [17.3]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [17.3] (above). Thus, I incorporate my analysis for element [17.3] for this claim element.

182. Accordingly, it is my opinion that Loomans in view of Stahl discloses *responding to the one or more configuration queries representing questions*

involving configuration of a configurable product.

... [44.3] dividing one or more configuration queries into multiple configuration sub-queries, wherein the multiple configuration sub-queries represent the one or more configuration queries;

183. This limitation is identical to claim element [17.4]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [17.4] (above). Thus, I incorporate my analysis for element [17.4] for this claim element.

184. Accordingly, it is my opinion that Loomans in view of Stahl discloses *dividing one or more configuration queries into multiple configuration sub-queries, wherein the multiple configuration sub-queries represent the one or more configuration queries.*

... [44.4] processing each sub-query using at least one configuration sub-model per sub-query,

185. This limitation is identical to claim element [17.5]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [17.5] (above). Thus, I incorporate my analysis for element [17.5] for this claim element.

186. Accordingly, it is my opinion that Loomans in view of Stahl discloses *processing each sub-query using at least one configuration sub-model per sub-*

query.

... [44.5] wherein each configuration sub-model collectively models the configurable product and each configuration sub-model includes data to define compatibility relationships between parts included in the configuration sub-model,

187. This limitation is identical to claim element [17.6]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [17.6] (above). Thus, I incorporate my analysis for element [17.6] for this claim element.

188. Accordingly, it is my opinion that Loomans in view of Stahl discloses that *each configuration sub-model collectively models the configurable product and each configuration sub-model includes data to define compatibility relationships between parts included in the configuration sub-model.*

... [44.6] generating a response to the one or more configuration queries based upon the processing of each sub-query using at least one configuration sub-model per sub-query

189. This limitation is identical to claim element [17.8]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [17.8] (above). Thus, I incorporate my analysis for element [17.8] for this claim element.

190. Accordingly, it is my opinion that Loomans in view of Stahl discloses

generating a response to the one or more configuration queries based upon the processing of each sub-query using at least one configuration sub-model per sub-query.

... [44.7] and each configuration sub-model (i) represents a portion of a configuration model of the configurable product and (ii) allows answers from each configuration sub-model to be combined to provide a consolidated answer to the one or more configuration queries; and

191. This limitation is identical to claim element [17.7]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [17.7] (above). Thus, I incorporate my analysis for element [17.7] for this claim element.

192. Accordingly, it is my opinion that Loomans in view of Stahl discloses that *each configuration sub-model (i) represents a portion of a configuration model of the configurable product and (ii) allows answers from each configuration sub-model to be combined to provide a consolidated answer to the one or more configuration queries.*

... [44.8] providing the response to the one or more configuration queries as data for display by a display device.

193. This limitation is identical to claim element [17.9]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element

[17.9] (above). Thus, I incorporate my analysis for element [17.9] for this claim element.

194. Accordingly, it is my opinion that Loomans in view of Stahl discloses *providing the response to the one or more configuration queries as data for display by a display device.*

2. Analysis of Claims 45 and 46

a. Claim 45

... [45.0] A computer system to implement an inference procedure for responding to one or more configuration queries using configuration sub-models, the system comprising:

195. For the same reasons discussed above for claim 17, it is my opinion that a person having ordinary skill in the art would have concluded that it was obvious to combine the teachings of Loomans with the teachings of Stahl. For brevity, I incorporate my opinions and analysis regarding that combinability of Loomans and Stahl above for claim 17 for claims 45-46.

196. Loomans discloses configuring a parent configuration model by configuring and validating child sub-models created by partitioning the parent configuration model.

Techniques to performing sub-configuration of components of an entity. **In one method, the entity is configured via a parent model and each sub-configurable component is configured via one of a**

number of sub-models. Initially a selection to configure a particular sub-configurable component of the entity is received, and a sub-model for the selected component is identified. One or more values for one or more features of the selected component are received (e.g., from the parent model or via the sub-model) and form a configuration for the component, **which is then validated based on the associated sub-model and the received values.** Configuration of the entity is also validated based on the parent model and the validated configuration for the selected component. Feedbacks may be provided for each configuration of the parent model and sub-models. The data for the parent model and sub-models may be localized or globalized.

(Ex. 1105 [Loomans] at Abstract, emphasis added.)

The invention provides techniques to configure complicated entities using sub-configuration, which effectively partitions the configuration of a complicated top-level entity into a set of configurations of smaller sub-level entities (i.e., components of the top-level entity) in combination with a configuration of a "simplified" top-level entity. The top-level entity may be represented by and configured via a parent model, and each sub-configurable component may be represented by and configured via a child sub-model.

(Ex. 1105 [Loomans] at 1:57-65, emphasis added.)

197. A person having ordinary skill in the art would have understood that the teachings of configuring a sub-model or validating a sub-model described in Loomans corresponds to the “query” nomenclature of the ‘057 Patent. Indeed, the

processing and validation of the model and sub-models in Loomans would include processing one or more queries associated with those models. Otherwise, those models would be unable to perform any functions, as the query begins the configuration process. At a minimum the system of Loomans discloses processing the query, e.g., “Is this submodel in a valid state?”.

198. Loomans further discloses that one or more configuration queries are responded to via the disclosed sub-models.

With sub-configuration, a complicated entity may be more efficiently modeled, configured, and validated. Each sub-configurable option of the entity may be configured and validated via a respective child sub-model. Generally, sufficient information is made available to the child sub-model such that the associated option can be validly configured. **The required information may be provided from the parent level, queried and entered at the sub-level via the child sub-model,** and/or provided via some other mechanisms.

(Ex. 1105 [Loomans] at 4:59-67, emphasis added.)

199. Loomans also describes a computer system, which is capable of implementing the disclosed configuration technology described above.

FIG. 8 is a simplified diagram of an embodiment of a configuration system 800 that may be capable of implementing various aspects and embodiments of the invention. In this embodiment, configuration system 800 is implemented on a set of one

or more host servers 810 that couple to and interact with one or more client computers 830 via direct connections, a computer network (e.g., the Internet), and/or some other means. Host server(s) 810 further couples to a database server 820 that stores data (typically in a "raw" form) used by the system.

(Ex. 1105 [Loomans] at 16:26-35, emphasis added.)

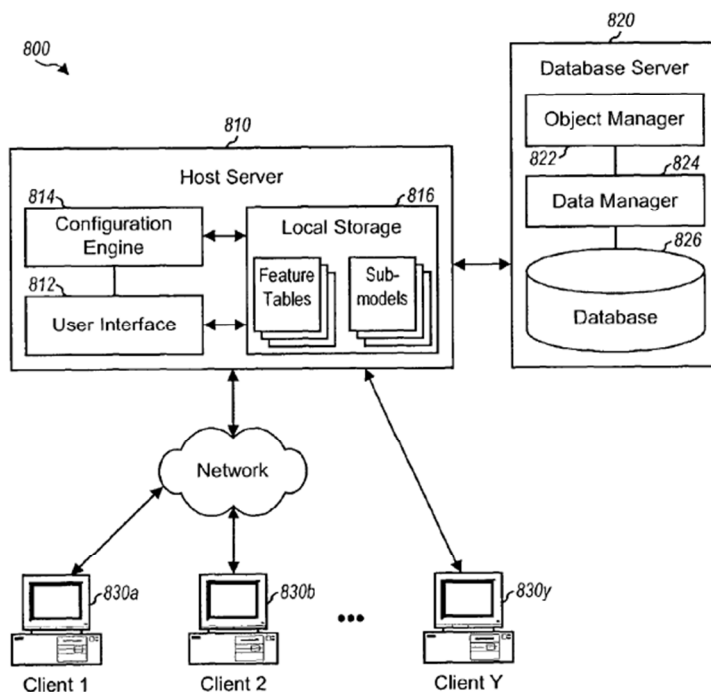


FIG. 8

(Ex. 1105 [Loomans] at Fig. 8.)

200. As annotated below, the exemplar computer includes a processor (depicted in yellow in Figure 9 below), a data storage subsystem (depicted in blue in Figure 9 below), and a memory subsystem (depicted in orange in Figure 9 below).

