UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

FORD MOTOR COMPANY
Petitioner,

v.

VERSATA DEVELOPMENT GROUP, INC.
Patent Owner.

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

Case No.: IPR2017-00151

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 1-16, 18-29 AND 31-43 OF U.S. PATENT NO. 7,882,057)

TABLE OF CONTENTS

List of Exhibits

Exhibit No.	Description	Date	Identifier
1201	U.S. Patent No. 7,882,057	Feb. 1, 2011	'057 Patent
1202	Expert Declaration of Dr. Philip Greenspun	n/a	Greenspun Decl.
1203	Curriculum Vitae of Dr. Philip Greenspun	n/a	Greenspun CV
1204	U.S. Patent No. 7,882,057 File History	n/a	'057 Patent File History
1205	U.S. Patent No. 7,873,503 to Loomans et al.	Jan. 18, 2011	Loomans
1206	A. Stahl, R. Bergmann, S. Schmitt, A Customization Approach for Structured Products in Electronic Shops, Electronic Commerce: The End of the Beginning, *13th International Bled Electronic Commerce Conference* (June 19-21, 2000)	Jun. 2000	Stahl
1207	Alexander Kott, Gerald Agin, David Fawcett, Configuration Tree Solver: A Technology for Automated Design and Configuration, *ASME Journal of Mechanical Design* 114(1): 187-195 (1992)	1992	Kott
1208	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)	2003	Anselma

Exhibit No.	Description	Date	Identifier
1209	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)	2002	Magro
1210	Judith Bachant, John McDermott, R1 Revisited: Four Years in the Trenches, *AI Magazine* Volume 5, Number 3 (1984)	1984	Bachant
1211	John McDermott, R1: A Rule-Based Configurer of Computer Systems, *Artificial Intelligence* (1982)	1982	McDermott
1212	Bryan M. Kramer, Knowledge-Based Configuration of Computer Systems Using Hierarchial Partial Choice, *IEEE* (1991)	1991	Kramer
1213	Bei Yu and Hans Jorgen Skovgaard, A Configuration Tool to Increase Product Competitiveness, *IEEE Intelligent Systems*, 34-41 (July/August 1998)	1998	Yu
1214	U.S. Patent Application Publication No. 2003/0187950 to Rising	Oct. 2, 2003	Rising
1215	Martin R. Wagner, Understanding the ICAD System, ICAD, Inc., 1990	1990	ICAD
1216	Oracle Configurator Developer, User's Guide, Release 11*i* for Windows 95/98/2000 and Windows NT 4.0	April 2002	Oracle
1217	Stefan Schulz, CBR-Works A State-of-the-Art Shell for Case-Based Application Building, TECINNO GmbH, 1999	1999	CBR Works Paper

Exhibit No.	Description	Date	Identifier
1218	Richard M. Stallman and Gerald Jay Sussman, Forward Reasoning and Dependency-Directed Backtracking In a System for Computer-Aided Circuit Analysis, MIT Artificial Intelligence Laboratory, Memo No. 380, Sept. 1976	Sept. 1976	Stallman
1219	Bergmann Declaration	Oct. 14, 2016	
1220	Schmitt Declaration	Oct. 3, 2016	
1221	Winston Textbook	1992	
1222	*Ford v. Versata*, Versata Answer & Counterclaims (Dkt. #59)	10/28/15	
1223	*Versata v. Ford*, Versata Complaint (Dkt. #1)	5/7/15	
1224	*Ford v. Versata*, Opinion and Order Denying Motion to Dismiss (Dkt. #55)	10/14/15	
1225	*Versata v. Ford*, Order to File Notice of Good Cause (Dkt. #68)	11/5/15	
1226	*Versata v. Ford*, Versata Notice Regarding Dismissal (Dkt. #69)	11/30/15	
1227	*Versata v. Ford*, Order of Dismissal (Dkt. #70)	12/3/15	
1228	*Ford v. Versata*, Amended Complaint (Dkt. #6)	3/16/15	

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 1203.

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.

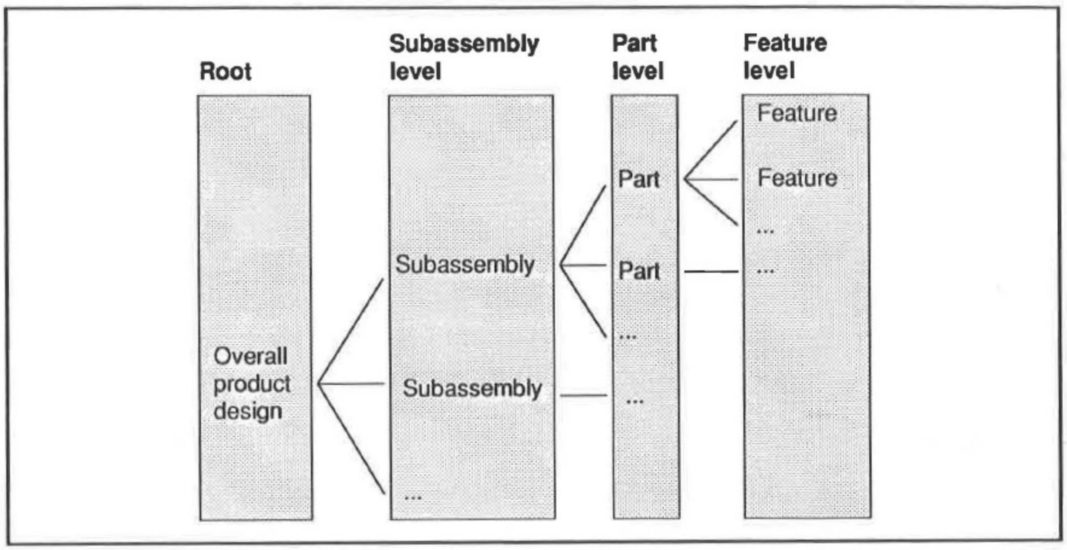
13. ICAD went public as "Concentra" in the 1990s and was acquired by Oracle Corporation in 2002. The product's mechanical design capabilities were deemphasized and its configuration capabilities were improved for use as a general-purpose sales configuration system. The product survives today as Oracle

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. 1215 [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.

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." The toolkit was adopted by approximately 10,000 web sites worldwide.

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. In addition, I participated in developing postclipper.com, a Facebook application that allows parents to create electronic baby books from a subset of their Facebook postings.

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. 1201 ['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. 1201 [‘057 Patent] at 1:16-25.)

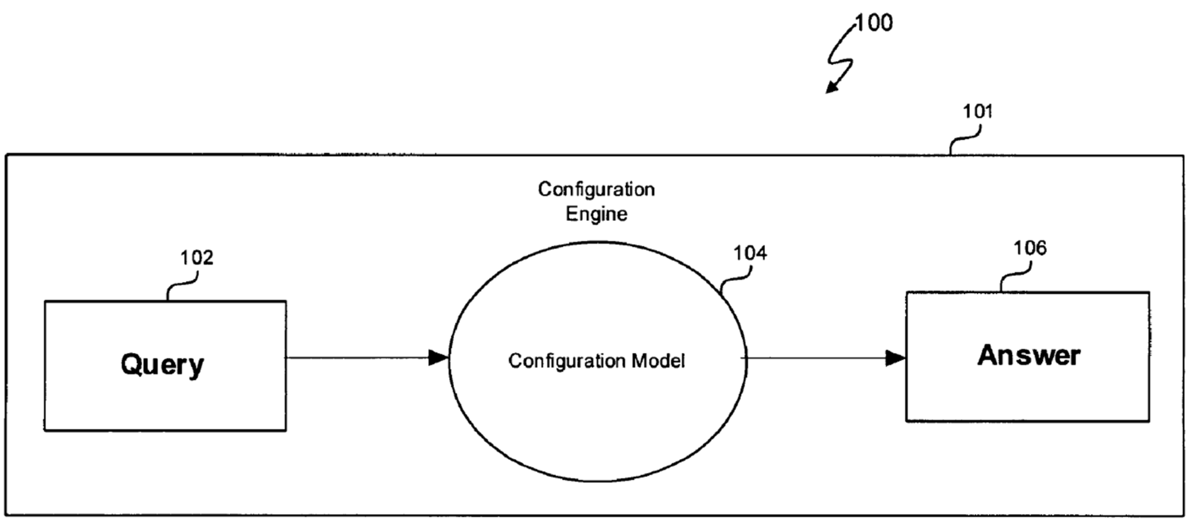


Figure 1 (prior art)

(Ex. 1201 [‘057 Patent] at Figure 1.)

33. The ‘057 Patent further discloses that relationships among features or parts are recorded in a configuration model of a product, which can then be queried:

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. 1201 [‘057 Patent] at 1:26-54, emphasis added.)

34. The ‘057 Patent describes that, at least with a monolithic configuration model, 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. 1201 [‘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, when using prior art techniques that were known to the inventors. 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).

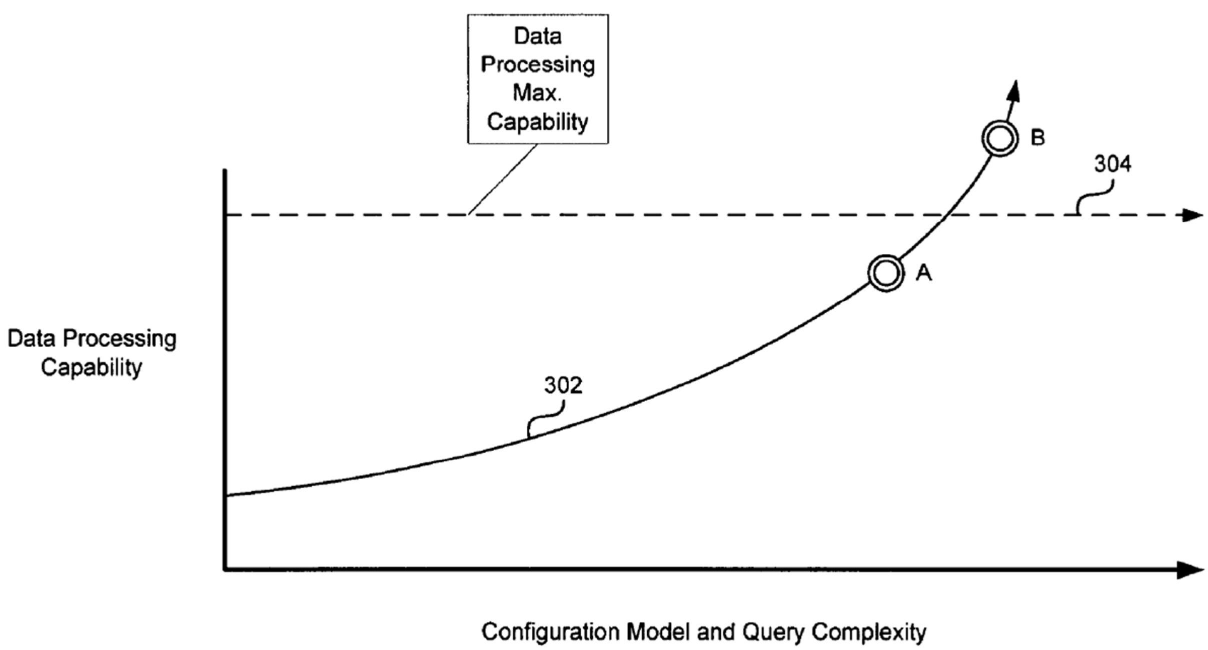


Figure 3 (prior art)

(Ex. 1201 [‘057 Patent] at Fig. 3.)

36. To overcome the limitations imposed by data processing capabilities

on at least some prior art systems, 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. 1201 ['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. 1201 [‘057 Patent] at 4:40-5:4.)

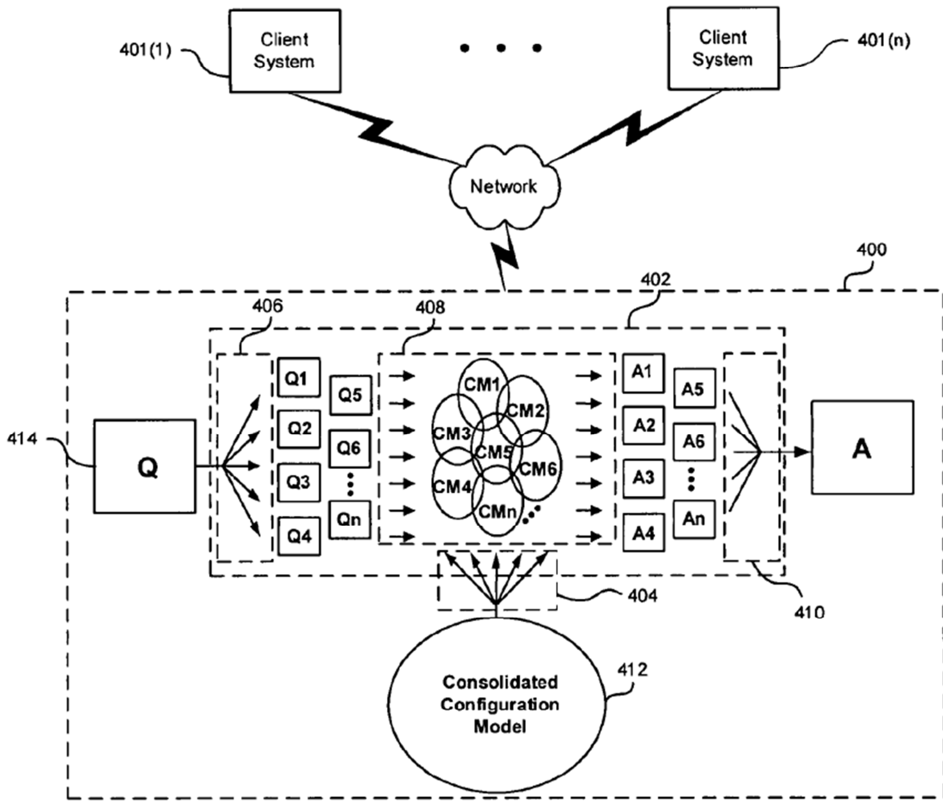


Figure 4

(Ex. 1201 [‘057 Patent] at Fig. 4.)

38. Figure 6 of the ‘057 Patent illustrates an example of a consolidated configuration model (412), which has been divided into configuration sub-models CM_1, CM_2 and CM_3. Note that “In another embodiment, a consolidated configuration model 412 is never actually created, and model developers develop only configuration sub-models to collectively model a configurable product.” (Ex. 1201 [‘057 Patent] at 5:5-8.)

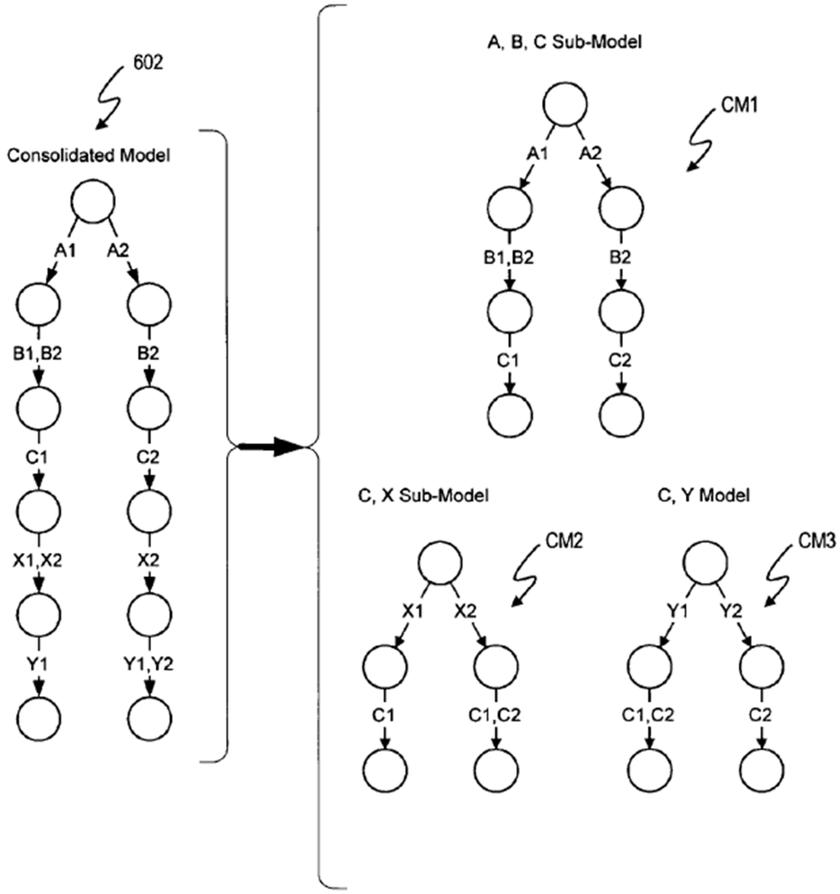


Figure 6

(Ex. 1201 [‘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. 1201 ['057 Patent] at 5:11-18.)

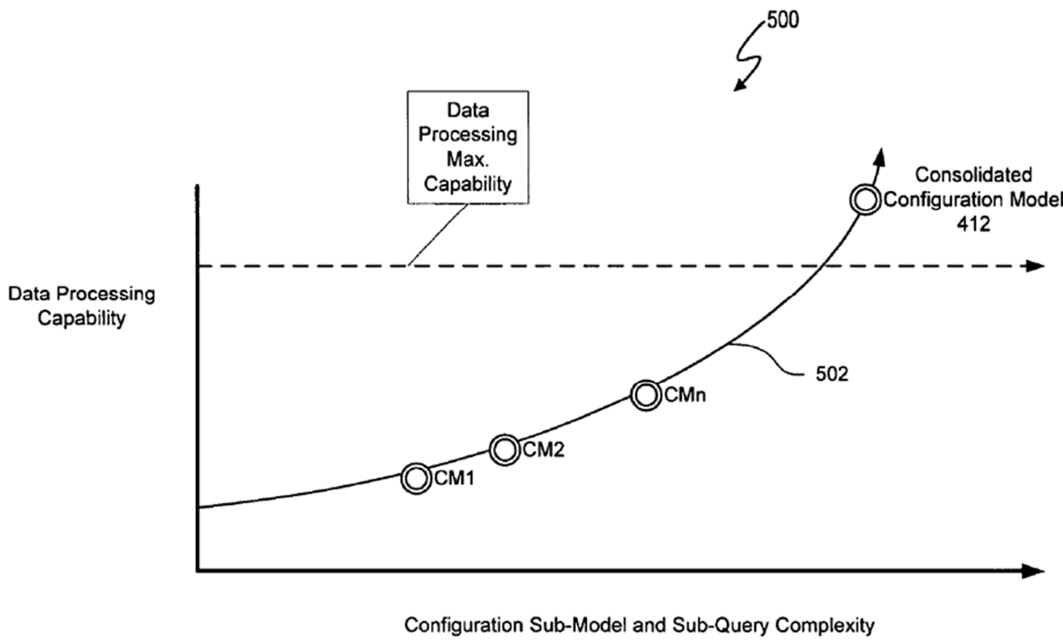


Figure 5

(Ex. 1201 ['057 Patent] at Fig. 5.)