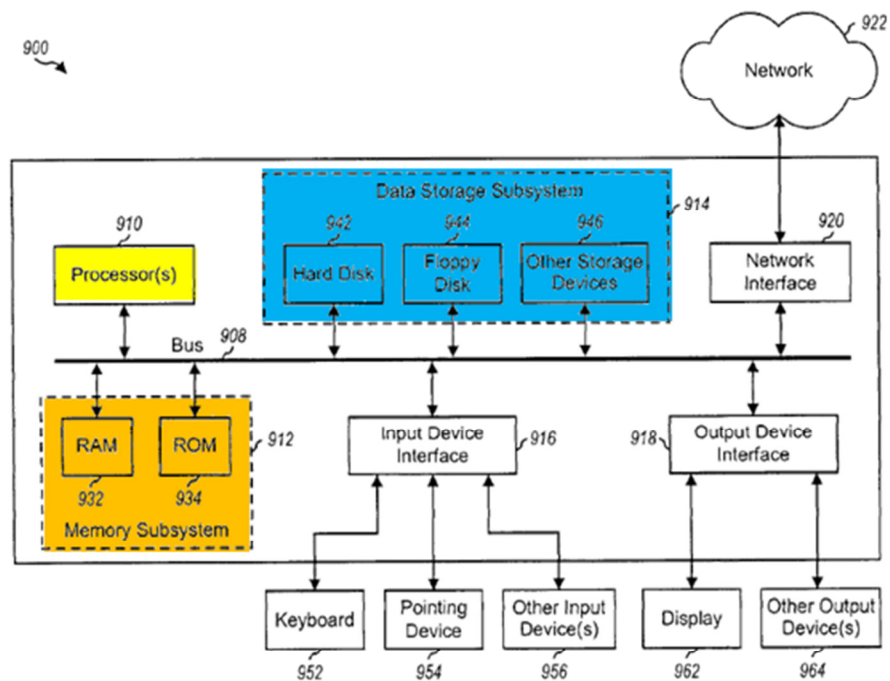


FIG. 9

(Ex. 1105 [Loomans] at Fig. 9 (annotated).)

201. A person having ordinary skill in the art would have appreciated that the teachings in Stahl could be applied to the teachings in Loomans. Stahl discloses dividing a configuration query into sub-problems (i.e., sub-queries), which are then solved and combined back together to form the overall solution of the configuration problem (i.e., the original configuration query)

Query. The starting point of the configuration process is the query represented by an incomplete instantiation of the compositional structure. When looking at the example query shown in Fig. 3, we can interpret the root note as our actual problem, i.e., we are searching a PC with a set of special properties. To reach this goal it is necessary to select appropriate components that fulfill

the properties of the respective part-queries. In our example, one part-query states that the PC should have a hard-disk with 12GB of capacity. To fulfill this demand we can, e.g., integrate the concrete hard-disk "Maxtor91303D6" for the hard-disk part in the PC. **Thus, we can interpret the different leaf nodes of the query as sub-problems that must be solved to solve the overall problem, i.e., the configuration of the required PC.** If we have found suitable sub-solutions, i.e. suitable components, for every part-query, **we have to combine these sub-solutions to a final solution for the overall configuration problem.**

(Ex. 1106 [Stahl] at 4-5, emphasis added.)

202. A person having ordinary skill in the art would have understood that the sub-problems described in Stahl represent a form of sub-query, which Stahl teaches are solved with "suitable sub-solutions" (i.e., answers) by determining the suitable parts for a particular "part-query". (Ex. 1106 [Stahl] at 5.) To the extent Stahl does not expressly disclose models and sub-models, and/or lacks details thereof, for providing solutions and sub-solutions to the Query and sub-problems in Stahl, a person having ordinary skill in the art would have appreciated that models and sub-models like those described in Loomans could have been used to process the solutions and sub-solutions to the query and sub-problems described in Stahl. Such models (and the rules contained therein) are commonly used during the resolving stage in a configuration system. .) Finally, the '057 Patent identifies a

“configuration process” – like the configuration processes disclosed in Loomans and Stahl – as an “inference procedure.” (Ex. 1101 at 2:64-65.) Such models (and the rules contained therein) are commonly used during the resolving stage of a configuration problem in a configuration system.

203. Accordingly, it is my opinion that Loomans in view of Stahl discloses *a computer system to implement an inference procedure for responding to one or more configuration queries using configuration sub-models, the system comprising.*

... [45.1] means for receiving one or more configuration queries representing a questions involving parts and part relationships in a configuration of a configurable product;

204. I have been informed that this claim limitation is to be construed in accordance with 35 U.S.C. § 112 ¶6, which provides that an “element in a claim for a combination may be expressed as a means or step for performing a specified function without the recital of structure.” 35 U.S.C. § 112 ¶6. I have also been informed that such claim limitations “shall be construed to cover the corresponding structure, material, or acts described in the specification and equivalents thereof.” *Id.*

205. Based on my review of the ‘057 Patent, the structure that is disclosed for performing the function described in this limitation is a general purpose

microprocessor. (*See* Ex. 1101, ['057 Patent] at 11:41-12:10.)

206. I have also reviewed the '057 Patent for any disclosure regarding the implementation of this limitation by a general purpose microprocessor. My review uncovered that the only disclosure in the specification pertaining to this claim limitation is nothing more than a summary of the function of this claim limitation. (Ex. 1101 ['057 Patent] at 4:40-5:4, Fig. 4.) My review of the '057 specification did not uncover a disclosure of an algorithm by which the general purpose microprocessor performs the function of this claim limitation. As described in my analysis below for this limitation, it is my opinion that Loomans in view of Stahl discloses the functionality of this claim limitation.

207. Loomans describes a system, which is capable of implementing the disclosed configuration technology: "FIG. 8 is a simplified diagram of an embodiment of a configuration system 800 that may be capable of implementing various aspects and embodiments of the invention." (Ex. 1105 at 16:26-28.) Loomans discloses an embodiment of a computer that can be used with the invention described Loomans, which includes processor(s) 910.

FIG. 9 is a block diagram of an embodiment of a computer system 900 that may be used to implement host server 810 or client computers 820. System 900 includes a bus 908 that interconnects major subsystems such as one or more **processors 910**, a memory subsystem 912, a data storage subsystem 914, an input device

interface 916, an output device interface 918, and a network interface 920. **Processor(s) 910 perform many of the processing functions for system 900 and communicate with a number of peripheral devices via bus 908.**

(Ex. 1105 [Loomans] at 17:42-50, emphasis added.)

Loomans further discloses “[a] computer program product for performing sub-configuration of components of an entity, comprising a computer-usable storage medium having embodied therein computer-readable program codes executable by a processor. (Ex. 1105 [Loomans] at 18:35-39, emphasis added.) Thus, Loomans discloses a general purpose microprocessor for performing sub-configuration.