40. As discussed in the "Scope and Content of the Prior Art" and the "Ground for Challenge" sections 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. 1204 ['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 1214.) 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. 1204 at 174-195.) The Applicant distinguished Henson by arguing that Henson did not teach "any type of configuration sub-model or 'processing the one or more configuration queries using configuration sub-models.'" (Ex. 1204 at 227-229.)

Hence, the applicants viewed the point of novelty of the ‘057 Patent as processing queries with sub-models.

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. 1201 [‘057 Patent] at 1:12-25, emphasis added.)

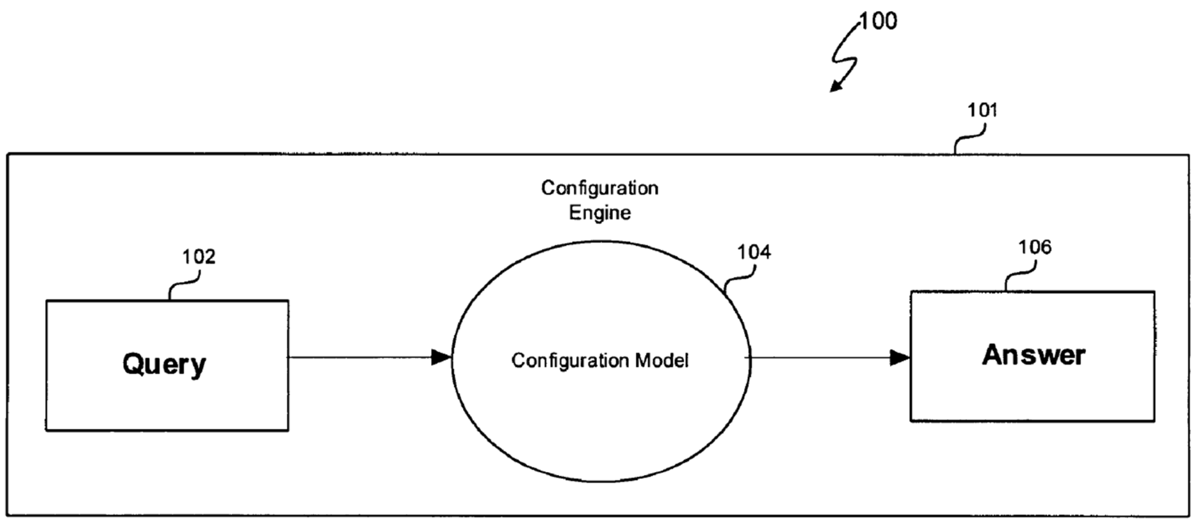


Figure 1 (prior art)

(Ex. 1204 [‘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 1210):

[1] Ex. 1210 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. 1210 [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. 1210 [Bachant] at 6.) A 1982 article, "R1: A Rule-based Configurer of Computer Systems"[2] ("McDermott") (McDermott; *Artificial Intelligence* 19; Attached as Exhibit 1211) describes the process of configuring a computer system

[2] Ex. 1211 is a true and accurate copy of: John McDermott, "R1: A Rule-Based Configurer of Computer Systems," Artificial Intelligence (1982).

as a "task" and that "The configuration task can be viewed as a hierarchy of subtasks…" (Ex. 1211 [McDermott] at 3 and 11.) The minicomputer's 420 different types of components are broken down into 15 classes, such as "cabinet" and "unibus device" (*Id.* at 8.) (Note that a VAX 11/780 "minicomputer" system to be configured with the software described in the paper cost more than $1 million in today's dollars and was the size of several commercial refrigerators.)

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 1212) describes a system in which a computer system is broken down into subcomponents:

[3] Ex. 1212 is a true and accurate copy of: Bryan M. Kramer, "Knowledge-based Configuration of Computer Systems Using Hierarchial 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_Publisher[2] (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. 1212 [Kramer] at 3.)

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

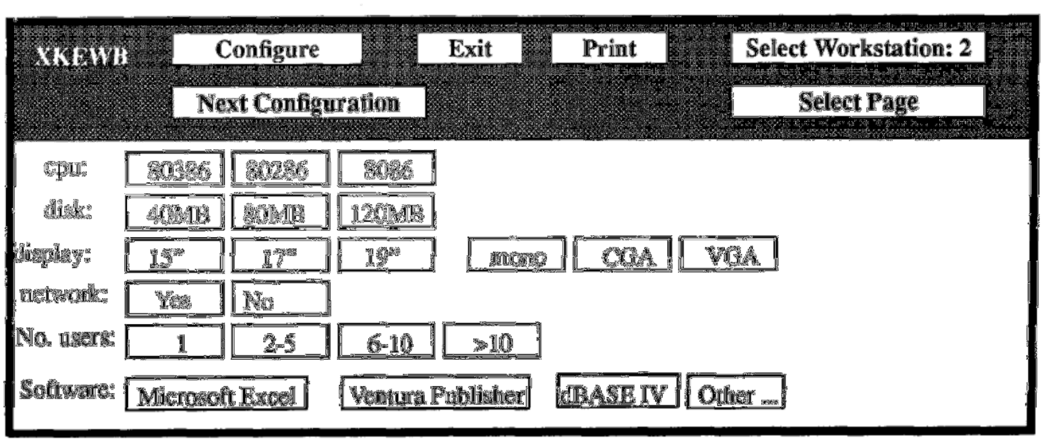


Figure 2: The XKEWB User Interface

(Ex. 1212 [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 Exhibit 1213). 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." (Ex. 1213 [Yu] at 1.)

52. salesPLUS supported submodels:

[4] Ex. 1213 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.

> **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 subassemblies (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. 1213 [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. 1213 [Yu] at 5.) Users ended up with conventional menu-based configuration screen, e.g.,

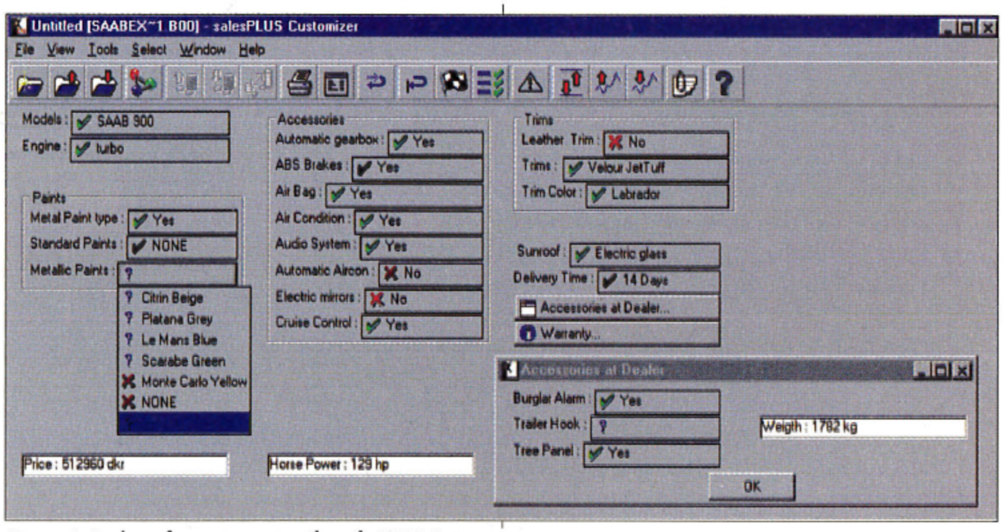


Figure 6. Saab configuration, using the salesPLUS Customizer.

(Ex. 1213 [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.)

54. 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.

55. 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 1207.) Knott begins by stating that "[a] strong hierarchic structure is inherent in many industrial products and systems." (Ex. 1207 [Kott] at 3). Then Knott proceeds to describe tree-based techniques for limiting computational requirements in solving configuration problems.

> 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. 1207 [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

---

[5] Ex. 1207 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).

> 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 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. 1207 [Kott] at 1.)

56. 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 Exhibit 1208); 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 1209). 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

---

[6] Ex. 1208 is a true and accurate copy of: 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).

[7] Ex. 1209 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).

> 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.

(Ex. 1209 [Magro] at 9.)

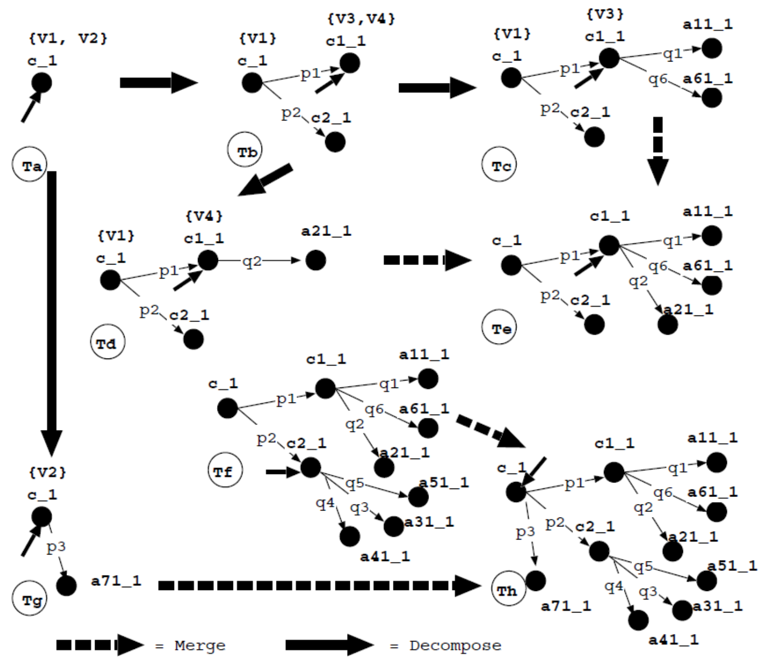


Fig. 3. A configuration example

(Ex. 1209 [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. 1209 [Magro] at 5.)

57. 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 companies need to store, e.g., invoices that have been sent to customers, the names, addresses, and salaries of all employees, etc. Oracle Configurator was marketed as a tool for supporting sales by businesses that run Oracle Applications.

58. A true and accurate copy Oracle Configurator Developer: User's Guide, April 2002, is attached as Exhibit 1216. 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. 1216 [Oracle] at 27-28.)

59. 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. 1216 [Oracle] at 30.)

60. 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.

61. Thus, the exciting research result of 1980 by 2002 had become 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

62. Loomans[8] is a U.S. Patent that was filed on November 18, 2002, and issued on January 18, 2011. It is 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, a 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.

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

[8] Ex. 1205 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.

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. 1205 [Loomans] at 1:57-65, emphasis added.)

64. 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 for 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. 1205 [Loomans] at 11:39-55.)

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

65. Stahl is a printed publication titled "A Customization Approach for

Structured Products in Electronic Shops,"[9] which was published in the proceedings book at the 13th International Bled Electronic Commerce Conference in Bled, Slovania in June 2000. (Ex. 1220 [Schmitt Declaration], at ¶¶3-4.) It is also my understanding that a copy of the proceedings book was provided to all registered conference participants, and that the 13th International Bled Electronic Commerce Conference was open to the public. (*Id.* at ¶4.)

66. Further, it is my understanding that the article titled "A Customization Approach for Structured Products in Electronic Shops," was made publicly

---

[9] Ex. 1206 is a true and accurate copy of: Armin Stahl, Ralph Bergmann, Sascha Schmitt, "A Customization Approach for Structure Products in Electronic Shops," which I understand was included in the proceedings book distributed to attendees at the 13th International Bled Electronic Commerce Conference June 19-21, 2000 (*see also* Ex. 1220 [Schmitt Declaration] at ¶¶3-4.). I also understand that the Bled conference organizers provided a website where the proceedings book for the 13th International Bled Electronic Commerce Conference June 19-21, 2000 could be purchased. (*Id.* at ¶5.) I understand that a version of that website can be found at: https://domino.fov.uni-mb.si/ecomframes.nsf/pages/bled2000. (*Id.*) Exhibit 1206 is a scanned version of the proceedings pages that were photographed and attached to co-author Dr. Sascha Schmitt's declaration. (Ex. 1220 [Schmitt Declaration] at Exhibit A.)

available no later than May 31, 2000, on Dr. Ralph Bergmann's webserver, another co-author of the paper. (Ex. 1219 [Bergmann Declaration] at ¶2.) It is also my understanding that Dr. Bergmann's webserver was publicly available on the Internet and was linked to Dr. Bergmann's research group webpage, which specialized in Artificial Intelligence and Knowledge Based Systems. (*Id.* at ¶3.) And based in part on this linking, it is my understanding that individuals seeking information about Artificial Intelligence and Knowledge Based Systems would have been able to find the research group page, including a link to the article titled "A Customization Approach for Structured Products in Electronic Shops." (*Id.* at ¶4.) I have also compared the version of the article titled "A Customization Approach for Structured Products in Electronic Shops" from the Bled Proceedings book (attached to Dr. Schmitt's declaration) with the version of that same paper attached to Dr. Bergmann's declaration, and have concluded that both versions of the paper are substantively identical, with the differences limited to page number, minor formatting choices, and automatically generated references from within the text to figure numbers. Thus, it is 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 Techinno GmbH." (Ex. 1206 [Stahl] at 12.)

68. Stahl references a 1999 paper "CBR-Works, A State-of-the-Art Shell for Case-Based Application Building,"[10] by Stefan Schulz ("CBR-Works Paper"). I have attached this prior art reference as Exhibit 1217.

69. CBR-Works is capable of modeling features more flexibly 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. 1217 [CBR Works Paper] at 6.) In Loomans, all attributes are of equal weight. The lack of a match in any attribute

[10] Ex. 1217 is a true and accurate copy of: Stefan Schulz, "CBR-Works A State-of-the-Art Shell for Case-Based Application Building," (1999).

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 case to be valid." (Ex. 1217 [CBR Works Paper] at 2-3.)

70. The additional richness of the rules environment used as a starting point for Stahl gives Stahl the ability to handle cases in which an exact match cannot be found for a consumer's preferences. This is explained on Pages 8-9 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. 1206 [Stahl] at 8-9.)

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 10:

> **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. 1206 [Stahl] at 10.)

X. Ground for Challenge

A. Ground 1 – Claims 1-16, 18-29 And 31-43 Are 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 1-16, 18-29, And 31-43 In View Of Loomans, Stahl And The General Knowledge Of A Person Having Ordinary Skill In The Art

a. Claim 1

… ***[1.0] 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:***

72. Loomans discloses configuring an entity by reference to a set of rules partitioned (i.e., divided) via a tree including a parent model and multiple child sub-models:

> 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. 1205 [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. 1205 [Loomans] at 1:57-65, emphasis added.)

73. 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 configuration 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?".

74. Loomans further discloses that one or more configuration queries are

responded to via the disclosed 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**.
>
> **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. 1205 [Loomans] at 4:50-67, emphasis added.)

75. Loomans also describes a computer system 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. 1205 [Loomans] at 16:26-35, emphasis added.)

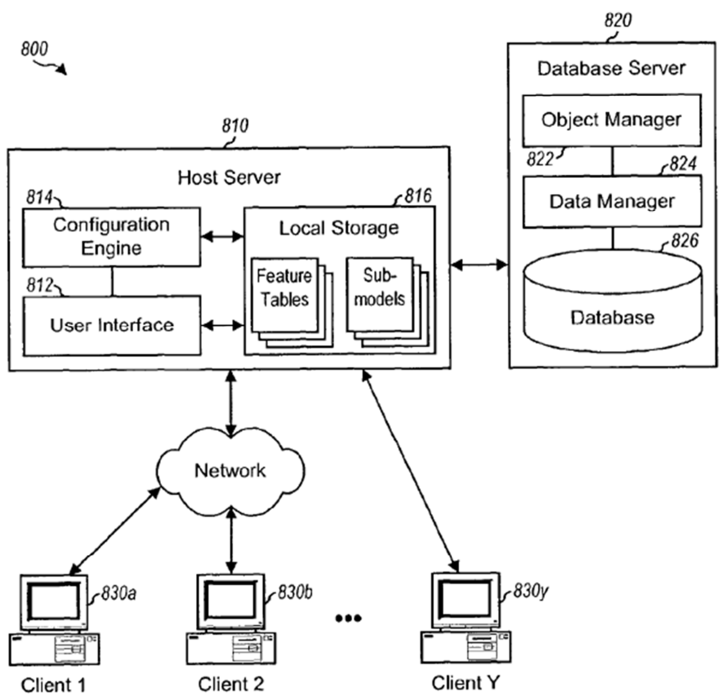


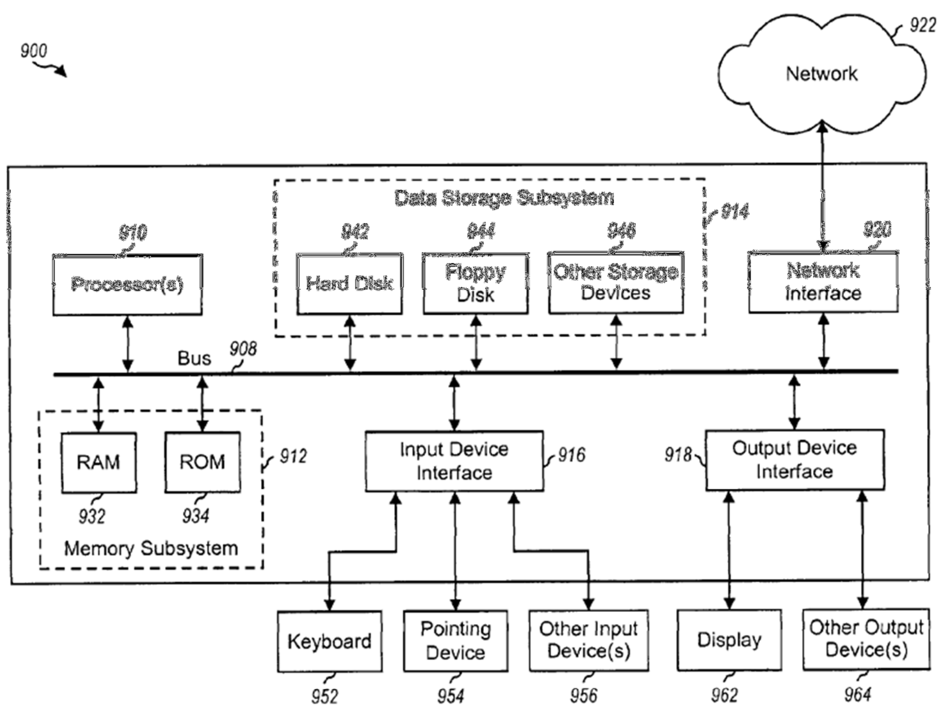
FIG. 8

(Ex. 1205 [Loomans] at Fig. 8.)