208. During prosecution of the ‘057 Patent, the applicants admitted that “after selection of different components, such as a printer, **the selections themselves are used to form a configuration-type query.** (Ex. 1104 [‘057 Patent File History (3/18/09 Office Action Response)] at 273.) Thus, the applicants acknowledged that a selection of a component can function as a configuration query. Further, the applicants stated that validating a configuration was an example of querying the configuration: “[d]etermining whether a set of selections represents a valid configurable build can be an example of a configuration query.” (*Id.*)

209. Stahl discloses that the “[t]he starting point of the configuration process is the query represented by an incomplete instantiation of the **compositional structure.**” (Ex. 1106 [Stahl] at 4.) Stahl discloses Figure 3

(below) as an example query with a “root note as our actual problem.” (*Id.*)

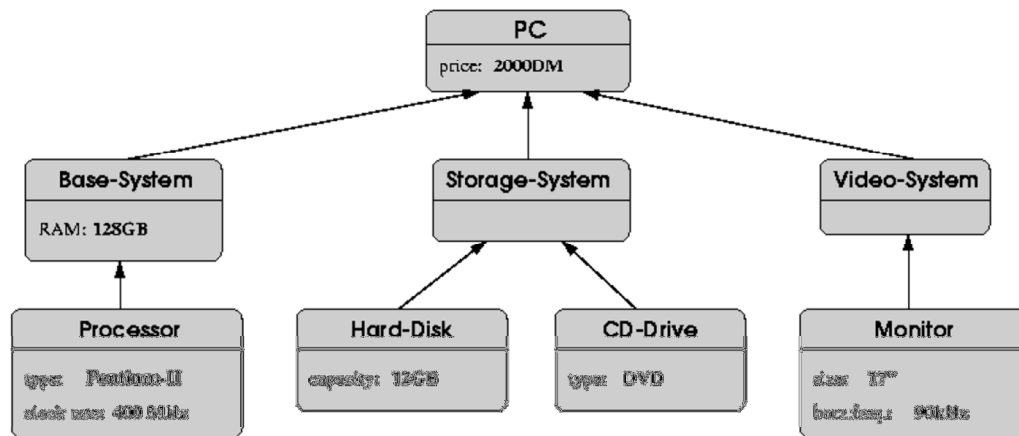


Fig. 3. An Example Query.

(*Id.* at Figure 3.)

Referring to Figure 3, Stahl describes the actual problem, i.e. the query, as a search for “a PC with a set of special properties.” Stahl then goes on to discuss that it “is necessary to select appropriate components that fulfill the properties of the respective part-queries.” (*Id.* at 4.) A person having ordinary skill in the art would have understood that Stahl’s discussion of “part-queries” in the context of configuring a PC would constitute the “one or more questions involving parts and part relationships in a configuration of a configurable product” of this claim limitation. Indeed, such “part-queries” would be necessary to determine if certain selected components are compatible with other selected components – i.e., whether a proposed configuration is buildable.

210. Further, Loomans discloses a technique to configure complex entities by partitioning the complete configuration model into sub-models, which can then

be individually configured and validated.

The invention provides techniques to configure complicated entities using sub-configuration, which effectively partitions the configuration of a complicated top-level entity into a set of configurations of smaller sub-level entities (i.e., components of the top-level entity) in combination with a configuration of a "simplified" top-level entity. The top-level entity may be represented by and configured via a parent model, and each sub-configurable component may be represented by and configured via a child sub-model.

(Ex. 1105 [Loomans] at 1:57-65, emphasis added.)

211. Loomans next discloses configuring an entire entity using a parent model and configuring each sub-configurable component with a sub-model.

A specific embodiment of the invention provides a method for performing sub-configuration of components of an entity. In the method, **the entity is configured via a parent model and each sub-configurable component is configured via one of a number of sub-models. Initially a selection to configure a particular sub-configurable component of the entity is received, and a sub-model for the selected component is identified.** One or more values for one or more features of the selected component are received (e.g., from the parent model or via the sub-model) and form a configuration for the component, which is then validated based on the associated sub-model and the received values. Configuration of the entity is also

validated based on the parent model and the validated configuration for the selected component.

(Ex. 1105 [Loomans] at 2:25-38, emphasis added.)

212. Loomans further discloses the use of queries as part of the configuration process:

With sub-configuration, a complicated entity may be more efficiently modeled, configured, and validated. Each sub-configurable option of the entity may be configured and validated via a respective child sub-model. Generally, sufficient information is made available to the child sub-model such that the associated option can be validly configured. **The required information may be provided from the parent level, queried and entered at the sub-level via the child sub-model, and/or provided via some other mechanisms.**

(Ex. 1105 [Loomans] at 4:59-67, emphasis added.)

213. Loomans further discloses that the configuration process includes validating the configuration:

During the configuration process for computer system 100, **one of the available choices for each top-level option may be selected.** A default value may also be assigned to each top-level option and may be used as the initial choices or if no selection is received for the option. The combination of the default and selected choices for all top-level options comprises a specific configuration for system 100. **In a typical conventional implementation, once the choices for all**

options have been selected, the resultant configuration may be validated.

(Ex. 1105 [Loomans] at 4:7-16, emphasis added.)

214. A person having ordinary skill in the art would have understood that the configuring a sub-model or validating a sub-model described in Loomans corresponds to the “query” nomenclature of the ‘057 Patent. Indeed, the processing and validation of the model and sub-models in Loomans would include processing one or more queries associated with those models. Otherwise, those models would be unable to perform any functions, as the query begins the configuration process. At a minimum the system of Loomans discloses processing the query, e.g., “Is this submodel in a valid state?”.

215. Accordingly, it is my opinion that Loomans in view of Stahl discloses *a means for receiving one or more configuration queries representing a questions involving parts and part relationships in a configuration of a configurable product.*

... [45.2] means for dividing one or more configuration queries into multiple configuration sub-queries, wherein the multiple configuration sub-queries represent the one or more configuration queries;

216. I have been informed that this claim limitation is to be construed in accordance with 35 U.S.C. § 112 ¶6, which provides that an “element in a claim for a combination may be expressed as a means or step for performing a specified

function without the recital of structure.” 35 U.S.C. § 112 ¶6. I have also been informed that such claim limitations “shall be construed to cover the corresponding structure, material, or acts described in the specification and equivalents thereof.”

Id.

217. Based on my review of the ‘057 Patent, the structure that is disclosed for performing the function described in this limitation is a general purpose microprocessor. (*See* Ex. 1101, [’057 Patent] at 11:41-12:10.)

218. I have also reviewed the ‘057 Patent for any disclosure regarding the implementation of this limitation by a general purpose microprocessor. My review uncovered that the only disclosure in the specification pertaining to this claim limitation is nothing more than a summary of the function of this claim limitation. (Ex. 1101 [’057 Patent] at 5:55-6:15.) My review of the ‘057 specification did not uncover a disclosure of an algorithm by which the general purpose microprocessor performs the function of this claim limitation. As described in my analysis below for this limitation, it is my opinion that Loomans in view of Stahl discloses the functionality of this claim limitation.

219. Loomans describes a system, which is capable of implementing the disclosed configuration technology: “FIG. 8 is a simplified diagram of an embodiment of a configuration system 800 that may be capable of implementing various aspects and embodiments of the invention.” (Ex. 1105 at 16:26-28.)

Loomans discloses an embodiment of a computer that can be used with the invention described Loomans, which includes processor(s) 910.

FIG. 9 is a block diagram of an embodiment of a computer system 900 that may be used to implement host server 810 or client computers 820. System 900 includes a bus 908 that interconnects major subsystems such as one or more **processors 910**, a memory subsystem 912, a data storage subsystem 914, an input device interface 916, an output device interface 918, and a network interface 920. **Processor(s) 910 perform many of the processing functions for system 900 and communicate with a number of peripheral devices via bus 908.**

(Ex. 1105 [Loomans] at 17:42-50, emphasis added.)

Loomans further discloses “[a] computer program product for performing sub-configuration of components of an entity, comprising a computer-usable storage medium having embodied therein computer-readable program codes executable by a processor. (Ex. 1105 [Loomans] at 18:35-39, emphasis added.) Thus, Loomans discloses a general purpose microprocessor for performing sub-configuration.

220. Stahl discloses partitioning a query into sub-problems (i.e., sub-queries), which are then solved and combined back together to form the overall solution to the configuration problem (i.e., configuration query)

Query. The starting point of the configuration process is the query represented by an incomplete instantiation of the compositional structure. When looking at the example query shown

in Fig. 3, **we can interpret the root note as our actual problem, i.e., we are searching a PC with a set of special properties.** To reach this goal it is necessary to select appropriate components that fulfill the properties of the respective part-queries. In our example, one part-query states that the PC should have a hard-disk with 12GB of capacity. To fulfill this demand we can, e.g., integrate the concrete hard-disk "Maxtor91303D6" for the hard-disk part in the PC. **Thus, we can interpret the different leaf nodes of the query as sub-problems that must be solved to solve the overall problem, i.e., the configuration of the required PC.** If we have found suitable sub-solutions, i.e. suitable components, for every part-query, **we have to combine these sub-solutions to a final solution for the overall configuration problem.**

(Ex. 1106 [Stahl] at 4-5, emphasis added.)

221. A person having ordinary skill in the art would have understood that the sub-problems disclosed in Stahl when combined represent the original query. Indeed, Stahl discloses that "we can interpret **the different leaf nodes of the query as sub-problems** that must be solved to solve the overall problem." (Ex. 1106 [Stahl] at 5, emphasis added.) Thus, the sub-problems (leaf nodes) ultimately make up the complete query, as displayed in the enhancement of Figure 3 below.

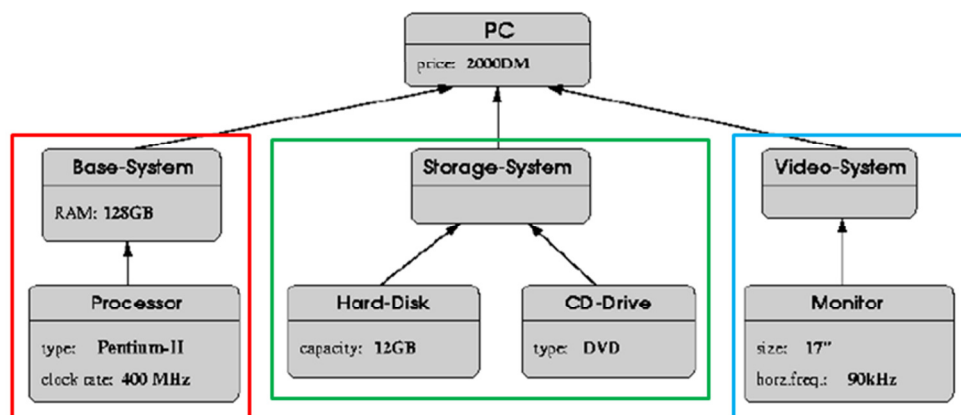


Fig. 3. An Example Query.

Base-System sub-problem + Storage System sub-problem + Video-System sub-problem = PC Problem (Query)

(Ex. 1106 [Stahl] at Fig. 3)

222. Further, Loomans discloses dividing a top-level entity, such as a computer system, represented via a parent model into sub-configurable components represented by child sub-models.

The invention provides techniques to configure complicated entities using sub-configuration, which provides numerous benefits. Sub-configuration effectively **partitions** the overall configuration of a complicated top-level entity (e.g., computer system 100) into a set of configurations of smaller sub-level entities (e.g., Drive Bays 1, 2, and 3) in combination with a less complicated configuration of a "simplified" top-level entity. **The top-level entity may be represented by and configured via a parent model, and each sub-configurable component (or sub-level entity) may be represented by and configured via a child sub-model.**

(Ex. 1105 [Loomans] at 4:48-58, emphasis added.)

223. Loomans next discloses validating sub-configurable options with a child sub-model, a process which includes querying each child sub-model.

With sub-configuration, a complicated entity may be more efficiently modeled, configured, and validated. **Each sub-configurable option of the entity may be configured and validated via a respective child sub-model.** Generally, sufficient information is made available to the child sub-model such that the associated option can be validly configured. **The required information may be provided from the parent level, queried and entered at the sub-level via the child sub-model,** and/or provided via some other mechanisms.

(Ex. 1105 [Loomans] at 4:59-67, emphasis added.)

224. A person having ordinary skill in the art would have understood that the configuring a sub-model or validating a sub-model described in Loomans corresponds to the “query” nomenclature of the ‘057 Patent. Indeed, the processing and validation of the model and sub-models in Loomans would include processing one or more queries associated with those models. Otherwise, those models would be unable to perform any functions, as the query begins the configuration process. At a minimum the system of Loomans discloses processing the query, e.g., “Is this submodel in a valid state?”.

225. Further, a person having ordinary skill in the art would have understood that the one or more configuration queries received in regards to the top entity (parent) model in Loomans would have needed to be divided into sub-

queries when being applied to each child sub-model. Otherwise, the system would be unable to determine the appropriate sub-model to apply to the portion of the query being configured.

226. Finally, to the extent Loomans does not expressly disclose *dividing one or more configuration queries into multiple configuration sub-queries*, and/or lacks details thereof, a person having ordinary skill in the art would have understood that the teachings of dividing a query into sub-problems described in Stahl could have been applied. Indeed, a person having ordinary skill in the art would have understood that querying sub-models, like the sub-models described in Loomans, would have necessitated a condensed sub-problem (i.e., sub-query) like the sub-problems described in Stahl, to ensure efficient operation of the configuration system. It would be inefficient to use a consolidated configuration query to query a sub-model, which only contains information for a portion of the overall product.

227. Accordingly, it is my opinion that Loomans in view of Stahl discloses *a means for dividing one or more configuration queries into multiple configuration sub-queries, wherein the multiple configuration sub-queries represent the one or more configuration queries*.

... [45.3] means for processing each sub-query using at least one configuration sub-model per sub-query,

228. I have been informed that this claim limitation is to be construed in accordance with 35 U.S.C. § 112 ¶6, which provides that an “element in a claim for a combination may be expressed as a means or step for performing a specified function without the recital of structure.” 35 U.S.C. § 112 ¶6. I have also been informed that such claim limitations “shall be construed to cover the corresponding structure, material, or acts described in the specification and equivalents thereof.” *Id.*

229. Based on my review of the ‘057 Patent, the structure that is disclosed for performing the function described in this limitation is a general purpose microprocessor. (*See* Ex. 1101 [‘057 Patent] at 11:41-12:10.)

230. I have also reviewed the ‘057 Patent for any disclosure regarding the implementation of this limitation by a general purpose microprocessor. My review uncovered that the only disclosure in the specification pertaining to this claim limitation is nothing more than a summary of function of this claim limitation. (Ex. 1101 [‘057 Patent] at 6:16-21.) My review of the ‘057 specification did not uncover a disclosure of an algorithm by which the general purpose microprocessor performs the function of this claim limitation. As described in my analysis below for this limitation, it is my opinion that Loomans in view of Stahl discloses the

functionality of this claim limitation.

231. Loomans discloses dividing a top-level entity, such as a computer system, represented by a parent model, into sub-configurable components represented by child sub-models.

The invention provides techniques to configure complicated entities using sub-configuration, which provides numerous benefits. Sub-configuration effectively **partitions** the overall configuration of a complicated top-level entity (e.g., computer system 100) into a set of configurations of smaller sub-level entities (e.g., Drive Bays 1, 2, and 3) in combination with a less complicated configuration of a "simplified" top-level entity. **The top-level entity may be represented by and configured via a parent model, and each sub-configurable component (or sub-level entity) may be represented by and configured via a child sub-model.**

(Ex. 1105 [Loomans] at 4:48-58, emphasis added.)

232. Loomans next discloses querying for information via the child sub-model.

With sub-configuration, a complicated entity may be more efficiently modeled, configured, and validated. **Each sub-configurable option of the entity may be configured and validated via a respective child sub-model.** Generally, sufficient information is made available to the child sub-model such that the associated option can be validly configured. **The required information may be provided from the**

parent level, queried and entered at the sub-level via the child sub-model, and/or provided via some other mechanisms.

(Ex. 1105 [Loomans] at 4:59-67, emphasis added.)

233. Loomans further discloses configuring and validating one or more features of a selected component using an associated sub-model.

A specific embodiment of the invention provides a method for performing sub-configuration of components of an entity. In the method, **the entity is configured via a parent model and each sub-configurable component is configured via one of a number of sub-models. Initially a selection to configure a particular sub-configurable component of the entity is received, and a sub-model for the selected component is identified. One or more values for one or more features of the selected component are received (e.g., from the parent model or via the sub-model) and form a configuration for the component, which is then validated based on the associated sub-model and the received values.** Configuration of the entity is also validated based on the parent model and the validated configuration for the selected component.

(Ex. 1105 [Loomans] at 2:25-38, emphasis added.)

234. As discussed above, the act of selecting a component, and then validating the configuration, includes a configuration query. Indeed, the query is used in the configuration of the product. Thus, a PHOSITA would have understood that Loomans' disclosure of selecting and validating a particular component

(represented by a sub-model) would include querying that sub-model.