76. 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. 1205 [Loomans] at 17:42-50, emphasis added.)



(Ex. 1205 [Loomans] at Fig. 9.)

77. 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 [Figure 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. 1206 [Stahl] at 6, emphasis added.)

78. 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. 1206 [Stahl] at 6.) 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.

79. It is therefore my opinion that a person having ordinary skill in the art would have concluded 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*.

> *… **[1.1] receiving one or more configuration queries representing one or more questions involving parts and part relationships in a configuration of a configurable product;***

80. It is my opinion that Loomans in view of Stahl discloses receiving one or more configuration queries representing one or more questions involving parts and part relationships in a configuration of a configurable product.

81. 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. 1204 ['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.*)

82. 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. 1206 [Stahl] at 6.) Stahl discloses Figure 3 (below) as an example query with a "root note as our actual problem." (*Id.*)

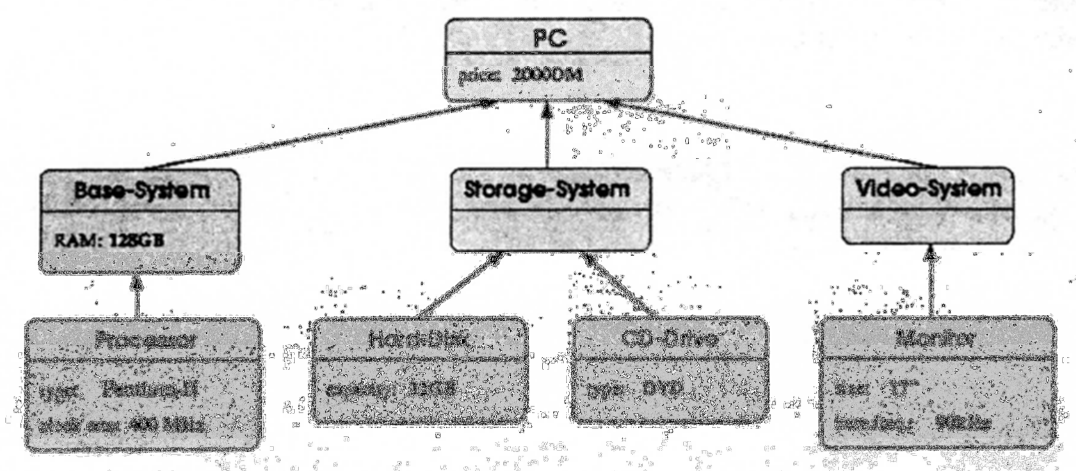


Figure 3: An Example Query

(*Id.* at Figure 3.)

83. 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 6.) 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.

84. 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. 1205 [Loomans] at 1:57-65, emphasis added.)

85. 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. 1205 [Loomans] at 2:25-38, emphasis added.)

86. 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. 1205 [Loomans] at 4:59-67, emphasis added.)

87. 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. 1205 [Loomans] at 4:7-16, emphasis added.)

88. 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?".

89. Thus, a person having ordinary skill in the art would have understood that the configuration process described in Loomans in view of Stahl would have

included one or more queries to determine if the proposed combination of parts is compatible and therefore forms a valid configuration. Accordingly, it is my opinion that Loomans in view of Stahl discloses *receiving one or more configuration queries representing one or more questions involving parts and part relationships in a configuration of a configurable product*.

… ***[1.2a] and performing with the computer system:***

90. Loomans describes a computer system 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. 1205 [Loomans] at 16:26-35, emphasis added.)

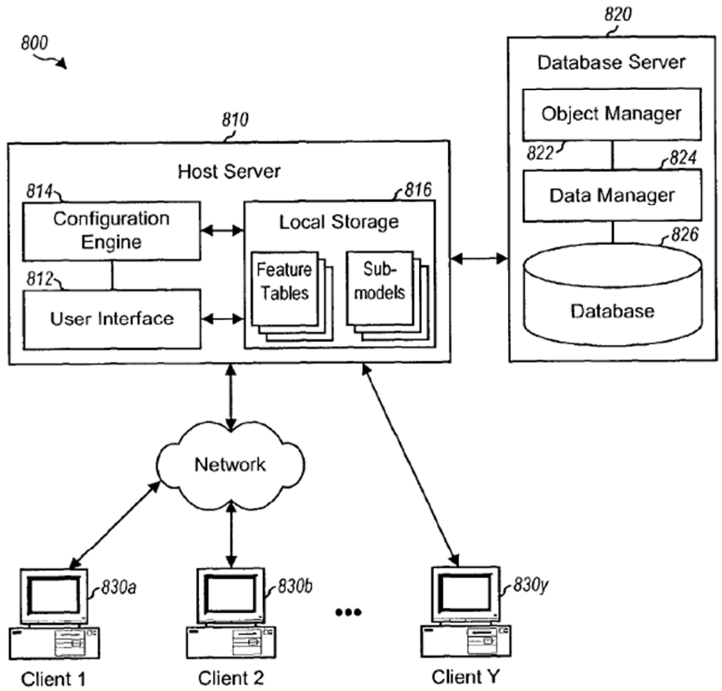


FIG. 8

(Ex. 1205 [Loomans] at Fig. 8.)

91. 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. 1205 [Loomans] at 17:42-50, emphasis added.)

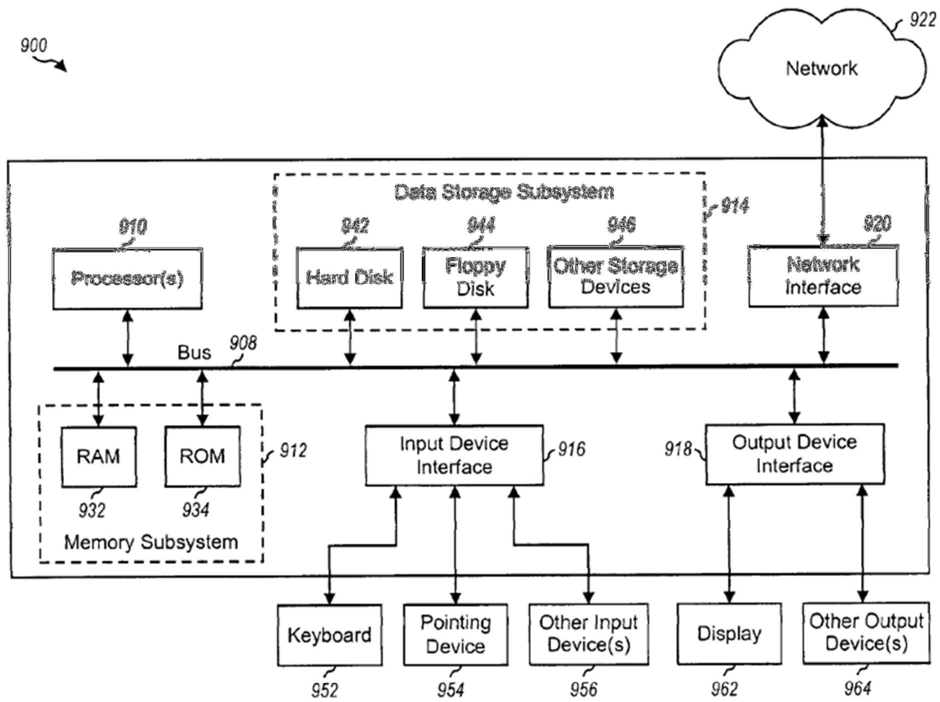


FIG. 9

(Ex. 1205 [Loomans] at Fig. 9.)

92. 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 [Figure 5]) and the result presentation**.

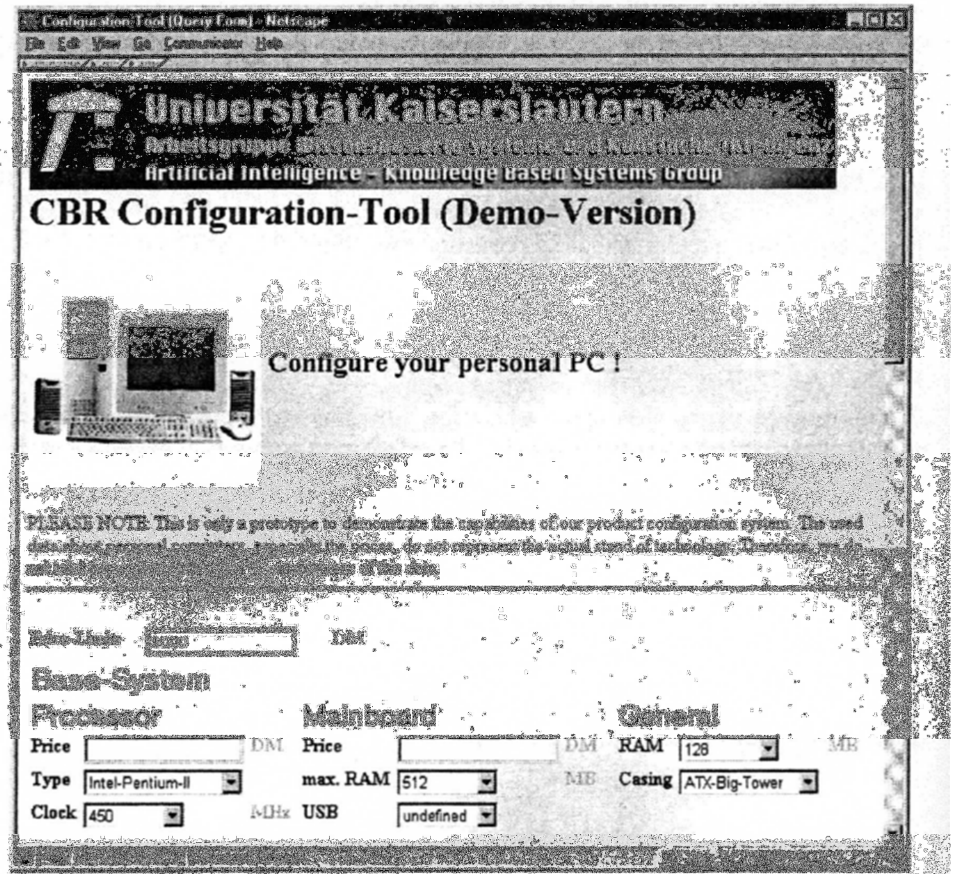


Figure 5: Implemented Demo Version

(Ex. 1206 [Stahl] at 12 and Figure 5.)

93. 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.

94. Therefore, it is my opinion that Loomans in view of Stahl discloses performing steps [1.3]-[1.8] (below) with the *computer system*.

… ***[1.2b] dividing one or more configuration queries into multiple configuration sub-queries, wherein the multiple configuration sub-queries represent the one or more configuration queries;***

95. 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 [Figure 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. 1206 [Stahl] at 6, emphasis added.)

96. 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. 1206 [Stahl] at 6, emphasis added; note that a person having ordinary skill in the art would have understood that the "we" in "we can interpret" refers to the authors and human readers and "interpret" is used to mean "understand".)

97. Further, a person having ordinary skill in the art would have understood that Stahl's disclosure that there are "different leaf nodes of the query" indicates that the query has been divided. First, if Stahl included a system capable of processing a solidary query in a single operation there would be no need to describe "different leaf nodes of the query." Instead, each leaf node represents a separate – divided out – portion of the query – *i.e.*, a sub-query. Indeed, that disclosure explains that the divided out "leaf nodes" represent (i.e., can be interpreted or understood as) sub-problems, which are solved as a subtask in solving the overall problem. A second reason why a person of ordinary skill would understand that a query in the Stahl system would be divided is that conventional computer systems are capable of processing only in small steps. Thus, anything other than the simplest request, e.g., adding two numbers, must be divided into pieces for processing. In the conventional RDBMS world, for example, a query will be in the SQL language and the SQL parser and optimizer will divide it into

concrete processing steps.

98. Further, Stahl's disclosure that "we can interpret the different leaf nodes of the query as sub-problems that must be solved to solve the overall problem," indicates that the sub-problems (i.e., sub-queries) together form the overall problem (i.e., form the overall query). In this case, the overall problem – i.e., query – is "searching a PC with a set of special properties [e.g., a certain price, hard-disk size, CD-Drive type]," which is represented in Figure 3 as the root node "PC, price: 2000DM." (Ex. 1206 [Stahl] at 6 and Figure 3.) Stahl teaches that the sub-problems, and ultimately the solutions to the sub-problems, are combined to form "a final solution for the overall configuration problem." (*Id.* at 6.) That is, the sub-queries (sub-problems) are solved to provide sub-solutions, which form the overall solution to the query (overall configuration problem). A person having ordinary skill in the art would have understood that in order for the sub-solutions to be combinable into a solution to the overall configuration problem, the underlying sub-problems would likewise need to collectively represent the query answered by the final solution to the overall configuration problem answers. This idea is depicted pictorially below in my enhancement of Figure 3 of Stahl.

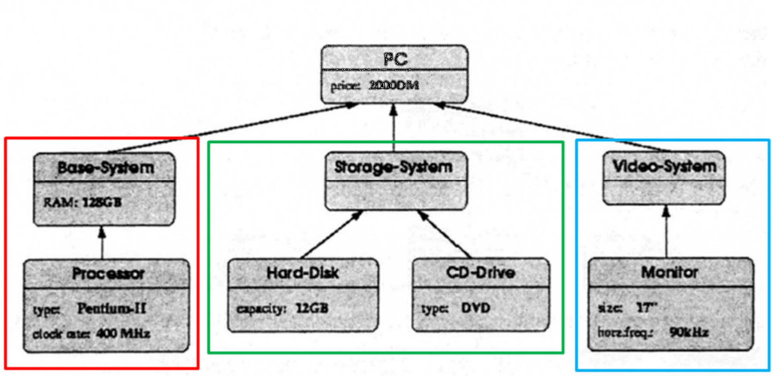


Figure 3: An Example Query

Base-System sub-problem + Storage System sub-problem + Video-System sub-problem = Final Solution to Overall PC Problem (Query)

(Ex. 1206 [Stahl] at Figure 3)

99. 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. 1205 [Loomans] at 4:48-58, emphasis added.)

100. Loomans next discloses configuring and 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. 1205 [Loomans] at 4:59-67, emphasis added.)

101. 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?".

102. 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 any particular portion of the query. This would negate the feature of Loomans, namely to sub-configure an entity at the component level. (Ex. 1205 [Loomans] at 3:26-27.)

103. 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. Indeed, 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.

104. 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.

… ***[1.3] processing each sub-query using at least one configuration sub-model per sub-query,***

105. 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. 1205 [Loomans] at 4:48-58, emphasis added.)

106. 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. 1205 [Loomans] at 4:59-67, emphasis added.)

107. 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. 1205 [Loomans] at 2:25-38, emphasis added.)

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

109. Further, Stahl discloses viewing a query that has been broken down into a tree (with "leaf nodes") as a collection 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. 1206 [Stahl] at 6, emphasis added.)

110. 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. 1205 [Loomans] at Abstract.) And Stahl discloses that each sub-problem represents a "part-query" of a component. (Ex. 1206 [Stahl] at 6.) 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 person having ordinary skill in the art 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 person having ordinary skill in the art how to process and solve the sub-problems taught in Stahl for specific features.

111. 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.

… ***[1.4] 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***

112. Loomans discloses a top-level entity that is represented by a parent model and a collection of sub-models, referred to as "child sub-models" of the parent.

> 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. 1205 [Loomans] at 1:57-65, emphasis added.)

As with similar structures in most of the prior art, this is a natural way of organizing configuration information for a multi-component physical system.

113. Loomans further discloses that each sub-configurable option of a top entity can be configured via a child sub-model.

> 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.

> 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. 1205 [Loomans] at 4:50-67, emphasis added.)

114. Note that the primary distinction between a "parent" versus a "child" model is their respective locations within the tree. Loomans explicitly discloses that a parent model may function as a sub-model and vice versa: "each child sub-model may be a parent model that includes one or more child sub-models." (Ex. 1205 [Loomans] at 5:39-41.) The topmost-level parent model of Loomans may be as simple as a collection of pointers to child sub-models or it may contain some configurable options. These two options are explicitly disclosed:

> In general, if multiple levels of sub-configuration are supported, each sub-model may be a parent model that includes selectable and/or sub-configurable options. One local feature table may be provided for each selectable option in the sub-model, and one sub-model mapping set and its associated sub-models may be provided for each sub-configurable option in the sub-model. Each sub-model may thus be a

> model that is independent of the parent model and other sub-models, and may be individually and separately configured and validated.

(Ex. 1205 [Loomans] at 7:5-13)

115. Loomans discloses that the process of moving configuration rules into sub-models turns the top-level entity into a "simplified top-level entity." (*Id.* at 1:57-58) Thus, the top-level parent model corresponding to the top-level entity can no longer perform the complete configuration task. Although it still sits at the top of a tree data structure, to the extent that it can perform any configuration at all, e.g., via a "local feature table", it has effectively become a "sub-model" itself due to the fact that it is configuring only a portion of the real-world entity.

116. A person having ordinary skill in the art would have understood that the "simplified top-level entity" would at most function as a sub-level entity once other sub-level entities have been divided out from the top-level entity for some component from the original top-level entity. Indeed, by definition, once sub-level entities have been divided out, the remaining portions of the top-level entity no longer represent the complete top-level entity prior to creating the sub-models, i.e., creating the sub entities. A person having ordinary skill in the art would have understood that any configuration data structure according to Loomans, regardless of the number of levels of sub-models chosen by the user, would include all of the information necessary to fully configure a product: "Each sub-model is provided

with sufficient information needed to configure and validate the option represented by that 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 (Loomans at 6:14-22); "Sub-configuration thus allows a complicated entity to be configured in smaller portions and incrementally, one component at a time." (Loomans at 11:19-21). Thus the sub-models within a Loomans data structure would collectively model the configurable product, as required by the claim.