235. Further, Stahl discloses interpreting a query as a series of “sub-problems”, which can be solved with “sub-solutions.”

Thus, **we can interpret the different leaf nodes of the query as sub-problems** that must be solved to solve the overall problem, i.e., the configuration of the required PC. **If we have found suitable sub-solutions, i.e. suitable components, for every part-query**, we have to combine these sub-solutions to a final solution for the overall configuration problem.

(Ex. 1106 [Stahl] at 5, emphasis added.)

236. To the extent Stahl does not expressly disclose models and sub-models, and/or lacks details thereof, for providing solutions and sub-solutions to the Query and sub-problems in Stahl, a person having ordinary skill in the art would have understood that models and sub-models like those described in Loomans could have been used to process each sub-problem described in Stahl. Indeed, as discussed above the query (and sub-queries) are merely the starting point of the configuration process, which then must be solved using the rules provided in models (like the models and sub-models described in Loomans). Indeed, Loomans discloses that “each sub-configurable component is configured via one of a number of sub-models.” (Ex. 1105 [Loomans] at Abstract.) And Stahl discloses that each sub-problem represents a “part-query” of a component. (Ex.

1106 [Stahl] at 5.) A person having ordinary skill in the art would have recognized that the models, which contain the configuration rules for the product, would be queried to solve the sub-problems and the ultimate query. Indeed, a PHOSITA would recognize that the models (and sub-models) contain all of the required information to configure the final product. The feature-specific sub-models taught in Loomans would have informed a PHOSITA how to process and solve the sub-problems taught in Stahl for specific features.

237. Accordingly, it is my opinion that Loomans in view of Stahl discloses *a means for processing each sub-query using at least one configuration sub-model per sub-query.*

... [45.4] wherein each configuration sub-model collectively models the configurable product and each configuration sub-model includes data to define compatibility relationships between parts included in the configuration sub-model

238. Loomans discloses a top level entity that is represented by a parent model, which is then partitioned into sub-level entities that are represented as child sub-models and include the components of the top-level entity.

The invention provides techniques to configure complicated entities using sub-configuration, which **effectively partitions the configuration of a complicated top-level entity into a set of configurations of smaller sub-level entities (i.e., components of the top-level entity)** in combination with a configuration of a "simplified"

top-level entity. The top-level entity may be represented by and configured via a parent model, and **each sub-configurable component may be represented by and configured via a child sub-model.**

(Ex. 1105 [Loomans] at 1:57-65, emphasis added.)

239. Thus, Loomans discloses dividing the top-level entity, which is represented as the parent model, into sub-level entities that are represented as child sub-models. A person having ordinary skill in the art would have understood that the partitioned sub-models taught by Loomans would collectively include all of the information from the top-entity parent model to ensure that the full configurable product remains intact for configuration. This is consistent with the purpose of the invention of Loomans, which is to “configure complicated entities.” (*Id.* at 1:57-58.)

240. Loomans further discloses that each sub-configurable option may be configured based on its own set of part options/choices.

Each top-level sub-configurable option may be configured based on its associated sets of sub-level options. For example, each of Drive Bays 1, 2, and 3 may be used to install a disk drive, a CD-drive, or a hard disk (i.e., three possible options), or nothing at all (which may be a default). The Disk Drive, CD-Drive, and Hard Disk options may each be further associated with one or more sets of choices that are specific for that option.

(Ex. 1105 [Loomans] at 3:41-49, emphasis added.)

241. Loomans discloses that the sub-models include data maps, which defines the available parts in a sub-model for a particular option.

As shown in FIG. 3A, structure 300 includes a sub-model mapping set 310 that includes a number of (N) elements, one element for each sub-configurable option at the top level. For example, **sub-model mapping set 310 may include four elements for options C, D, E, and F (i.e., Drive Bays 1, 2, and 3 and Storage) of computer system 100.** Each element of mapping set 310 is associated with a respective sub-model map 320. For example, **option F (Storage) may be associated with a sub-model map that includes the two available types of storage devices ("ABC" and "XYZ").** Similarly, **option C (Drive Bay 1) may be associated with a sub-model map that includes the three available drive types (Disk Drive, CD-Drive, and Hard Disk).** Each sub-model map 320 may further be associated with a set of (M) sub-models 330, where M may be one or greater. For example, the sub-model map for option F (Storage) may include a first sub-model for the ABC storage device and a second sub-model for the XYZ storage device.

(Ex. 1105 [Loomans] at 6:30-46, emphasis added.)

242. Thus, Loomans discloses that each child sub-model includes information, in the form of sub-model mapping, which defines the part choices (e.g., types of storage) for particular options. A person having ordinary skill in the art would have understood that this would include defining compatibility

relationships between the part choices to ensure that the configured product is buildable, i.e., the chosen parts are compatible.

243. Accordingly, it is my opinion that Loomans in view of Stahl discloses that *each configuration sub-model collectively models the configurable product and each configuration sub-model includes data to define compatibility relationships between parts included in the configuration sub-model.*

... [45.5] and each configuration sub-model (i) represents a portion of a configuration model of the configurable product and (ii) allows answers from each configuration sub-model to be combined to provide a consolidated answer to the one or more configuration queries;

244. Loomans discloses a top-level entity that is represented by a parent model, which is then partitioned into sub-level entities that are represented as child sub-models and include the components of the top-level entity.

The invention provides techniques to configure complicated entities using sub-configuration, which **effectively partitions the configuration of a complicated top-level entity into a set of configurations of smaller sub-level entities (i.e., components of the top-level entity)** in combination with a configuration of a "simplified" top-level entity. The top-level entity may be represented by and configured via a parent model, and **each sub-configurable component may be represented by and configured via a child sub-model.**

(Ex. 1105 [Loomans] at 1:57-65, emphasis added.)

245. Thus, Loomans discloses dividing the top-level entity, which is represented as the parent model, into sub-level entities that are represented as child sub-models. A person having ordinary skill in the art would have understood that the partitioned sub-models each represent a portion of the parent model for the top-level entity (i.e., configurable product).

246. Loomans further discloses that information from each sub-model is made available to the parent model so that the parent model can be validated and configured in view of any new configurations made at the sub-level – i.e., the sub-model level.

Once a sub-configurable option has been configured and validated, the parent model may be returned to, **and any information from the sub-model that may be needed by the parent model is made available to the parent.** Upon a return from the sub-level, or whenever directed, the parent model can be run (i.e., executed) in the context of all options that have been selected or configured. This allows the parent model to be validated with any new configuration made at the sub-level.

(Ex. 1105 [Loomans] at 5:10-18, emphasis added.)

247. For example, Loomans discloses validating the parent model based on the configuration and validation of sub-features using sub-models. .

In one implementation for sub-configuration, which uses mapped features, parameter values needed for configuration and validation at the parent model and child sub-models are passed between these two levels. Each sub-model is provided with sufficient information needed to configure and validate the option represented by that sub-model. Some of the required information may be obtained at the parent model and passed to the sub-model as mapped features. Other information may be collected in the sub-model. **Upon returning from the child sub-model to the parent model, values for the mapped features are returned from the child sub-model to the parent model.**

* * *

Sub-configuration thus allows a complicated entity to be configured in smaller portions and incrementally, one component at a time. **As shown in FIG. 5, a particular option may be configured using sub-configuration and validated. The parent model is then validated based on the current set of features, which includes those for the sub-configured option.** Another option may then be configured using sub-configuration and validated. The parent model is then validated based on the new current set of features.

(Ex. 1105 [Loomans] at 6:11-22 and 11:19-27, emphasis added.)

248. A person having ordinary skill in the art would have understood that based on this teaching Loomans discloses the feature-based sub-models are used for the configuration and validation of the parent model. A person having ordinary skill in the art would have understood that the configuring a sub-model or

validating a sub-model described in Loomans corresponds to the “query” nomenclature of the ‘057 Patent. Indeed, the processing and validation of the model and sub-models in Loomans would include processing one or more queries associated with those models. Otherwise, those models would be unable to perform any functions, as the query begins the configuration process. At a minimum the system of Loomans discloses processing the query, e.g., “Is this submodel in a valid state?”.