117. 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. 1205 [Loomans] at 3:41-49, emphasis added.)

118. 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 a 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 storages 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. 1205 [Loomans] at 6:30-46, emphasis added.)

119. Further, Loomans discloses that mapped features may be used to enumerate relationships between various features in the sub-model.

> **In one embodiment, the mapped features 440 in a sub-model 430 may be used to enumerate relationships between the various features in the sub-model**. For example, sub-model 430a for the ABC storage device indicates that the power rating for the 20 GB device is 40 Amp, which implies that a power supply of 40 Amp or more is needed. Similarly, power supplies of 60 Amp (or more) and 80 Amp (or more) are needed for the 40 GB and 80 GB devices, respectively. Thus, in the configuration of the ABC sub-model, a

> value (e.g., 40 Amp) that has been selected for the Power Supply option G at the parent model may be passed down to the sub-model. **If a particular storage amount is selected (e.g., 40 GB), then the corresponding power rating is determined (e.g., 60 Amp) and compared against the selected power supply (e.g., 40 Amp). In this case, since the selected power supply is inadequate for the selected storage amount, feedback may be provided immediately so that the user may select either another storage amount (e.g., 20 GB) or another power supply (e.g., 60 Amp or more).** The tables (or feature mapping set) for the mapped features may thus be used to enumerate relationships that are used to validate configuration at the sub-model. The feature map values may be returned to the parent model upon leaving the sub-model.

(Ex. 1205 [Loomans] at 8:44-67, emphasis added.)

120. 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 and the compatibility of those part choices with other part choices (e.g., the compatibility between types of storage and the selected power supply). 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.

121. 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.

> … ***[1.5] 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;***

122. 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. 1205 [Loomans] at 1:57-65, emphasis added.)

123. 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. Each of the sub-level entities represents a component of the top-level entity. 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).

124. 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. 1205 [Loomans] at 5:10-18, emphasis added.)

125. For example, Loomans discloses configuring and 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. 1205 [Loomans] at 6:11-22 and 11:19-27, emphasis added.)

126. A person having ordinary skill in the art would have understood that based on this teaching Loomans discloses that feature-based sub-models are used for the configuration and validation of the parent model which is representative of top-level entity configurable products. 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?".

127. 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. 1206 [Stahl] at 6, emphasis added.)

128. 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).

129. 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.

130. 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.

> … ***[1.6] 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;***

131. 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. 1205 [Loomans] at 2:25-38, emphasis added.)

132. 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. 1205 [Loomans] at 6:11-22 and 11:19-27, emphasis added.)

133. 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. 1205 [Loomans] at 4:59-67, emphasis added.)

134. 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?".

135. 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. 1205 [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.

136. 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.

137. Further, Stahl discloses that a query may be considered a collection 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. 1206 [Stahl] at 6, emphasis added.)

138. 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).

139. Accordingly, it is my opinion that a person having ordinary skill in the art would have understood 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*.

… ***[1.7] and providing the response to the one or more configuration queries as data for display by a display device.***

140. 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. 1205 [Loomans] at 16:36-49, emphasis added.)

141. 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. 1205 [Loomans] at 16:18-22, emphasis added.)

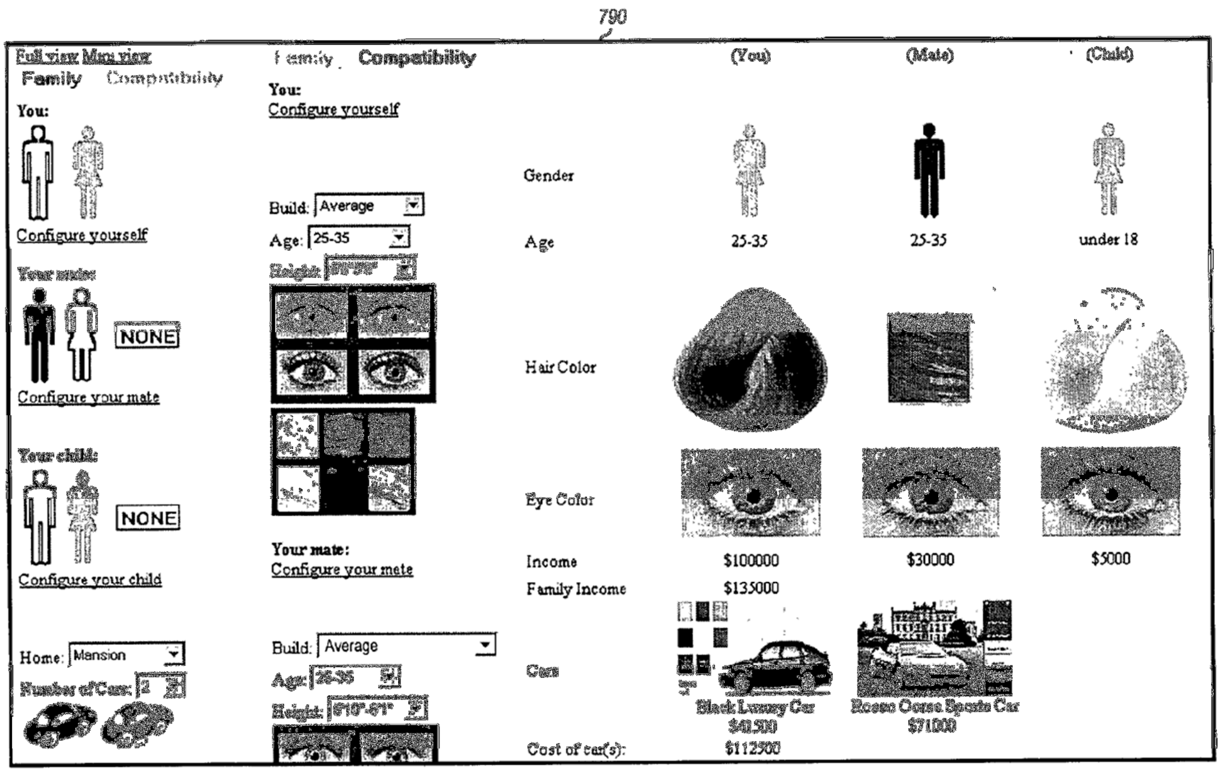


FIG. 7C

(*Id.* at Fig. 7C)

142. 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 [Figure 5]) and the result presentation**.

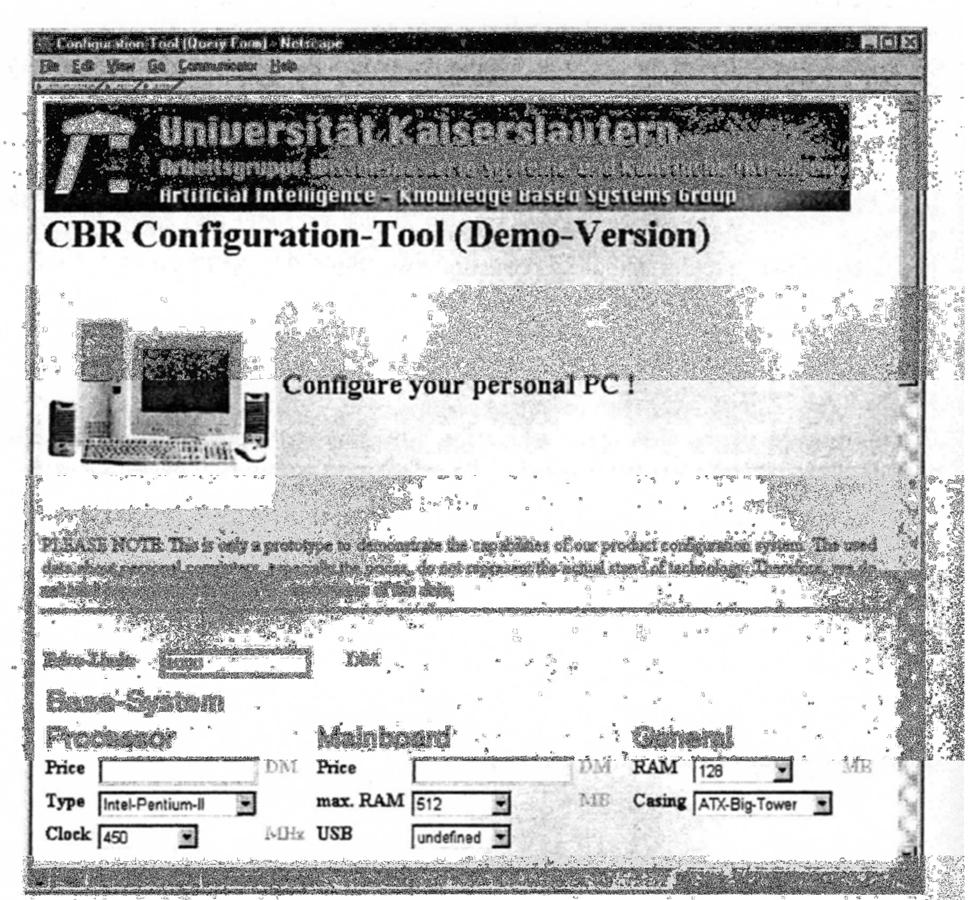


Figure 5: Implemented Demo Version

(Ex. 1206 [Stahl] at 12 and Figure 5.)

143. 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. Reasons to Combine Loomans with Stahl

144. 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. 1205 [Loomans] at 1:57-65.)

145. 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. 1206 [Stahl] at 6.) 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.

146. Although both Loomans and Stahl are addressing a common problem in the prior art, the focus of Loomans and Stahl are a bit 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. 1205 [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. 1206 [Stahl] at 9); "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.*), 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, including the method of decomposing configuration models into sub-models, with the rule evaluation system described in Stahl.

147. 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. 1205 [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.

148. 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. If a configuration is found in a configuration table it is valid; if not found in the configuration table it is not valid. (Ex. 1205 [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. The concern of customers calling up to ask "Why can't I place an order?" would have motivated a

person having ordinary skill in the art to use the more flexible rule evaluation system of Stahl at least in whichever sub-model was proving difficult to satisfy.

149. 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 having 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 2

… ***[2.0] The method of claim 1 wherein the one or more configuration queries relate to a configuration completion problem.***

150. The '057 Patent describes an example of a configuration completion problem as the circumstance when a part is not present from every required part group. (Ex. 1201 ['057 Patent] at 10:38-45.) This example is present in the prior

art as described below.

151. Stahl discloses part-queries, which are used to determine if the product to be configured does not include required components. For example, Stahl points to configuration of a PC, which does not include a modem, but the particular requested configuration requires a modem. Stahl describes a component with a missing feature as a "weak part."

> **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. . . 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. 1206 [Stahl] at 8-9, emphasis added.)

152. Thus, Stahl discloses circumstances, such as the PC example described above, where a part (modem) is not present from every required group (Communication Model Group) – in other words Stahl discloses a *configuration completion problem*.

153. The process described above is also disclosed at a high-level in Figure 4 of Stahl (see below).

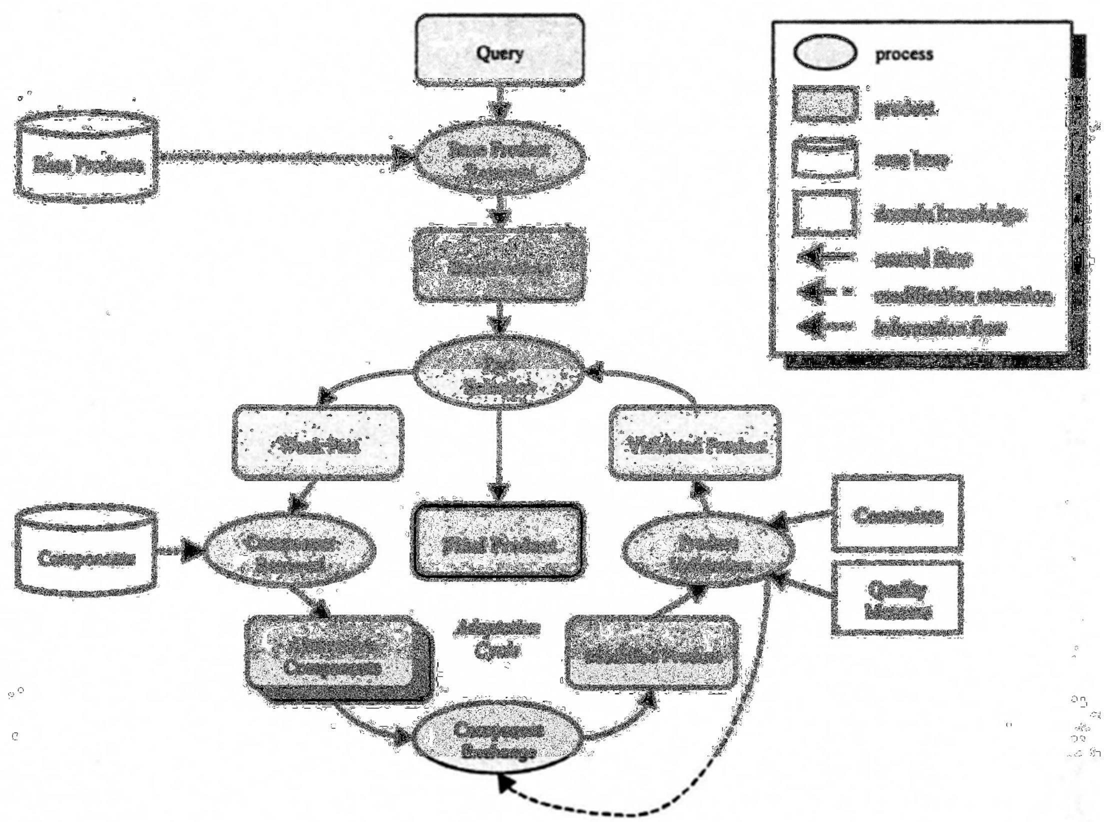


Figure 4: The Configuration Process

(Ex. 1206 [Stahl] at Figure 4.)

154. Stahl further describes that "[a]t 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." (Ex. 1206 [Stahl] at 8.) Stahl then discloses that "[t]his can be done by an examination of the local similarities between the different product parts and the related part-queries." (*Id.*) Thus, Stahl discloses a query at the beginning of the adaption cycle, which seeks to determine if a base product (e.g., PC without modem in example above) fulfills the requirements of the

configuration, e.g., whether that product includes all of the required parts. Thus, the query disclosed in Stahl is related to whether a part (e.g., modem) is not present from every required part group, and therefore is related to a *configuration completion problem.*

155. Further, as shown in Figure 5 below, Loomans discloses a sub-configuration process in which a sub-model configured.

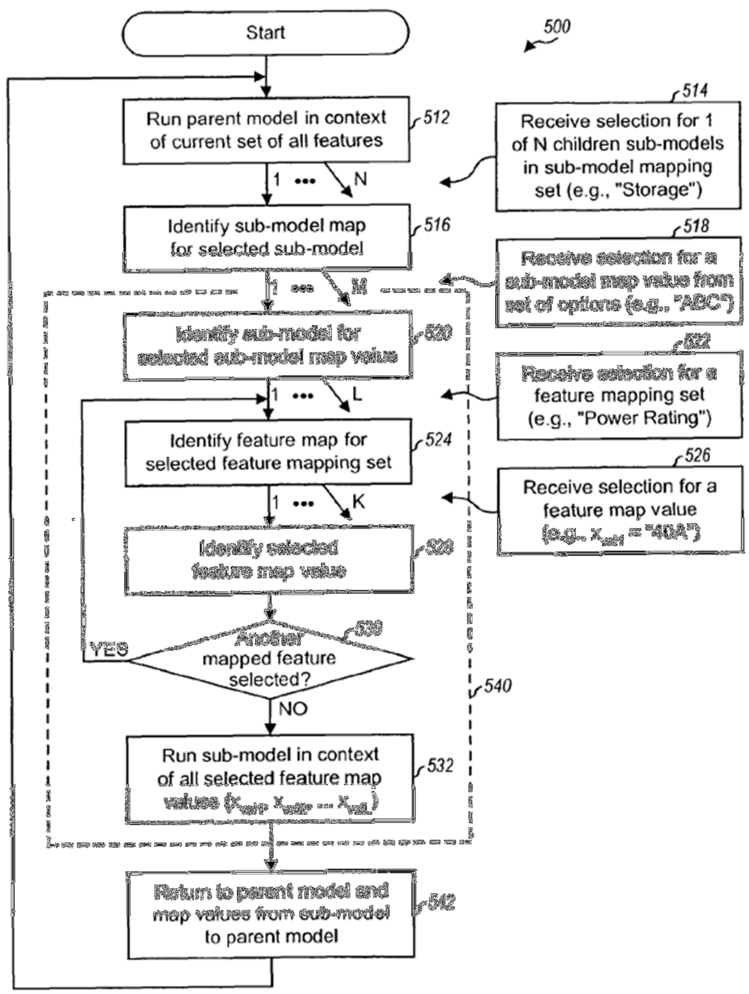


FIG. 5

(Ex. 1205 [Loomans] at Fig. 5.)

156. Loomans further discloses the validation process depicted in block

540 of Figure 5 above.

> The steps within block 540 perform the sub-configuration for the selected sub-model. **If the configuration for the sub-model is not valid, feedback can be provided at the sub-level (e.g., via a warning or error message), and the process may return to the sub-model at step 520 (not shown in FIG. 5).** The invalid configuration for the sub-model can be thus modified at this point, e.g., by selecting another value for each feature that causes the invalid configuration. The sub-model may then be re-executed with the new values. The steps within block 540 may thus be repeated as many times as necessary or desired.

(Ex. 1205 [Loomans] at 10:54-64, emphasis added.)

157. As discussed above, 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?".

158. Thus, a person having ordinary skill in the art would have understood that the configuration (and validation) process for sub-models described in Figure

5 involves *one or more configuration queries relate to a configuration completion problem*. Indeed, when the configuration for the sub-model is "not valid," the corresponding query to that sub-model would relate to a *configuration completion problem*.