249. Further, Stahl discloses combining sub-solutions to each of the sub-problems (i.e., sub queries) to form a final solution to the overall configuration problem.

Thus, we can interpret the different leaf nodes of the query as sub-problems that must be solved to solve the overall problem, i.e., the configuration of the required PC. If we have found suitable sub-solutions, i.e. suitable components, for every part-query, **we have to combine these sub-solutions to a final solution for the overall configuration problem.**

(Ex. 1106 [Stahl] at 5, emphasis added.)

250. Thus, Stahl teaches answering the sub-problems and then combining the answers (i.e., solutions) into a consolidated answer for the overall configuration problem (i.e., the configuration query).

251. To the extent Stahl does not expressly disclose models and sub-

models, and/or lacks details thereof, for providing solutions and sub-solutions to the Query and sub-problems in Stahl, a person having ordinary skill in the art would have understood that models and sub-models like those described in Loomans could have been used to answer each sub-problem described in Stahl. The answers to each sub-problem would then be combined to form the final solution for the overall configuration problem, as described in Stahl. A person having ordinary skill in the art would have recognized that the sub-models, which contain the configuration rules for various features of the product, would be queried to solve the sub-problems and the query. A person having ordinary skill in the art would have recognized that the models (and sub-models) contain all of the required information to configure the final product.

252. Accordingly, it is my opinion that Loomans in view of Stahl discloses that *each configuration sub-model (i) represents a portion of a configuration model of the configurable product and (ii) allows answers from each configuration sub-model to be combined to provide a consolidated answer to the one or more configuration queries.*

... [45.6] means for generating a response to the one or more configuration queries based upon the processing of each sub-query using at least one configuration sub-model per sub-query; and;

253. I have been informed that this claim limitation is to be construed in

accordance with 35 U.S.C. § 112 ¶6, which provides that an “element in a claim for a combination may be expressed as a means or step for performing a specified function without the recital of structure.” 35 U.S.C. § 112 ¶6. I have also been informed that such claim limitations “shall be construed to cover the corresponding structure, material, or acts described in the specification and equivalents thereof.”

Id.

254. Based on my review of the ‘057 Patent, the structure that is disclosed for performing the function described in this limitation is a general purpose microprocessor. (*See* Ex. 1101 [‘057 Patent] at 11:41-12:10.)

255. I have also reviewed the ‘057 Patent for any disclosure regarding the implementation of this limitation by a general purpose microprocessor. My review uncovered that the only disclosure in the specification pertaining to this claim limitation is nothing more than a summary of function of this claim limitation. (Ex. 1101 [‘057 Patent] at 6:16-23.) My review of the ‘057 specification did not uncover a disclosure of an algorithm by which the general purpose microprocessor performs the function of this claim limitation. As described in my analysis below for this limitation, it is my opinion that Loomans in view of Stahl discloses the functionality of this claim limitation.

256. Loomans discloses an embodiment of a computer that can be used with the invention described Loomans, which includes processor(s) 910.

FIG. 9 is a block diagram of an embodiment of a computer system 900 that may be used to implement host server 810 or client computers 820. System 900 includes a bus 908 that interconnects major subsystems such as one or more **processors 910**, a memory subsystem 912, a data storage subsystem 914, an input device interface 916, an output device interface 918, and a network interface 920. **Processor(s) 910 perform many of the processing functions for system 900 and communicate with a number of peripheral devices via bus 908.**

(Ex. 1105 [Loomans] at 17:42-50, emphasis added.)

Loomans further discloses “[a] computer program product for performing sub-configuration of components of an entity, comprising a computer-usable storage medium having embodied therein computer-readable program codes executable by a **processor**. (Ex. 1105 at 18:35-39, emphasis added.) Thus, Loomans discloses a general purpose microprocessor for performing sub-configuration.

257. Loomans discloses configuring and validating one or more features of a selected component using an associated sub-model.

A specific embodiment of the invention provides a method for performing sub-configuration of components of an entity. In the method, **the entity is configured via a parent model and each sub-configurable component is configured via one of a number of sub-models**. Initially a selection to configure a particular sub-configurable component of the entity is received, and a sub-model for the selected component is identified. One or more values for one or more features

of the selected component are received (e.g., from the parent model or via the sub-model) and form a configuration for the component, which is then validated based on the associated sub-model and the received values. **Configuration of the entity is also validated based on the parent model and the validated configuration for the selected component.**

(Ex. 1105 [Loomans] at 2:25-38, emphasis added.)

258. Loomans discloses configuring and validating features at the sub-model level and then returning the configuration values back to the parent model.

In one implementation for sub-configuration, which uses mapped features, parameter values needed for configuration and validation at the parent model and child sub-models are passed between these two levels. Each sub-model is provided with sufficient information needed to configure and validate the option represented by that sub-model. Some of the required information may be obtained at the parent model and passed to the sub-model as mapped features. Other information may be collected in the sub-model. Upon returning from the child sub-model to the parent model, values for the mapped features are returned from the child sub-model to the parent model.

* * *

Sub-configuration thus allows a complicated entity to be configured in smaller portions and incrementally, one component at a time. As shown in FIG. 5, a particular option may be configured using sub-configuration and validated. The parent model is then validated based on the current set of features, which includes those for the sub-

configured option. Another option may then be configured using sub-configuration and validated. The parent model is then validated based on the new current set of features.

(Ex. 1105 [Loomans] at 6:11-22 and 11:19-27.)

259. Loomans further discloses querying for information via the child sub-model.

With sub-configuration, a complicated entity may be more efficiently modeled, configured, and validated. Each sub-configurable option of the entity may be configured and validated via a respective child sub-model. Generally, sufficient information is made available to the child sub-model such that the associated option can be validly configured. The required information may be provided from the parent level, queried and entered at the sub-level via the child sub-model, and/or provided via some other mechanisms.

(Ex. 1105 [Loomans] at 4:59-67, .)

260. A person having ordinary skill in the art would have understood that the configuring a sub-model or validating a sub-model described in Loomans corresponds to the “query” nomenclature of the ‘057 Patent. Indeed, the processing and validation of the model and sub-models in Loomans would include processing one or more queries associated with those models. Otherwise, those models would be unable to perform any functions, as the query begins the configuration process. At a minimum the system of Loomans discloses processing the query, e.g., “Is this

submodel in a valid state?”.

261. Loomans further discloses that “[t]he parent model is then validated based on the current set of features, which includes those for the sub-configured option.” (Ex. 1105 [Loomans] at 11:22-24.) Thus, a response (e.g., validation answer) is provided to the configuration query of the parent model, which includes (is based on) the information generated via sub-configuration of the child sub-models.

262. Thus, a person having ordinary skill in the art would have understood that Loomans discloses configuring each sub-configurable component via sub-models, and then configuring/validating the overall entity via the parent model.

263. Further, Stahl discloses interpreting a query as a series of “sub-problems”, which can be solved with “sub-solutions.”

Thus, we can interpret the different leaf nodes of the query as sub-problems that must be solved to solve the overall problem, i.e., the configuration of the required PC. **If we have found suitable sub-solutions, i.e. suitable components, for every part-query**, we have to combine these sub-solutions to a final solution for the overall configuration problem.

(Ex. 1106 [Stahl] at 5, emphasis added.)

264. To the extent Stahl does not expressly disclose models and sub-models, and/or lacks details thereof, for providing solutions and sub-solutions to

the Query and sub-problems in Stahl, a person having ordinary skill in the art would have understood that models and sub-models like those described in Loomans could have been used to process each sub-problem described in Stahl. Indeed, A person having ordinary skill in the art would have recognized that the sub-models, which contain the configuration rules for various features of the product, would be queried to solve the sub-problems and the query. A person having ordinary skill in the art would have recognized that the models (and sub-models) contain all of the required information to configure the final product. As Stahl discloses, the processed sub-solutions could then be combined to provide a final solution for the overall configuration problem (i.e., a response to the configuration query).