159. Accordingly, it is my opinion that Loomans in view of Stahl discloses that the one or more configuration queries relate to a configuration completion problem.

d. Claim 3

… ***[3.0] The method of claim 1 further comprising: processing each sub-query using multiple configuration sub-models per sub-query.***

160. Loomans discloses that "[w]ith sub-configuration, shared options (e.g., Drive Bays 1, 2, and 3) may be extracted and modeled with a single sub-model or a set of sub-models. Each sub-model may be referenced and instantiated as needed by any of the common options." (Ex. 1205 at 5:20-24.) Hence, Loomans teaches that multiple sub-models can be used for the configuration of related options (i.e., parts) of a configurable product.

161. As discussed above, 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?"

162. Accordingly, it is my opinion that Loomans in view of Stahl discloses processing each sub-query using multiple configuration sub-models per sub-query.

e. Claim 4

> … ***[4.0] The method of claim 1 wherein the one or more configuration queries relate to a configuration validation problem and processing one or more configuration queries further comprises: processing at least one of the sub-queries using different configuration sub-models until a configuration validation answer can be determined.***

163. Loomans discloses that "[w]ith sub-configuration, shared options (e.g., Drive Bays 1, 2, and 3) may be extracted and modeled with a single sub-model or a set of sub-models. Each sub-model may be referenced and instantiated as needed by any of the common options." (Ex. 1205 at 5:20-24.)

164. Loomans further discloses using multiple sub-models to configure the storage device of a computer system.

> Each element of mapping set 410 is then either associated with a sub-model map or a feature table. For example, option F (Storage) may be associated with a sub-model map 420 that includes the two available

> types of storages devices, ABC and XYZ. **Sub-model map 420 is further associated with two sub-models 430a and 430b, one for each type of storage device**.

(Ex. 1205 [Loomans] at 7:53-59, emphasis added.)

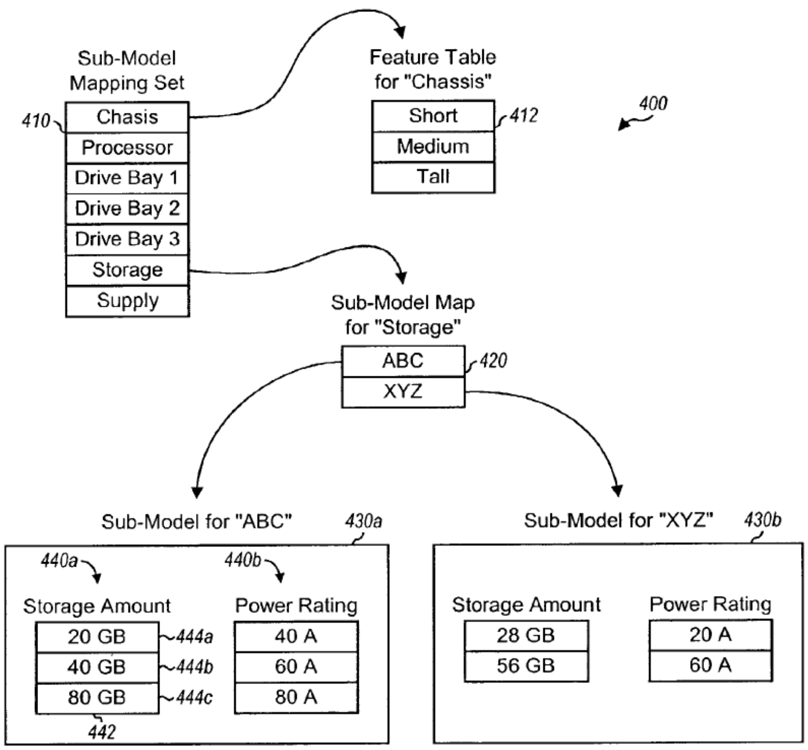


FIG. 4

(Ex. 1205 [Loomans] at Fig. 4.)

165. Dependency-directed backtracking is a familiar concept in the world of rule-based or constraint-based systems, which is the general family of software that includes configuration systems. The idea was introduced in a September 1976 MIT Artificial Intelligence Laboratory Memo, "Forward Reasoning and Dependency-Directed Backtracking in a System for Computer-Aided Circuit

Analysis," by Stallman and Sussman (attached as Exhibit 1218).[11] Page 3 of this document explains dependency-directed backtracking: "The system notes a contradiction when it attempts to solve an impossible algebraic relationship, or when discovers [*sic*] that a transistor's operating point is not within the possible range for its assume region. The antecedents of the contradictory facts are scanned to find which nonlinear device state guess (more generally, the backtrackable choicepoints) are relevant; ARS never tries that combination of guesses again. A short list of relevant choicepoints eliminates from consideration a large number of combinations of answers to all other (irrelevant) choices." In short, if the system goes down a branch of the configuration tree that isn't fruitful it will remember not to try that branch again. This would work out to be functionally equivalent to the claimed "try a different sub-model if one sub-model yields only the answer 'invalid configuration'." (Note that dependency-directed background quickly became part of the standard undergraduate computer science artificial intelligence class and is covered in Chapter 14 of *Artificial Intelligence, Third Edition* (Winston 1992; Addison-Wesley; May 1993 reprint attached as Ex. 1221).)

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

[11] Ex. 1218 is a true and accurate copy of the article titled "Forward Reasoning and Dependency-Directed Backtracking in a System for Computer-Aided Circuit Analysis," by Stallman and Sussman in September 1976.

multiple sub-models could be used to validate the sub-configurable option until a validation answer is determined. Indeed, if the initial sub-model does not provide a validation answer, applying additional sub-models (where necessary) to confirm the validity of a configuration would increase the chances of creating a configuration for a buildable product. On the other hand, if only one sub-model is used, and that sub-model does not provide a validation answer, the process would effectively stall without completion of a configurable product. Thus, a person having ordinary skill in the art would have understood that the set of sub-models described in Loomans could have been used to validate potential configuration to ensure that a valid configuration is created.

167. Accordingly, it is my opinion that Loomans in view of the knowledge of a person of ordinary skill in the art disclose that the one or more configuration queries relate to a configuration validation problem and processing one or more configuration queries further comprises: processing at least one of the sub-queries using different configuration sub-models until a configuration validation answer can be determined.

f. Claim 5

… ***[5.0] The method of claim 1 wherein the data collectively included in the configuration sub-models provides a response for each of the sub-queries being processed.***

168. 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. 1205 [Loomans] at 4:48-58, emphasis added.)

169. Thus, a person having ordinary skill in the art would have understood that Loomans discloses that the described child sub-models of Loomans collectively include the data that makes the top-level entity parent model.

170. Loomans next discloses configuring the top entity (with a corresponding parent model) by configuring sub-configurable component via sub-models.

> 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. 1205 [Loomans] at 2:25-38, emphasis added.)

171. As discussed above, 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?". Thus, a person having ordinary skill in the art would have understood that Loomans discloses providing a response to said query by selecting and validating particular components based on associated sub-models. To do this, a person having ordinary skill in the art would have understood that the sub-models disclosed in Loomans

would need to have the data necessary to answer any sub-queries processed in view of the sub-models.

172. Further, Stahl discloses understanding a query as a collection 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. 1206 [Stahl] at 6, emphasis added.)

173. As described above, 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. 1206 [Stahl] at 5.)

174. 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 further understood that the sub-models described in Loomans could be partitioned in such a way to provide the solutions (i.e., answers) to each part-query

raised in each sub-problem disclosed in Stahl. Indeed, Loomans discloses that "each sub-configurable component is configured via one of a number of sub-models." (Ex. 1205 [Loomans] at Abstract.) And Stahl discloses sub-problems, which represent a form of sub-query that Stahl discloses are solved with "suitable sub-solutions" (i.e., answers) by determining the suitable parts for a particular "part-query." (Ex. 1206 [Stahl] at 6.)

175. Loomans further discloses that "the required information 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.) This would be required in order for the configuration systems in both Loomans and Stahl to validate and configure the requested product. Thus, a person having ordinary skill in the art would have understood that the sub-models disclosed in Loomans would include all of the data needed to respond to the sub-problems (i.e., sub-queries) described in Stahl to ensure that each sub-configurable component can be validated and configured for combination and completion of the product.

176. Accordingly, it is my opinion that Loomans in view of Stahl discloses wherein the data collectively included in the configuration sub-models provides a response for each of the sub-queries being processed.

g. Claim 6

… ***[6.0] The method of claim 1 wherein at least two sub-queries include overlapping information.***

177. Stahl discloses that sub-problems (i.e., sub-queries) can include dependencies between different sub-problems or their solutions. Stahl discloses that these dependencies can occur as constraints between different components.

> **If we decompose a configuration problem into the sub-problems of finding suitable components it is not sufficient to handle these sub-problems absolutely isolated. The reason are dependencies between the different sub-problems or their solutions respectively. In the described application domain such dependencies occur in form of technical constraints between the different components. For example, it is impossible to combine every kind of processor with any mainboard model because the respective interfaces must fit together**. To be able to handle these technical restrictions during the configuration process, the domain model must include a formal description of the existing constraints. For our approach, we suppose the existence of a special constraint system that is able to check whether the constraints of a given configuration are fulfilled or not.

(Ex. 1206 [Stahl] at 7, emphasis added.)

178. Stahl provides an additional example of "constraints between the components" (each "component" in Stahl corresponds to a sub-model or sub-problem) in Section 3.3: "if we want to install a SCSI-hard-disk we have perhaps

also to replace the IO-controller." (Ex. 1206 [Stahl] at 10.) A person having ordinary skill in the art would have understood that "SCSI" refers to the standard "Small Computer System Interface" and that this is a constraint requiring a controller card (or motherboard) within a PC having an interface that is compatible with a disk drive. The standard SCSI system of the late 1990s had a 16-bit data bus and a 68-pin connector on the disk drive, or an 80-pin SCA (Single Connector Attachment) connector that included power (I personally purchased servers around the time of Stahl's publication with SCSI disk drives employing the SCA standard). Both types of connectors had a trapezoidal shape. The competitive interface standard for personal computer hard drives was "IDE" (Integrated Drive Electronics; also sometimes called "Parallel ATA"). This employed a rectangular 40-pin connector. It is not physically or electrically possible to plug an IDE connector into a SCSI drive nor a SCSI connector into an IDE disk drive.

179. Thus, Stahl discloses that multiple sub-problems can share information in the form of dependencies, such as constraints. In other words, the disclosed sub-problems include overlapping information.

180. Loomans further discloses using a table of mapped features that are used to enumerate relationships that are used to validate configuration at the model.

> In one embodiment, the mapped features 440 in a sub-model 430 may be used to enumerate relationships between the various features in the sub-model. For example, sub-model 430a for the ABC storage device indicates that the power rating for the 20 GB device is 40 Amp, which implies that a power supply of 40 Amp or more is needed. Similarly, power supplies of 60 Amp (or more) and 80 Amp (or more) are needed for the 40 GB and 80 GB devices, respectively. Thus, in the configuration of the ABC sub-model, a value (e.g., 40 Amp) that has been selected for the Power Supply option G at the parent model may be passed down to the sub-model. If a particular storage amount is selected (e.g., 40 GB), then the corresponding power rating is determined (e.g., 60 Amp) and compared against the selected power supply (e.g., 40 Amp). **In this case, since the selected power supply is inadequate for the selected storage amount, feedback may be provided immediately so that the user may select either another storage amount (e.g., 20 GB) or another power supply (e.g., 60 Amp or more). The tables (or feature mapping set) for the mapped features may thus be used to enumerate relationships that are used to validate configuration at the sub-model**. The feature map values may be returned to the parent model upon leaving the sub-model.

(Ex. 1205 [Loomans] at 8:45-67, emphasis added.)

181. In the example above, when an inadequate power supply is selected to be paired with a certain storage amount, feedback may be provided so that the user may select a different storage amount or a different power supply. A person having

ordinary skill in the art would have understood that the second selection, which would function as a query (i.e., sub-query), would include information about the first selection (i.e., overlapping information) to ensure that the same inadequate selection is not made twice.

182. Accordingly, a person having ordinary skill in the art would have understood that Loomans in view of Stahl discloses that *at least two sub-queries include overlapping information*.

h. Claim 7

… ***[7.0] The method of claim 1 further comprising: dividing a consolidated configuration model into the multiple configuration sub-models in accordance with a predetermined data structure; wherein at least one of the configuration queries into multiple configuration sub-queries further comprises dividing the sub-queries in accordance with the sub-model structure.***

183. The '057 patent specification does not describe a "predetermined data structure." However, the use of a "data structure" is described as a "design choice" (presumably to be made by a person of ordinary skill). (Ex. 1201, ['057 Patent] at 11:5-9.) Claim 8 informs a reader's understanding of the "predetermined data structure" of Claim 7 in that an example is "a data structure divided along configuration model part groups." The structures illustrated in Figure 6 are conventional tree data structures. Thus a person of ordinary skill would have

understood that a "predetermined data structure" is one whose basic shape is established prior to a specific configuration model being divided. That would include, for example, the tree data structure that has been conventionally used in the prior art.

184. Loomans illustrates the use of a predetermined data structure in Figures 3A and 3B, described as showing "a data modeling structure and supports sub-configuration using mapped features;" (Ex. 1205 [Loomans] at 2:65-66.) Loomans teaches that, regardless of the specific type of product being configured, the same data structure can be used, though of course it might have more levels or more branches and leaves for a more complicated product.

185. Loomans discloses dividing a top-entity (e.g., Computer System), which is represented as a parent model, into sub-level entities (e.g., Drive Bays 1, 2 and 3 of the Computer System), which are represented as 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. 1205 [Loomans] at 4:48-58, emphasis added.)

186. Loomans further discloses a sub-configuration process where mapped features are passed between the parent model and the child 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.

(Ex. 1205 [Loomans] at 6:11-22, emphasis added.)

187. Thus, Loomans discloses dividing the parent model into sub-models in a way that allows mapped features to be passed between the parent model and the sub-model. A person having ordinary skill in the art would have understood that this means that the sub-models would have been divided with a predetermined data structure (with predetermined interfaces for communicating data) to ensure that sub-model could receive the appropriate mapped feature data from the parent

model.

188. A person having ordinary skill in the art would have understood that the child sub-models disclosed in Loomans would need to be partitioned out in a predetermined way to ensure that the sub-models were related to particular components of the top level entity, e.g., through the Sub-Model maps illustrated in Fig. 3A.

189. As discussed above, 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 configuration 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?".

190. A person having ordinary skill in the art would have understood that each query would have been divided out to correspond with a particular sub-model to ensure that the proper information is being queried from the proper sub-model for a particular mapped feature.

191. Further, as discussed above, Stahl discloses understanding a query as a collection 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. 1206 [Stahl] at 6, emphasis added.)

192. For similar reasons as discussed with regards to the sub-models in Loomans, a person having ordinary skill in the art would have understood that dividing the sub-queries to correspond with the subject matter in the sub-models would lead to increased efficiencies. Indeed, such a division would limit the information sought by each sub-query to that included in its corresponding sub-model. This 1-to-1 correlation would eliminate the wasting of processing resources by avoiding queries broader than the information contained in the sub-model.

193. Accordingly, it is my opinion that Loomans in view of Stahl discloses dividing a consolidated configuration model into the multiple configuration sub-models in accordance with a predetermined data structure; wherein at least one of the configuration queries into multiple configuration sub-queries further comprises dividing the sub-queries in accordance with the sub-model structure.

i. Claim 8

> … ***[8.0] The method of claim 7 wherein the predetermined data structure comprises a data structure divided along configuration model part groups, wherein the part groups are a collection of related parts.***

194. An example of a part group is "an 'Engines' group [that] might contain the parts 'V6 engine' and '4 cylinder engine'." (Ex. 1201 ['057 Patent] at 1:37-38.)

195. Loomans discloses that "[t]he sub-models reduce the amount of redundancy by **localizing data and logic needed to represent the common options**, and further simplify the data structure and logic for the top-level entity." (Ex. 1205 [Loomans] at 5:24-27, emphasis added.) Finally, Loomans discloses that sub-model maps can be grouped based on the type of part:

> 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 storages 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).

(Ex. 1205 [Loomans] at 6:35-42.) The alternative of three different types of disk drives, only one of which can fit into a drive bay, is analogous to the example part group in the preferred embodiment of the '057 Patent, an alternative of two different types of engines, only one of which can fit into a car's engine compartment.

196. Further, as discussed above, Loomans discloses dividing a top-entity (e.g., Computer System), which is represented as a parent model, into sub-level entities (e.g., Drive Bays 1, 2 and 3 of the Computer System), which are represented as child sub-models. In the example in Loomans, one such sub-model relates the part group for the computer drive bays.

> 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. 1205 [Loomans] at 4:48-58, emphasis added.)

197. Thus, Loomans discloses dividing sub-models in such a way that data for common options, e.g., common parts, are localized together.

198. Accordingly, it is my opinion that Loomans in view of Stahl discloses that the predetermined data structure comprises a data structure divided along

configuration model part groups, wherein the part groups are a collection of related parts.

j. Claim 9

… ***[9.0] The method of claim 1 wherein generating a response to the one or more configuration queries based upon the processed one or more configuration queries and the configuration sub-models further comprises: generating a response for each processed configuration sub-model; and combining each response for each processed configuration sub-model to generate the answer.***

199. 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. 1205 [Loomans] at 2:25-38, emphasis added.)

200. Loomans further discloses querying and entering 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. 1205 [Loomans] at 4:59-67, emphasis added.)

201. Finally Loomans discloses:

> 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**.

(Ex. 1205 [Loomans] at 5:10-16, emphasis added.)

202. Thus, Loomans discloses configuring each sub-configurable component via sub-model. 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 constitute a response to the underlying query associated with that sub-model (i.e., sub-query). A person having ordinary skill in the art would have further understood that each processed sub-model would require a response to ensure that all sub-queries are answered so that answers could be combined to form a complete configured product.

203. Further, Stahl discloses understanding a query as a collection of "sub-problems", which can be solved with "sub-solutions," and then combined 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. 1206 [Stahl] at 6, emphasis added.)

204. To the extent Stahl does not expressly disclose dividing a model into sub-models to answer sub-queries 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. As discussed above, a person having ordinary skill in the art would have concluded that the sub-models disclosed in Loomans could have been used to provide responses (i.e., solutions) to

each sub-problem disclosed in Stahl. And because Stahl discloses that all sub-solutions are later combined to form the final solution, a person having ordinary skill in the art would have appreciated reading Stahl in view of Loomans that the response from the sub-models taught in Loomans could be combined to form the final solution (answer) to the configuration query.