265. Accordingly, it is my opinion that Loomans in view of Stahl discloses *means for generating a response to the one or more configuration queries based upon the processing of each sub-query using at least one configuration sub-model per sub-query.*

... [45.7] means for providing the response to the one or more configuration queries as data for display by a display device.

266. I have been informed that this claim limitation is to be construed in accordance with 35 U.S.C. § 112 ¶6, which provides that an “element in a claim for a combination may be expressed as a means or step for performing a specified

function without the recital of structure.” 35 U.S.C. § 112 ¶6. I have also been informed that such claim limitations “shall be construed to cover the corresponding structure, material, or acts described in the specification and equivalents thereof.”

Id.

267. Based on my review of the ‘057 Patent, the structure that is disclosed for performing the function described in this limitation is a “video amplifier”. The ‘057 Patent states that “video amplifier 816 is used to drive the display 817. Video amplifier 816 is well known in the art and may be implemented by any suitable means. This circuitry converts pixel DATA stored in video memory 814 to a raster signal suitable for use by display 817.” (*See* Ex. 1101 [’057 Patent] at 12:39-41.) As described my analysis below for this limitation, it is my opinion that Loomans in view Stahl disclose a video amplifier as that device is described in the ‘057 Patent.

268. Loomans discloses an “[o]utput device interface 918 [that] provides an interface with various output devices such as a display 962 (e.g., a CRT or an LCD).” (Ex. 1105 [Loomans] at 17:65-67.)

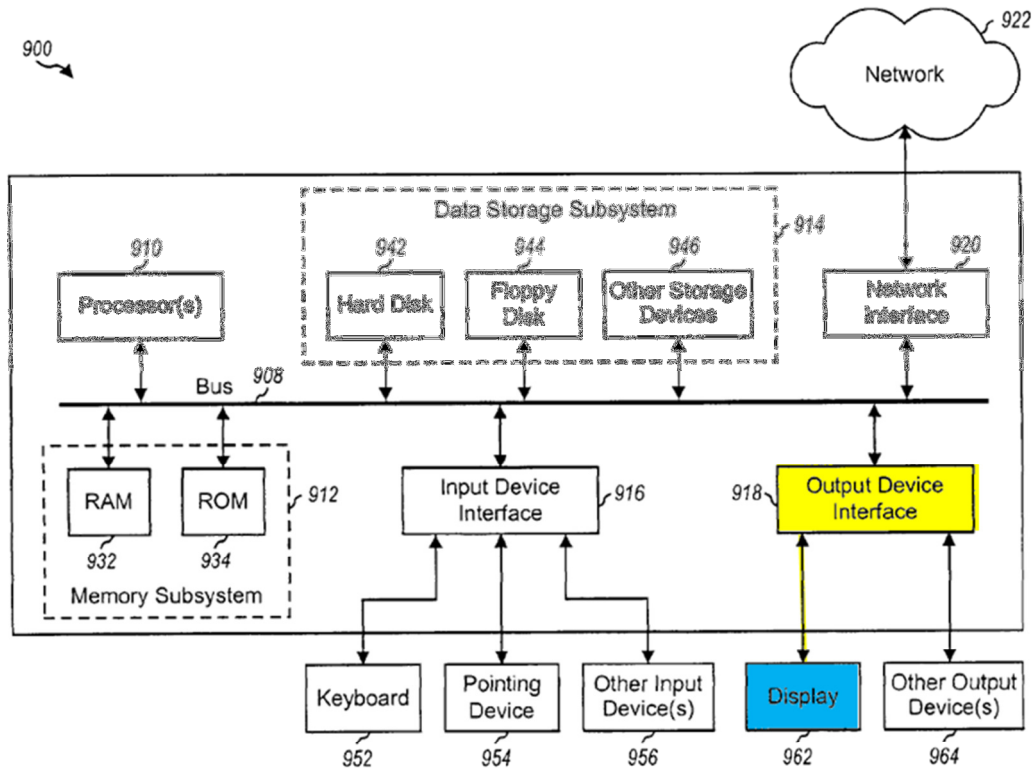


FIG. 9

(Ex. 1105 [Loomans] at Fig. 9)

269. A person having ordinary skill in the art that the computer system disclosed in Loomans would include a video amplifier to convert and send data to the display device for displaying to the user. Indeed, without a video amplifier, the display disclosed in Loomans would be unable to display content, preventing the user from seeing the configuration results. Further, a person having ordinary skill in the art would have concluded that the “output device interface 918” described in Loomans embodies a video amplifier, as that is the device, which provides the interface between the computer system and the display and would require the functionality to convert the data from the processor into a format that can be

interpreted by the display.

270. Loomans discloses a user interface that is used to provide information to an administrator and/or user of the configuration system:

In an embodiment, configuration system 800 includes a number of modules such as a **user interface module 812** and a configuration engine 814. Additional, fewer, and/or different modules may also be included in configuration system 800, and this is within the scope of the invention. **User interface module 812 provides the interface (e.g., screens such as those shown in FIGS. 7A through 7C) used to present information to an administrator and/or a user of the configuration system.** User interface module 812 further receives and interprets user commands and data, which may be provided via mouse clicks, mouse movements, keyboard inputs, and other means. **User interface module 812 then provides the received data and commands to other modules, which then perform the appropriate responsive action.**

(Ex. 1105 [Loomans] at 16:36-49, emphasis added.)

271. Loomans discloses an example of the user interface in Figure 7C: **FIG. 7C shows portion of a screen 790 that may be displayed to show the selections for various options.** In this example screen, the selected feature values for each sub-configurable option and each selectable option are shown on the screen.

(Ex. 1105 [Loomans] at 16:18-22, emphasis added.)

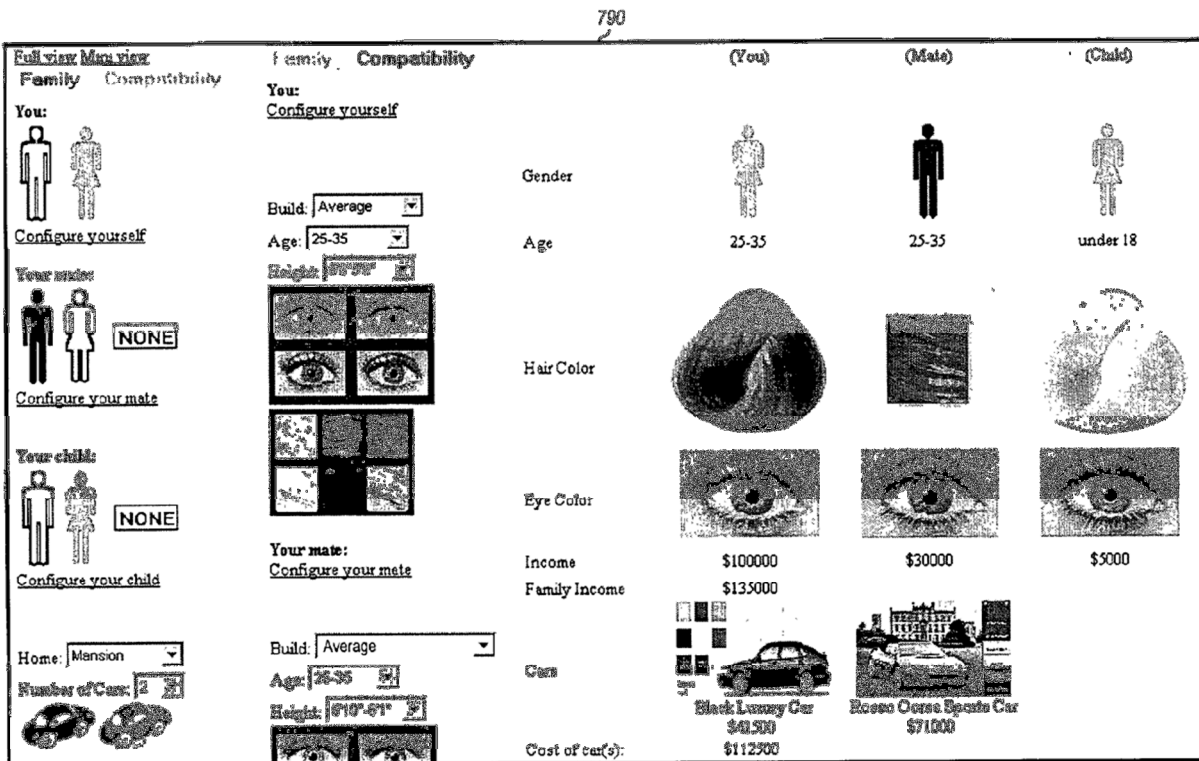


FIG. 7C

(Id. at Fig. 7C)

272. Likewise, Stahl discloses using a display to implement a demo of its disclosed configuration methods over the world wide web.