205. 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 processed one or more configuration queries and the configuration sub-models further comprises: generating a response for each processed configuration sub-model; and combining each response for each processed configuration sub-model to generate the answer.

k. Claim 10

> … ***[10.0] The method of claim 1 further comprising: dividing a consolidated configuration model into the configuration sub-models.***

206. 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. 1205 [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.)

207. In addition to, and in view of, the teachings of Loomans identified above, a person having ordinary skill in the art would have understood that, for an uncomplicated entity, the system of Loomans could function without sub-configuration but that as a real-world entity became more complex it would be more efficient to divide a consolidated configuration model into configuration sub-models.

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

dividing a consolidated configuration model into the configuration sub-models.

l. Claim 11

… ***[11.0] The method of claim 10 wherein dividing the consolidated configuration model into multiple configuration sub-models further comprises: dividing the configuration model so that complexity of each configuration sub-model allows processing using available data processing capabilities of the computer assisted configuration technology while still representing the relationships included in the the consolidated configuration model.***

209. Loomans discloses that sub-configuration simplifies data modeling for configuring complicated entities.

> Sub-configuration simplifies the data modeling for configuring complicated entities. (Data modeling is the process of encoding data and logic that represent information used to represent configuration information.) With sub-configuration, shared options for the top-level entity may be extracted and modeled with a single sub-model or a set of sub-models. Each sub-model may be referenced and instantiated as needed for any of the common options. The sub-models reduce the amount of redundancy by localizing data and logic needed to represent the shared options, and further simplify the data structure and logic for the top-level entity.

(Ex. 1205 [Loomans] at 1:66-2:9.)

210. Loomans further discloses that the "use of sub-models for top-level sub-configurable options can both significantly reduce the amount of data needed

to be stored for a complicated entity and further simplify the maintenance of the configuration information." (Ex. 1205 [Loomans] at 7:18-21.)

211. A person having ordinary skill in the art would have understood that utilizing the sub-configuration method taught in Loomans to reduce redundancy and simplify data modelling would reduce the processing resources for configuration. Indeed, Loomans disclose that sub-configuration "allows a complicated entity to be configured in smaller portions and incrementally, one component at a time." (Ex. 1205 [Loomans] at 11:19-21.) By configuring an entity in smaller portions a person having ordinary skill in the art would have understood that less processing power would be needed for the configuration of each smaller portion. And because the configuration occurs "one component at a time," the use of processing resources would remain reduced throughout the configuration process.

212. Finally, Loomans discloses that the sub-models continue to represent the relationships included in the consolidated (parent) configuration model. Indeed, Loomans discloses that the child sub-models are partitioned from, and therefore include the relationships/information from, the parent model.

> 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. 1205 [Loomans] at 4:48-58, emphasis added.)

213. A person having ordinary skill in the art would have understood that Loomans' disclosure of partitioning the parent model into sub-models, which represent smaller sub-level entities of the larger model, indicates that the sub-models maintain the information of the larger parent model. Indeed, the example in Loomans indicates that the top-level entity (represented by a parent model) is a computer system and the sub-level entities (represented by child sub-models) are Drive bay options (specific components), with corresponding relationships, of the computer system top-level entity. Thus, the sub-models described in Loomans maintain that component (drive bay options) relationships included in the pare model of the top-level (the computer system)

214. Accordingly, a person having ordinary skill in the art would have understood that Loomans in view of Stahl discloses that dividing the consolidated configuration model into multiple configuration sub-models further comprises: dividing the configuration model so that complexity of each configuration sub-

model allows processing using available data processing capabilities of the computer assisted configuration technology while still representing the relationships included in the consolidated configuration model.

m. Claim 12

> … ***[12.0] The method of claim 10 wherein each configuration sub-model represents a portion of the consolidated configuration model.***

215. Loomans discloses dividing a consolidated parent model into multiple child sub-models so that each child sub-model represents a sub-configurable component of the parent model.

> 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. 1205 [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.)

216. A person having ordinary skill in the art would have understood that each sub-configurable component represented by a sub-model makes up a portion of the top-level entity parent model. Indeed, Loomans discloses that the overall configuration of the top-level entity is partitioned into sub-level entities, which are represented by a child sub-models. In other words, the sub-models in Loomans are representations of portions of the overall configuration model.

217. Accordingly it is my opinion that Loomans in view of Stahl discloses that each configuration sub-model represents a portion of the consolidated configuration model.

n. Claim 13

> … ***[13.0] The method of claim 1 wherein the configurable product is a vehicle.***

218. Loomans discloses configuring aspects of a vehicle:

> Sub-level pageset 780 (a child sub-model) includes a number of features (e.g., Class, Seat Material, Number of Doors, Interior Color,

> and Exterior Color). Each of these features is associated with a respective set of choices, which may be presented with a list box (for the Class, Seat Material, and Number of Doors features), a set of icons (for the Interior and Exterior Color features), or some other representation. For each feature, any one of the available choices may be selected for the feature to configure the option.

(Ex. 1205 [Loomans] at 15:62-16:3.)

219. Thus, Loomans by itself discloses a configurable product that is a vehicle.

220. Further, both Loomans and Stahl disclose a configuration system for configuring a PC. (Ex. 1205 [Loomans] at 3:28-49; Ex. 1206 [Stahl] at 4-6.) A person having ordinary skill in the art would have understood that the methodology described in Loomans and Stahl to configure a PC could equally be applied to a vehicle. Indeed, like a PC, a vehicle configuration requires that several sub-components (such as the seat material, number of doors, etc. described in Loomans above) be configured and validated to ensure that proposed vehicle would constitute a buildable configuration. Thus, like a PC, configuring a vehicle using the teachings of Loomans and Stahl would result in similarly beneficial efficiencies.

221. Moreover, other configuration references teach the notion of solving configuration problems of vehicles. For example, the article "Automatically

Decomposing Configuration Problems" discloses that "[c]onfiguration problems can concern different domains. For instance, we might want to configure a PC, given different kinds of CPUs, memory modules, and so on; or a **car, given different kinds of engines, gears, etc.**" (Ex. 1208 [Anselma] at 1, emphasis added.)

222. Accordingly, it is my opinion that Loomans in view of Stahl discloses configuring a vehicle.

o. Claim 14

… ***[14.0] The method of claim 1 further comprising: displaying the response on display device.***

223. 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. 1205 [Loomans] at 16:36-49, emphasis added.)

224. 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. 1205 [Loomans] at 16:18-22, emphasis added.)

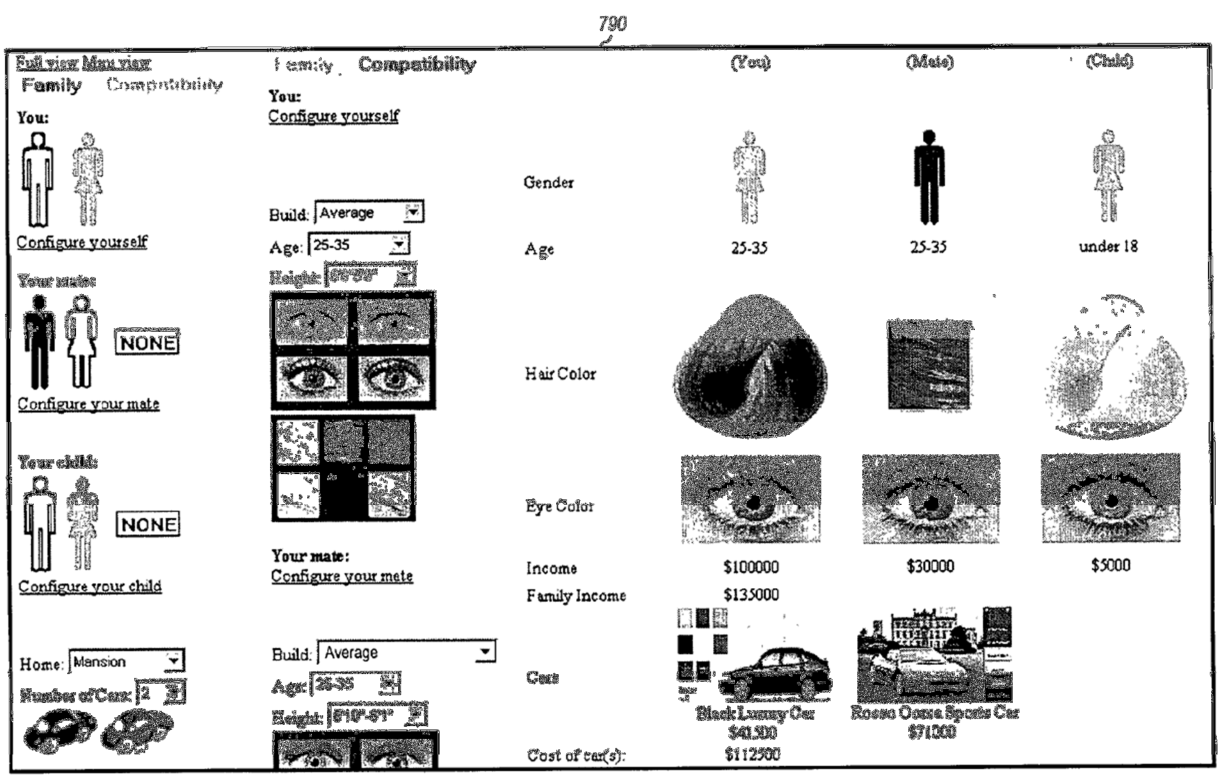


FIG. 7C

(*Id.* at Fig. 7C)

225. 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 [Figure 5]) and the result presentation**.

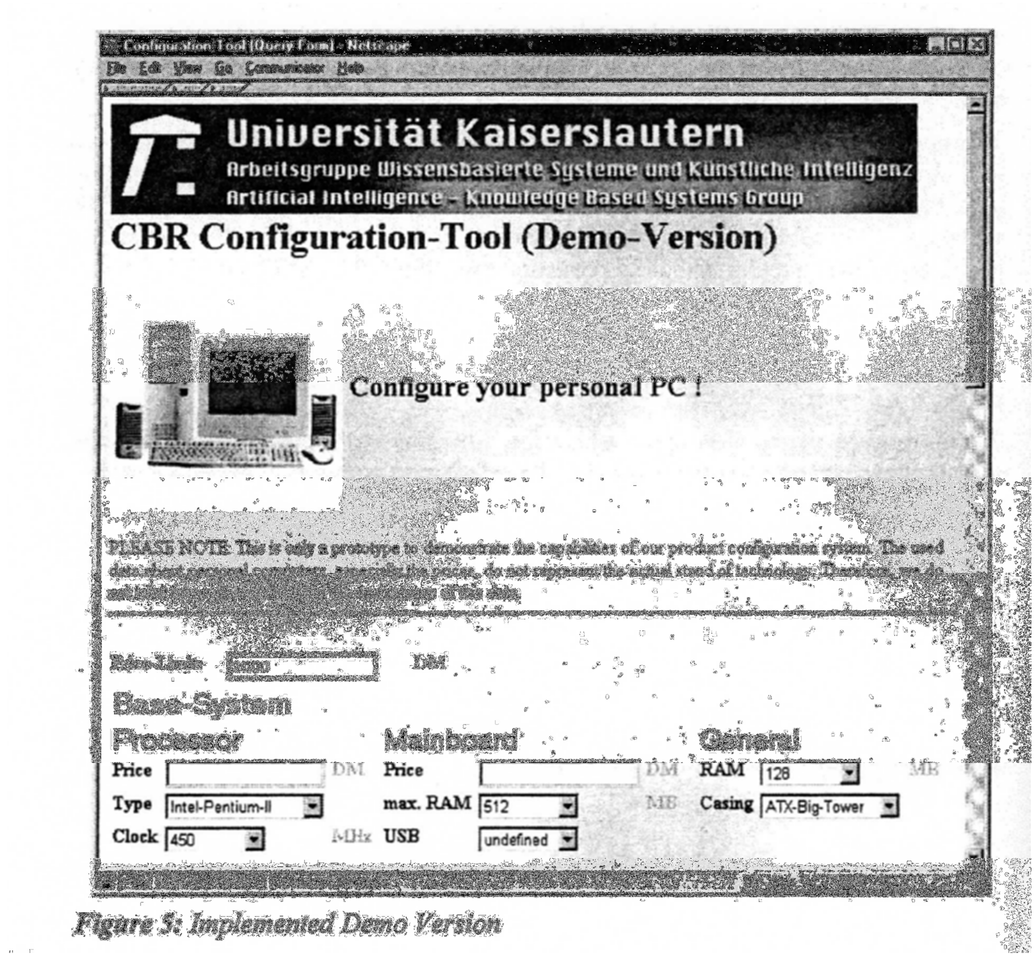


Figure 5: Implemented Demo Version

(Ex. 1206 [Stahl] at 12 and Figure 5, emphasis added.)

226. Accordingly, it is my opinion that Loomans in view of Stahl discloses *displaying the response on display device*.

p. Claim 15

> … ***[15.0] The method of claim 1 wherein the configuration sub-models each comprise data and rules to define compatibility relationships between parts included in a product.***

227. Loomans discloses configuring an entity by reference to a set of rules partitioned (i.e., divided) via a tree data structure including a parent model and multiple child sub-models.

> 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. 1205 [Loomans] at 1:57-65, emphasis added.)

228. 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. 1205 [Loomans] at 3:41-49, emphasis added.)

229. Loomans then discloses that the described 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 a 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 storages 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. 1205 [Loomans] at 6:30-46, emphasis added.)

230. Loomans further discloses an example where map sets are used to define the relationship between different components of a computer system.

> In one embodiment, the mapped features 440 in a sub-model 430 may be used to enumerate relationships between the various features in the sub-model. For example, sub-model 430a for the ABC storage device indicates that the power rating for the 20 GB device is 40 Amp, which implies that a power supply of 40 Amp or more is needed. Similarly, power supplies of 60 Amp (or more) and 80 Amp (or more) are needed for the 40 GB and 80 GB devices, respectively. Thus, in the configuration of the ABC sub-model, a value (e.g., 40 Amp) that has been selected for the Power Supply option G at the parent model may be passed down to the sub-model. **If a particular storage amount is selected (e.g., 40 GB), then the corresponding power rating is determined (e.g., 60 Amp) and compared against the selected power supply (e.g., 40 Amp). In this case, since the selected power supply is inadequate for the selected storage amount, feedback may be provided immediately so that the user may select either another storage amount (e.g., 20 GB) or another power supply (e.g., 60 Amp or more). The tables (or feature mapping set) for the mapped features may thus be used to enumerate relationships that are used to validate configuration at the sub-model**. The feature map values may be returned to the parent model upon leaving the sub-model.

(Ex. 1205 [Loomans] at 8:45-67, emphasis added.)

231. 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.

232. Further, a person having ordinary skill in the art would have understood that these compatibility relationships would have been defined using rules/constraints in the sub-models. Indeed, Loomans discloses that "[t]he use of sub-models for top-level sub-configurable options can both significantly reduce the amount of data needed to be stored for a complicated entity and further simplify the maintenance of the configuration information. **The common features and configuration rules for the entity may be extracted and represented using smaller sub-models**." (Ex. 1205 [Loomans] at 7:18-23, emphasis added.)

233. Thus, it is my opinion that Loomans in view of Stahl discloses that the configuration sub-models each comprise data and rules to define compatibility relationships between parts included in a product.

q. Claim 16

… ***[16.0] The method of claim 1 wherein the configuration problem comprises a configuration problem involving parts of a product.***

234. Stahl discloses an example problem of configuring a PC with specific

properties.

> **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 [Figure 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. 1206 [Stahl] at 6, emphasis added.)

235. In the above example, Stahl discloses that "[t]o reach this goal it is necessary to select appropriate components that fulfill the properties of the respective part-queries." (Ex. 1206 [Stahl] at 6.) Stahl then discloses using a part-query to select the type of hard drive in the example configuration of a PC. (*Id.*) A person having ordinary skill in the art would understand, based on the described example, that Stahl discloses a configuration problem (for configuring a PC),

which involves parts (e.g., hard drive) of a product (e.g., PC).

236. Accordingly, it is my opinion that Loomans in view of Stahl discloses that the configuration problem comprises a configuration problem involving parts of a product.

r. Claim 18

> … ***[18.0] A computer system to implement an inference procedure for responding to one or more configuration queries using configuration sub-models, the system comprising:***

237. For the same reasons discussed above for claim 1, 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 the combination of Loomans and Stahl above for claims 1-16 for claims 18-29.

238. 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. 1205 [Loomans] at 16:26-35, emphasis added.)

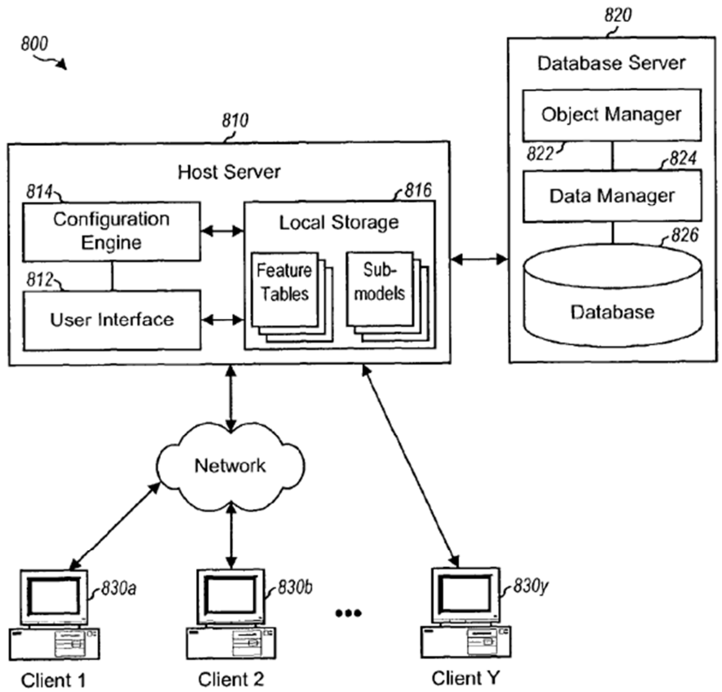


FIG. 8

(Ex. 1205 [Loomans] at Fig. 8.)

239. 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. 1205 [Loomans] at 17:42-50, emphasis added.)

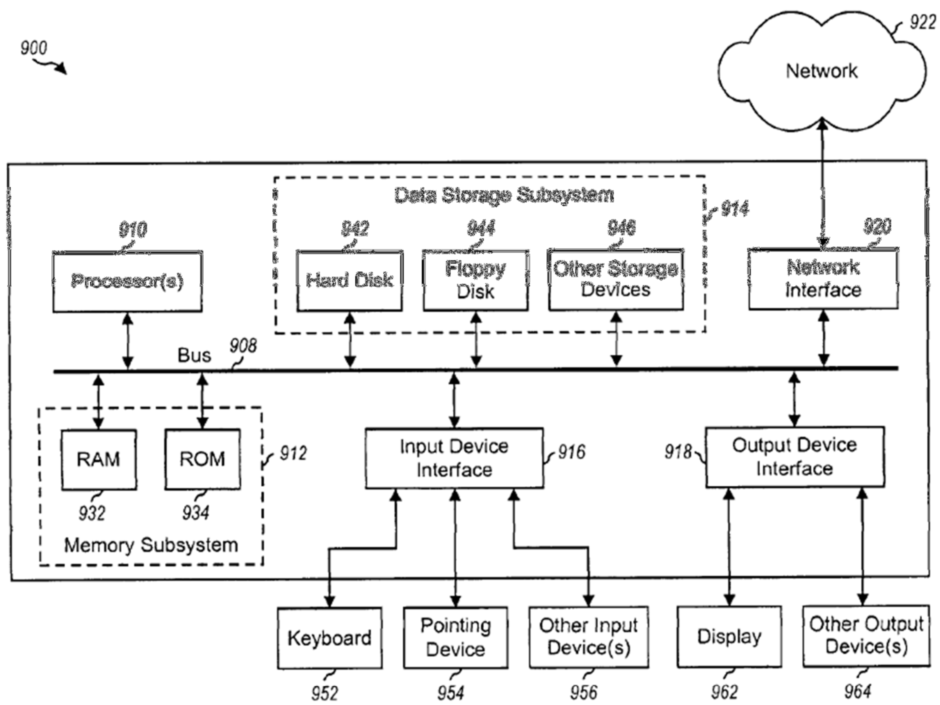


FIG. 9

(Ex. 1205 [Loomans] at Fig. 9.)

240. 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 [Figure 5]) and the result presentation.

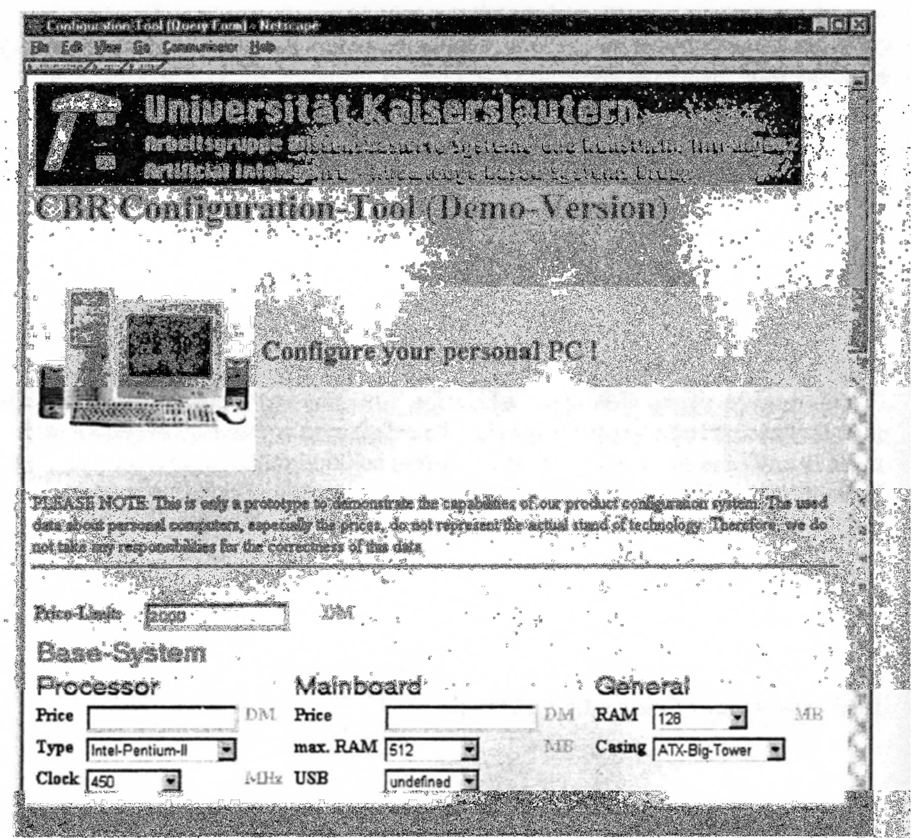


Figure 5: Implemented Demo Version

(Ex. 1206 [Stahl] at 12 and Figure 5, emphasis added.)

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.

241. 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. 1205 [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. 1205 [Loomans] at 1:57-65, emphasis added.)

242. 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. 1205 [Loomans] at 4:59-67, emphasis added.)

243. 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 [Figure 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. 1206 [Stahl] at 6, emphasis added.)

244. 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?".

245. 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. 1206 [Stahl] at 6.)

246. To the extent that Stahl does not expressly disclose dividing models into sub-models and using them to answer sub-problems, or equivalents thereof, 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. 1205 [Loomans] at Abstract.) Such models (and the rules contained therein) are commonly used during the resolving stage in a configuration system. 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.

247. 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.

… ***[18.1] a processor; and a storage medium having data encoded therein, the data comprising processor executable code for:***

248. Loomans discloses an example in Figure 9 of a computer used to implement the described configuration process. As highlighted 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. 1205 [Loomans] at 17:42-50, emphasis added.)

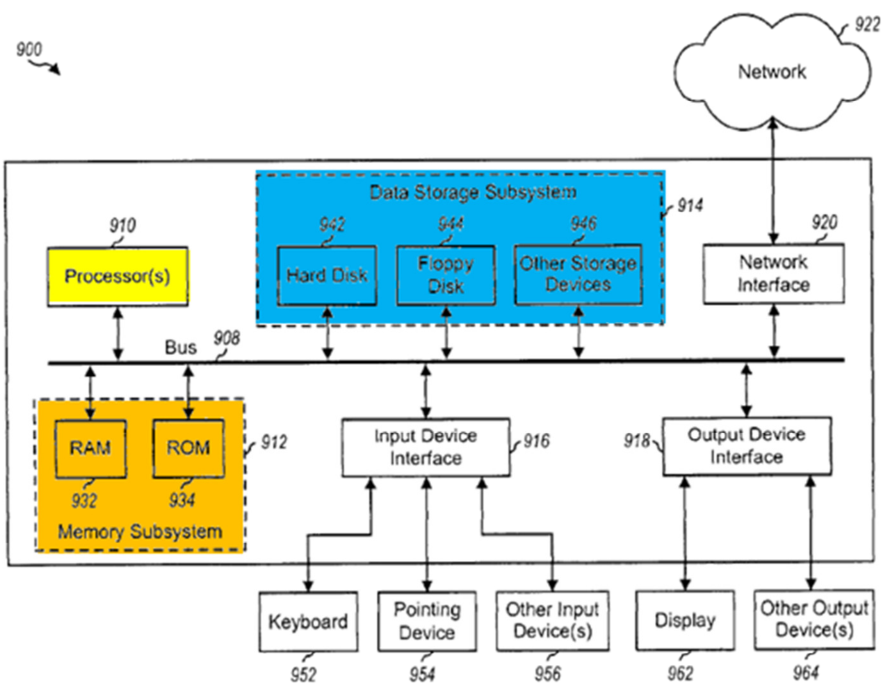


FIG. 9

(Ex. 1205 [Loomans] at Fig. 9, annotated.)

249. 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. 1205 [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.)

250. 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, the data comprising processor executable code* for performing steps [18.2]-[18.8] (below).

> … ***[18.2] receiving one or more configuration queries representing a questions involving parts and part relationships in a configuration of a configurable product;***

251. This limitation is identical to claim element [1.1], except that this claim includes the phrase "representing a questions" rather than "representing one or more questions." Despite the ambiguity between the article and the plural form 'questions' used this limitation, the minor distinction in phrasing does not change the scope of the claim limitation or the applicability of the analysis provided above with respect to claim element [1.1]. Accordingly, this limitation is met by the prior art as described in connection with claim element [1.1] (above). Thus, I incorporate my analysis for element [1.1] into my analysis for this claim element.

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

… ***[18.3] dividing one or more configuration queries into multiple configuration sub-queries, wherein the multiple configuration sub-queries represent the one or more configuration queries;***

253. This limitation is identical to claim element [1.2b]. Accordingly, this limitation is met by the prior art as described in my analysis of element [1.2b] (above). Thus, I incorporate my analysis for element [1.2b] into my analysis for this claim element.

254. 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**,**

… ***[18.4] processing each sub-query using at least one configuration sub-model per sub-query,***

255. This limitation is identical to claim element [1.3]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [1.3] (above). Thus, I incorporate my analysis for element [1.3] for this claim element.

256. 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**.**

… ***[18.5] 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***

257. This limitation is identical to claim element [1.4]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [1.4] (above). Thus, I incorporate my analysis for element [1.4] into my analysis for this claim element.

258. 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.

… ***[18.6] 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;***

259. This limitation is identical to claim element [1.5]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [1.5] (above). Thus, I incorporate my analysis for element [1.5] into my analysis for this claim element.

260. 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.

… ***[18.7] 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***

261. This limitation is identical to claim element [1.6]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [1.6] (above). Thus, I incorporate my analysis for element [1.6] into my analysis for this claim element.

262. 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.

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

263. This limitation is identical to claim element [1.7]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [1.7] (above). Thus, I incorporate my analysis for element [1.7] for this claim element.

264. 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.

s. Claim 19

> … ***[19.0] The computer system of claim 18 wherein the one or more configurations queries relate to a configuration completion problem.***

265. The limitation of this claim is identical to that of claim 2. Accordingly, this claim is met by the prior art as described in my analysis of claim 2 (above). Thus, I incorporate my analysis for claim 2 into my analysis for this claim element.

266. Accordingly, it is my opinion that Loomans in view of Stahl discloses that the one or more configurations queries relate to a configuration completion problem.

t. Claim 20

> … ***[20.0] The computer system of claim 18 wherein the one or more configuration queries relate to a configuration validation problem and when solving the configuration validation problem, and the code for processing one or more configuration queries further comprises: processing at least one of the sub-queries using different configuration sub-models until a configuration validation answer can be determined.***

267. This claim is identical to claim 4, except that this claim includes the phrase "*when solving the configuration validation problem*," and that the step is

provided in "*code*" which does not further limit claim 20 beyond claim 4 as the limitations of claim 4 likewise describe solving a validation problem. And Loomans discloses a computer used to implement the described configuration process that includes a memory subsystem that "may include a RAM 932 and a ROM 934 used to store codes and data that implement various aspects of the invention." (Ex. 1205 [Loomans] at 17:42-53.) As described above, the prior art discloses this limitation during the process of *solving a configuration validation problem*. Accordingly, notwithstanding the minor addition to the language of the claim, this claim is met by the prior art as described in my analysis of claim 4 (above). Thus, I incorporate my analysis for claim 4 into my analysis for this claim element.

268. Accordingly it is my opinion that Loomans in view of Stahl discloses that the one or more configuration queries relate to a configuration validation problem and when solving the configuration validation problem, and the code for processing one or more configuration queries further comprises: processing at least one of the sub-queries using different configuration sub-models until a configuration validation answer can be determined.

u. Claim 21

… ***[21.0] The computer system of claim 18 wherein the data collectively included in the configuration sub-models provides a response for each of the sub-queries being processed.***

269. The limitation of this claim is identical to that of claim 5. Accordingly, this claim is met by the prior art as described in connection with claim 5 (above). Thus, I incorporate my analysis for claim 5 into my analysis for this claim element.

270. Accordingly, it is my opinion that Loomans in view of Stahl discloses that the data collectively included in the configuration sub-models provides a response for each of the sub-queries being processed.

v. Claim 22

… ***[22.0] The computer system of claim 18 wherein at least two sub-queries include overlapping information.***

271. The limitation of this claim is identical to that of claim 6. Accordingly, this claim is met by the prior art as described in my analysis of claim 6 (above). Thus, I incorporate my analysis for claim 6 into my analysis for this claim element.

272. Accordingly, it is my opinion that Loomans in view of Stahl discloses that *at least two sub-queries include overlapping information.*

w. Claim 23

> … ***[23.0] The computer system of claim 18 wherein the code further comprises code for: dividing the configuration sub-models in accordance with a predetermined data structure; and dividing the sub-queries in accordance with the sub-model structure.***

273. Loomans discloses dividing a top-entity (e.g., Computer System), which is represented as a parent model, into sub-level entities (e.g., Drive Bays 1, 2 and 3 of the Computer System), which are represented as 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. 1205 [Loomans] at 4:48-58, emphasis added.)

274. Loomans further discloses a sub-configuration process where mapped features are passed between the parent model and the child 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.

(Ex. 1205 [Loomans] at 6:11-22, emphasis added.)

275. Thus, Loomans discloses dividing the parent model into sub-models in a way that allows mapped features to be passed between the parent model and the sub-model. A person having ordinary skill in the art would have understood that this means that the sub-models would have been divided with a predetermined data structure to ensure that sub-model could receive the appropriate mapped feature data from the parent model.

276. A person having ordinary skill in the art would have understood that the child sub-models disclosed in Loomans would need to be partitioned out in a predetermined way to ensure that the sub-models were related to particular components of the top level entity. If the sub-models were randomly defined, unrelated components could be included in the sub-model, which could reduce the system's configuration efficiency. Indeed, in such circumstances, the sub-model could require processing power for components that have no relevance to a

particular sub-query, but the system would still cycle through the unrelated components when configuring in view of the particular sub-model. Thus, a person having ordinary skill in the art would have understood that a predetermined structure, such as the one discussed in Loomans where the drive bays of the computer are assigned to a particular sub-model, would be preferred to minimize the complexity of the sub-models by including unnecessary components in a particular sub-model.

277. As discussed above, 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 configuration 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?".

278. A person having ordinary skill in the art would have understood that each query would have been divided out to correspond with a particular sub-model to ensure that the proper information is being queried from the proper sub-model for a particular mapped feature.

279. Further, as discussed above, Stahl discloses understanding a query as

a collection 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. 1206 [Stahl] at 6, emphasis added.)

280. For similar reason as discussed with regards to the sub-models in Loomans, a person having ordinary skill in the art would have understood that dividing the sub-queries in accordance with the subject matter in the sub-models would lead to increased efficiencies. Indeed, such a division would limit the information sought by each sub-query to that included in its corresponding sub-model. This 1-to-1 correlation would eliminate the wasting of processing resources by eliminating a query for components not included in the sub-model.

281. Accordingly, it is my opinion that Loomans in view of Stahl discloses that the code further comprises code for: dividing the configuration sub-models in accordance with a predetermined data structure; and dividing the sub-queries in accordance with the sub-model structure.

x. Claim 24

> … *__[24.0] The computer system of claim 23 wherein the predetermined data structure comprises a data structure divided along configuration model part groups, wherein the part groups are a collection of related parts.__*

282. The limitation of this claim is identical to that of claim 8. Accordingly, this claim is met by the prior art as described in my analysis of claim 8 (above). Thus, I incorporate my analysis for claim 8 into my analysis for this claim element.

283. Accordingly, it is my opinion that Loomans in view of Stahl discloses that the predetermined data structure comprises a data structure divided along configuration model part groups, wherein the part groups are a collection of related parts.

y. Claim 25

> … *__[25.0] The computer system of claim 18 wherein the code for generating a response to the one or more configuration queries based upon the processed one or more configuration queries and the configuration sub-models further comprises code for: generating a response for each processed configuration sub-model; and combining each response for each processed configuration sub-model to generate the answer.__*

284. The limitation of this claim is similar to that of claim 9. The only

difference is that step is provided in "*code*." Loomans discloses a computer used to implement the described configuration process that includes a memory subsystem that "may include a RAM 932 and a ROM 934 used to store codes and data that implement various aspects of the invention." (Ex. 1205 [Loomans] at 17:42-53.) Accordingly, this claim is met by the prior art as described in my analysis of claim 9 (above). Thus, I incorporate my analysis for claim 9 into my analysis for this claim element.

285. Accordingly, it is my opinion that Loomans in view of Stahl discloses that the code for generating a response to the one or more configuration queries based upon the processed one or more configuration queries and the configuration sub-models further comprises code for: generating a response for each processed configuration sub-model; and combining each response for each processed configuration sub-model to generate the answer.

z. Claim 26

… ***[26.0] The computer system of claim 18 wherein the code for dividing the consolidated configuration model into multiple configuration sub-models further comprises code for: dividing the configuration model so that complexity of each configuration sub-model allows processing using available data processing capabilities of the computer system while still representing the relationships included in the consolidated configuration model.***

286. The limitation of this claim is similar to that of claim 11. The only

difference is that step is provided in "code." Loomans discloses a computer used to implement the described configuration process that includes a memory subsystem that "may include a RAM 932 and a ROM 934 used to store codes and data that implement various aspects of the invention." (Ex. 1205 [Loomans] at 17:42-53.) Accordingly, this claim is met by the prior art as described in my analysis of claim 11 (above). Thus, I incorporate my analysis for claim 11 into my analysis for this claim element.

287. Accordingly, it is my opinion that Loomans in view of Stahl discloses that the code for dividing the consolidated configuration model into multiple configuration sub-models further comprises code for: dividing the configuration model so that complexity of each configuration sub-model allows processing using available data processing capabilities of the computer system while still representing the relationships included in the consolidated configuration model.

aa. Claim 27

> … ***[27.0] The computer system of claim 18 wherein the data further comprises processor executable code for: dividing a consolidated configuration model into the configuration sub-models.***

288. The limitation of this claim is similar to that of claim 10 and therefore I incorporate my analysis for claim 10 into my analysis for this claim element. The only difference is that "code" is associated with the "dividing" operation. The '057 Patent provides no information regarding how a division might be accomplished

automatically other than that it could be accomplished with a "divide-model" procedure (Ex. 1201 ['057 Patent] at 7:15-21). In other words, no algorithm or flowchart is disclosed for dividing the model. Loomans discloses a computer used to implement the described configuration process that includes a memory subsystem that "may include a RAM 932 and a ROM 934 used to store codes and data that implement various aspects of the invention." (Ex. 1205 [Loomans] at 17:42-53.) And Loomans Claim 1 describes "computer readable program codes" that can "extract the subset of the plurality of selectable features" from a parent model and "in response to the extracting, generate a sub-model of the parent model."[12] Accordingly, a person having ordinary skill in the art would have

[12] To the extent this claim requires a fully automated division of a consolidated model, a PHOSITA would have understood that such an automated division would be analogous to the computer science process of compilation. A conventional computer can perform just one task at a time and therefore automated systems for dividing up the complex into subcomponents have been conventional since the 1950s. This is made explicit in some of the background materials cited above, for example Section 3.2 of Kott: "In some cases an and-specialist may also generate the children of an and-node dynamically, as the design process progresses." (Ex. 1207 [Kott] at 7) See also McDermott, section 2.2, in which rules regarding in

understood that the code described in Loomans could be used to generate the sub-models described in Loomans.