To evaluate the functionality of our configuration approach we have implemented a generic prototype for the described configuration process. This prototype consists of an extension of the commercial CBR tool CBR-Works which has been developed jointly by the University of Kaiserslautern and Tecinno GmbH. **To be accessible over the World Wide Web the prototype also provides two respective interfaces for the demand acquisition (see Fig. 5) and the result presentation.**

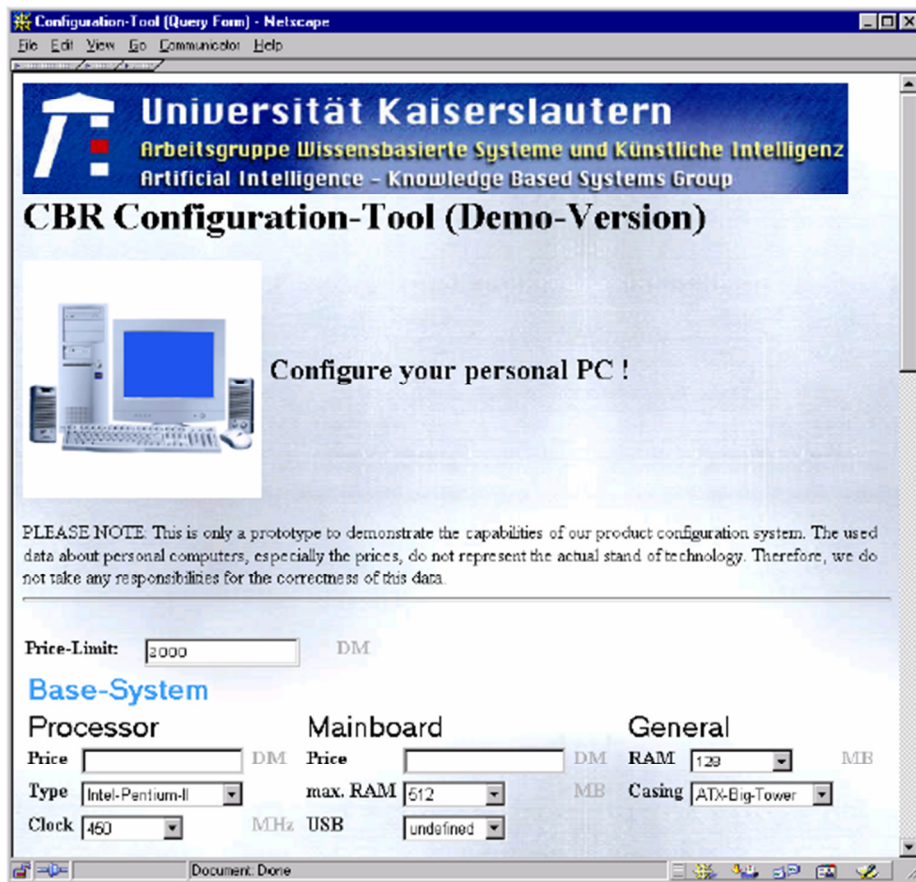


Fig. 5. Implemented demo version.

(Ex. 1106 [Stahl] at 8, emphasis added.)

273. Accordingly, it is my opinion that Loomans in view of Stahl discloses *means for providing the response to the one or more configuration queries as data for display by a display device.*

b. Claim 46

*... [46.0] The computer system of claim 45 further comprising:
means for dividing a consolidated configuration model into the
configuration sub-models.*

274. I have been informed that this claim limitation is to be construed in accordance with 35 U.S.C. § 112 ¶6, which provides that an “element in a claim for a combination may be expressed as a means or step for performing a specified function without the recital of structure.” 35 U.S.C. § 112 ¶6. I have also been informed that such claim limitations “shall be construed to cover the corresponding structure, material, or acts described in the specification and equivalents thereof.”

Id.

275. Based on my review of the ‘057 Patent, the structure that is disclosed for performing the function described in this limitation is a general purpose microprocessor. (*See* Ex. 1101 [‘057 Patent] at 11:41-12:10.)

276. I have also reviewed the ‘057 Patent for any disclosure regarding the implementation of this limitation by a general purpose microprocessor. My review uncovered that the only disclosure in the specification pertaining to this claim limitation is nothing more than a summary of function of this claim limitation. (Ex. 1101 [‘057 Patent] at 4:53-5:4.) My review of the ‘057 specification did not uncover a disclosure of an algorithm by which the general purpose microprocessor

performs the function of this claim limitation. As described in my analysis below for this limitation, it is my opinion that Loomans in view of Stahl discloses the functionality of this claim limitation.

277. Loomans discloses an embodiment of a computer that can be used with the invention described Loomans, which includes processor(s) 910.

FIG. 9 is a block diagram of an embodiment of a computer system 900 that may be used to implement host server 810 or client computers 820. System 900 includes a bus 908 that interconnects major subsystems such as one or more **processors 910**, a memory subsystem 912, a data storage subsystem 914, an input device interface 916, an output device interface 918, and a network interface 920. **Processor(s) 910 perform many of the processing functions for system 900 and communicate with a number of peripheral devices via bus 908.**

(Ex. 1105 [Loomans] at 17:42-50, emphasis added.)

Loomans further discloses “[a] computer program product for performing sub-configuration of components of an entity, comprising a computer-usable storage medium having embodied therein computer-readable program codes executable by a **processor.**” (Ex. 1105 [Loomans] at 18:35-39, emphasis added.) Thus, Loomans discloses a general purpose microprocessor for performing sub-configuration.

278. Loomans discloses dividing a consolidated parent model into multiple child sub-models.

The invention provides techniques to configure complicated entities using sub-configuration, which provides numerous benefits. **Sub-configuration effectively partitions the overall configuration of a complicated top-level entity (e.g., computer system 100) into a set of configurations of smaller sub-level entities (e.g., Drive Bays 1, 2, and 3) in combination with a less complicated configuration of a "simplified" top-level entity. The top-level entity may be represented by and configured via a parent model, and each sub-configurable component (or sub-level entity) may be represented by and configured via a child sub-model.**

(Ex. 1105 [Loomans] at 4:48-58, emphasis added.)

The invention provides techniques to configure complicated entities using sub-configuration, **which effectively partitions the configuration of a complicated top-level entity into a set of configurations of smaller sub-level entities (i.e., components of the top-level entity) in combination with a configuration of a "simplified" top-level entity. The top-level entity may be represented by and configured via a parent model, and each sub-configurable component may be represented by and configured via a child sub-model.**

(*Id.* at 1:57-65, emphasis added.)

279. Accordingly it is my opinion that Loomans in view of Stahl discloses *means for dividing a consolidated configuration model into the configuration sub-models.*

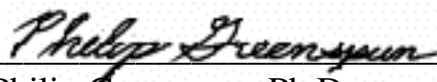
XI. Conclusion

280. In my opinion, all the elements of the challenged claim limitations from the '057 Patent are disclosed by the references discussed above and that the claims are unpatentable in view of these prior art references.

281. I reserve the right to supplement my opinions to address any information obtained, or positions taken, based on any new information that comes to light throughout this proceeding.

I declare under penalty of perjury that the foregoing is true and accurate to the best of my ability.

Executed on: 5/9/2016


Philip Greenspun, Ph.D.