289. Accordingly, it is my opinion that Loomans in view of Stahl and the knowledge of a person of ordinary skill in the art discloses that the data further comprises processor executable code for: dividing a consolidated configuration model into the configuration sub-models.

bb. Claim 28

… ***[28.0] The computer system of claim 27 wherein the code for dividing the consolidated configuration model into multiple configuration sub-models further comprises code for: dividing the configuration model so that complexity of each configuration sub-model allows processing using available data processing capabilities of the computer system while still representing the relationships included in the consolidated configuration model.***

290. The limitation of this claim is similar to that of claim 11 and I incorporate my analysis for claim 11 into my analysis for this claim element. The only difference is that "code" is associated with the division. As discussed above, Loomans discloses a computer used to implement the described configuration process that includes a memory subsystem that "may include a RAM 932 and a

---

which order to evaluate rules are described (McDermott at 11). This has the effect of dynamically and automatically constructing a partitioned configuration model.

ROM 934 used to store codes and data that implement various aspects of the invention." (Ex. 1205 [Loomans] at 17:42-53.) And Loomans Claim 1 describes "computer readable program codes" that can "extract the subset of the plurality of selectable features" from a parent model and "in response to the extracting, generate a sub-model of the parent model." Accordingly, a person having ordinary skill in the art would have understood that the code described in Loomans could be used to generate the sub-models described in Loomans. Thus, this claim is met by the prior art as described in connection with claim 11 (above).

291. Accordingly, it is my opinion that Loomans in view of Stahl discloses that the code for dividing the consolidated configuration model into multiple configuration sub-models further comprises code for: dividing the configuration model so that complexity of each configuration sub-model allows processing using available data processing capabilities of the computer system while still representing the relationships included in the consolidated configuration model.

cc. Claim 29

… ***[29.0] The computer system of claim 27 wherein each configuration sub-model represents a portion of the consolidated configuration model.***

292. The limitation of this claim is identical to that of claim 12. Accordingly, this claim is met by the prior art as described in my analysis of claim 12 (above). Thus, I incorporate my analysis for claim 12 into my analysis for this

claim element.

293. Accordingly, it is my opinion that Loomans in view of Stahl discloses that each configuration sub-model represents a portion of the consolidated configuration model.

dd. Claim 31

> … ***[31.0] 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 processor executable code for:***

294. For the same reasons discussed above for claim 1, 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 the combination of Loomans and Stahl above for claims 1-16 for claims 31-43.

295. 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. 1205 [Loomans] at 17:42-50, emphasis added.)

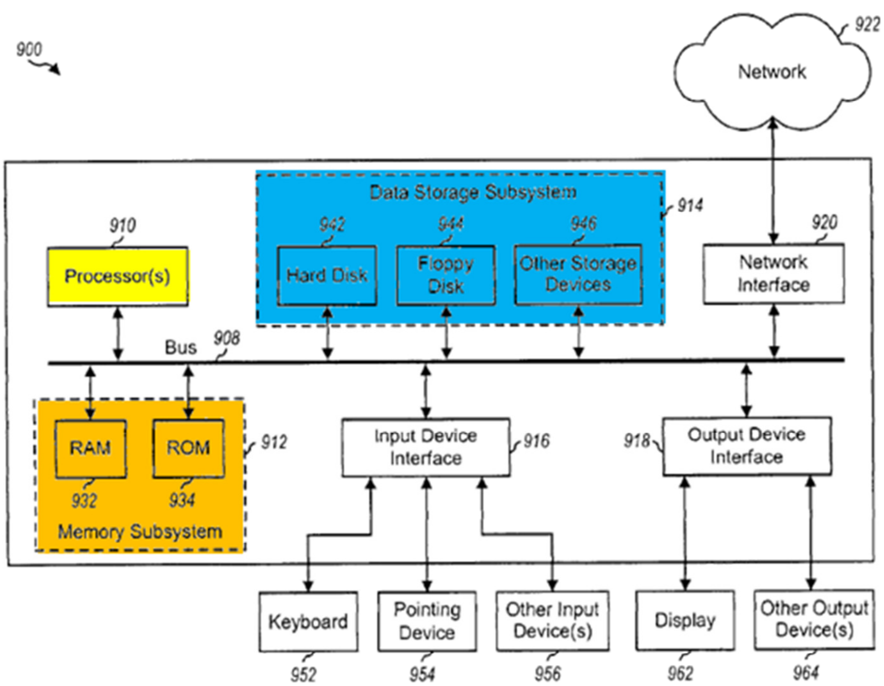


FIG. 9

(Ex. 1205 [Loomans] at Fig. 9, annotated.)

296. 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. 1205 [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.)

297. 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. 1205 [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. 1205 [Loomans] at 1:57-65, emphasis added.)

298. 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. 1205 [Loomans] at 4:59-67, emphasis added.)

299. 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 [Figure 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. 1206 [Stahl] at 6, emphasis added.)

300. 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?".

301. 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. 1206 [Stahl] at 6.)

302. To the extent that Stahl does not expressly disclose dividing models into sub-models and using them to answer sub-problems, or equivalents thereof, 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. Such models (and the rules contained therein) are commonly used during the resolving stage in a configuration system. Indeed, Loomans discloses that "each sub-configurable component is configured via one of a number of sub-models." (Ex. 1205 [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.

303. 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.

… ***[31.1] receiving one or more configuration queries representing a questions involving parts and part relationships in a configuration of a configurable product;***

304. This limitation is identical to claim element [1.1], except that this claim includes the phrase "representing a questions" rather than "representing one

or more questions." Despite the ambiguity between the article and the plural form 'questions' used this limitation, the minor distinction in phrasing does not change the scope of the claim limitation or the applicability of the analysis provided above with respect to claim element [1.1]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [1.1]. Thus, I incorporate my analysis for claim element [1.1] into my analysis for this claim element.

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

> … ***[31.2] dividing one or more configuration queries into multiple configuration sub-queries, wherein the multiple configuration sub-queries represent the one or more configuration queries;***

306. The function of this limitation is identical to claim element [1.2b]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [1.2b] (above). Thus, I incorporate my analysis for claim element [1.2b] into my analysis for this claim element. That this step is associated with "executable code" leads to an analysis identical to that undertaken for claim 27, above, which I incorporate by reference.

307. 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.

… ***[31.3] processing each sub-query using at least one configuration sub-model per sub-query,***

308. This limitation is identical to claim element [1.3]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [1.3] (above). Thus, I incorporate my analysis for claim element [1.3] into my analysis for this claim element.

309. 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.

… ***[31.4] 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***

310. This limitation is identical to claim element [1.4]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [1.4] (above). Thus, I incorporate my analysis for claim element [1.4] into my analysis for this claim element.

311. 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.

> … ***[31.5] 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;***

312. This limitation is identical to claim element [1.5]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [1.5] (above). Thus, I incorporate my analysis for claim element [1.5] into my analysis for this claim element.

313. 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.

> … ***[31.6] 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; and***

314. This limitation is identical to claim element [1.6]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [1.6] (above). Thus, I incorporate my analysis for claim element [1.6] into my analysis for this claim element.

315. 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.

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

316. This limitation is identical to claim element [1.7]. Accordingly, this limitation is met by the prior art as described in my analysis of claim element [1.7] (above). Thus, I incorporate my analysis for claim element [1.7] into my analysis for this claim element.

317. 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.

ee. Claim 32

… ***[32.0] The computer storage medium of claim 31 wherein the one or more configuration queries relate to a configuration completion problem.***

318. The limitation of this claim is identical to that of claim 2. Accordingly, this claim is met by the prior art as described in my analysis of claim 2 (above). Thus, I incorporate my analysis for claim 2 into my analysis for this claim element.

319. Accordingly, it is my opinion that Loomans in view of Stahl discloses that the one or more configuration queries relate to a configuration completion problem.

ff. Claim 33

… ***[33.0] The computer storage medium of claim 31 wherein the data further comprises processor executable code for: processing each sub-query using multiple configuration sub-models per sub-query.***

320. The limitation of this claim is identical to that of claim 3. Accordingly, this claim is met by the prior art as described in my analysis of claim 3 (above) Thus, I incorporate my analysis for claim 3 into my analysis for this claim element.

321. Accordingly, it is my opinion that Loomans in view of Stahl disclose that the data further comprises processor executable code for: processing each sub-query using multiple configuration sub-models per sub-query.

gg. Claim 34

… ***[34.0] The computer storage medium of claim 31 wherein the one or more configuration queries relate to a configuration validation problem and the code for processing one or more configuration queries further comprises: processing at least one of the sub-queries using different configuration sub-models until a configuration validation answer can be determined.***

322. The limitation of this claim is similar to that of claim 4. The only difference is that step is provided in "code." Loomans discloses a computer used to implement the described configuration process that includes a memory subsystem that "may include a RAM 932 and a ROM 934 used to store codes and data that implement various aspects of the invention." (Ex. 1205 [Loomans] at 17:42-53.) Accordingly, this claim is met by the prior art as described in my analysis of claim 4 (above). Thus, I incorporate my analysis for claim 4 into my analysis for this claim element.

323. Accordingly, it is my opinion that Loomans in view of Stahl disclose that the one or more configuration queries relate to a configuration validation problem and the code for processing one or more configuration queries further comprises: processing at least one of the sub-queries using different configuration sub-models until a configuration validation answer can be determined.

hh. Claim 35

… ***[35.0] The computer storage medium of claim 31 wherein the data collectively included in the configuration sub-models provides a response for each of the sub-queries being processed.***

324. The limitation of this claim is identical to that of claim 5. Accordingly, this claim is met by the prior art as described in my analysis of claim 5 (above). Thus, I incorporate my analysis for claim 5 into my analysis for this claim element.

325. Accordingly, it is my opinion that Loomans in view of Stahl discloses that the data collectively included in the configuration sub-models provides a response for each of the sub-queries being processed.

ii. Claim 36

… ***[36.0] The computer storage medium of claim 31 wherein at least two sub-queries include overlapping information.***

326. The limitation of this claim is identical to that of claim 6. Accordingly, this claim is met by the prior art as described in my analysis of claim 6 (above). Thus, I incorporate my analysis for claim 6 into my analysis for this claim element.

327. Accordingly, it is my opinion that Loomans in view of Stahl discloses that *at least two sub-queries include overlapping information.*

jj. Claim 37

… ***[37.0] The computer storage medium of claim 31 the code further comprises code for: dividing the configuration sub-models in accordance with a predetermined data structure; and dividing the sub-queries in accordance with the sub-model structure.***

328. The limitation of this claim is identical to that of claim 23. Accordingly, this claim is met by the prior art as described in my analysis of claim 23 (above). Thus, I incorporate my analysis for claim 23 into my analysis for this claim element. See also my analysis of claim 27 regarding the code-related aspect of this claim.

329. Accordingly, it is my opinion that Loomans in view of Stahl discloses that the code further comprises code for: dividing the configuration sub-models in accordance with a predetermined data structure; and dividing the sub-queries in accordance with the sub-model structure.

kk. Claim 38

… ***[38.0] The computer storage medium of claim 37 wherein the predetermined data structure comprises a data structure divided along configuration model part groups, wherein the part groups are a collection of related parts.***

330. The limitation of this claim is identical to that of claim 8. Accordingly, this claim is met by the prior art as described in my analysis of claim 8 (above). Thus, I incorporate my analysis for claim 8 into my analysis for this

claim element.

331. Accordingly, it is my opinion that Loomans in view of Stahl discloses that the predetermined data structure comprises a data structure divided along configuration model part groups, wherein the part groups are a collection of related parts.

ll. Claim 39

… ***[39.0] The computer storage medium of claim 31 wherein the code for generating a response to the one or more configuration queries based upon the processed one or more configuration queries and the configuration sub-models further comprises code for: generating a response for each processed configuration sub-model; and combining each response for each processed configuration sub-model to generate the answer.***

332. The limitation of this claim is similar to that of claim 9. The only difference is that step is provided in "code." Loomans discloses a computer used to implement the described configuration process that includes a memory subsystem that "may include a RAM 932 and a ROM 934 used to store codes and data that implement various aspects of the invention." (Ex. 1205 [Loomans] at 17:42-53.) Accordingly, this claim is met by the prior art as described in my analysis of claim 9 (above). Thus, I incorporate my analysis for claim 9 into my analysis for this claim element.

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

that the code for generating a response to the one or more configuration queries based upon the processed one or more configuration queries and the configuration sub-models further comprises code for: generating a response for each processed configuration sub-model; and combining each response for each processed configuration sub-model to generate the answer.

mm. Claim 40

> … ***[40.0] The computer storage medium of claim 31 wherein the code for dividing the consolidated configuration model into multiple configuration sub-models further comprises code for: dividing the configuration model so that complexity of each configuration sub-model allows processing using available data processing capabilities of the computer system while still representing the relationships included in the consolidated configuration model.***

334. The limitation of this claim is similar to that of claim 11 and I incorporate my analysis for claim 11 into my analysis for this claim element. The only difference is that "code" is associated with the division. As discussed above, Loomans discloses a computer used to implement the described configuration process that includes a memory subsystem that "may include a RAM 932 and a ROM 934 used to store codes and data that implement various aspects of the invention." (Ex. 1205 [Loomans] at 17:42-53.) And Loomans Claim 1 describes "computer readable program codes" that can "extract the subset of the plurality of selectable features" from a parent model and "in response to the extracting,

generate a sub-model of the parent model." Accordingly, a person having ordinary skill in the art would have understood that the code described in Loomans could be used to generate the sub-models described in Loomans. Thus, this claim is met by the prior art as described in connection with claim 11 (above).

335. Accordingly, it is my opinion that Loomans in view of Stahl discloses that the code for dividing the consolidated configuration model into multiple configuration sub-models further comprises code for: dividing the configuration model so that complexity of each configuration sub-model allows processing using available data processing capabilities of the computer system while still representing the relationships included in the consolidated configuration model.

nn. Claim 41

> … ***[41.0] The computer storage medium of claim 31 wherein the data further comprises processor executable code for: dividing a consolidated configuration model into the configuration sub-models.***

336. The limitation of this claim is similar to that of claim 10 and therefore I incorporate my analysis for claim 10 into my analysis for this claim element. The only difference is that "code" is associated with the "dividing" operation. The '057 Patent provides no information regarding how a division might be accomplished automatically other than that it could be accomplished with a "divide-model" procedure (Ex. 1201 ['057 Patent] at 7:15-21). In other words, no algorithm or

flowchart is disclosed for dividing the model. Loomans discloses a computer used to implement the described configuration process that includes a memory subsystem that "may include a RAM 932 and a ROM 934 used to store codes and data that implement various aspects of the invention." (Ex. 1205 [Loomans] at 17:42-53.) And Loomans Claim 1 describes "computer readable program codes" that can "extract the subset of the plurality of selectable features" from a parent model and "in response to the extracting, generate a sub-model of the parent model."[13] Accordingly, a person having ordinary skill in the art would have understood that the code described in Loomans could be used to generate the sub-

[13] To the extent this claim requires a fully automated division of a consolidated model, a PHOSITA would have understood that such an automated division would be analogous to the computer science process of compilation. A conventional computer can perform just one task at a time and therefore automated systems for dividing up the complex into subcomponents have been conventional since the 1950s. This is made explicit in some of the background materials cited above, for example Section 3.2 of Kott: "In some cases an and-specialist may also generate the children of an and-node dynamically, as the design process progresses." (Ex. 1207 [Kott] at 7) See also McDermott, section 2.2, in which rules regarding in which order to evaluate rules are described (McDermott at 11). This has the effect of dynamically and automatically constructing a partitioned configuration model.

models described in Loomans.

337. Accordingly, it is my opinion that Loomans in view of Stahl discloses that the data further comprises processor executable code for: dividing a consolidated configuration model into the configuration sub-models.

oo. Claim 42

… ***[42.0] The computer storage medium of claim 41 wherein the code for dividing the consolidated configuration model into multiple configuration sub-models further comprises code for: dividing the configuration model so that complexity of each configuration sub-model allows processing using available data processing capabilities of the computer system while still representing the relationships included in the consolidated configuration model.***

338. The limitation of this claim is similar to that of claim 11 and I incorporate my analysis for claim 11 into my analysis for this claim element. The only difference is that "code" is associated with the division. As discussed above, Loomans discloses a computer used to implement the described configuration process that includes a memory subsystem that "may include a RAM 932 and a ROM 934 used to store codes and data that implement various aspects of the invention." (Ex. 1205 [Loomans] at 17:42-53.) And Loomans Claim 1 describes "computer readable program codes" that can "extract the subset of the plurality of selectable features" from a parent model and "in response to the extracting, generate a sub-model of the parent model." Accordingly, a PHOSITA would have

understood that the code described in Loomans could be used to generate the sub-models described in Loomans. Thus, this claim is met by the prior art as described in connection with claim 11 (above).

339. Accordingly, it is my opinion that Loomans in view of Stahl discloses that the code for dividing the consolidated configuration model into multiple configuration sub-models further comprises code for: dividing the configuration model so that complexity of each configuration sub-model allows processing using available data processing capabilities of the computer system while still representing the relationships included in the consolidated configuration model**.**

pp. Claim 43

… ***[43.0] The computer storage medium of claim 41 wherein each configuration sub-model represents a portion of the consolidated configuration model.***

340. The limitation of this claim is identical to that of claim 12. Accordingly, this claim is met by the prior art as described in my analysis of claim 12 (above). I incorporate my analysis for claim 12 into my analysis for this claim element.

341. Accordingly, it is my opinion that Loomans in view of Stahl discloses that *each* configuration *sub-model represents a portion of the consolidated configuration model.*

XI. Conclusion

342. 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.

343. 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: 10/27/2016

Philip Greenspun

Philip Greenspun, Ph.D.