DETC2005-85377

DESIGN SPACE EXPLORATION FOR RAPID PRODUCT REALIZATION

Srikanth Devanathan
Robert Cunningham
Christopher Peters
Karthik Ramani¹

Purdue Research and Education Center for Information Systems in Engineering PRECISE,
School of Mechanical Engineering,
Purdue University,
West Lafayette, IN 47907, USA

ABSTRACT

In this work, a product centric view of the design process is used in capturing the process and the design knowledge of complex, hierarchical engineering systems. The use of individual product configurators expedites redesign and further exploration of the design space. The notion of concept representation is introduced to mathematically formulate the various aspects of a product concept. This representation also enhances the understanding of the overall objectives, allowing for design sensitivity analyses, pre-computation of the engineering design spaces and multi-disciplinary constraint management between modules. Product configurators are developed in a distributed framework to enable rapid design across different platforms. The Purdue Formula SAE vehicle is used as an example case in developing the configuration problem.

INTRODUCTION

Design is the most important stage in the process of realizing a product. Almost 95% of the cost is committed during design [1]. Hence manufacturers spend considerable time and effort in avoiding errors. This entails the designer to perform a set of tasks repeatedly to ascertain the feasibility and effectiveness of a particular design. Some of these tasks cannot be performed without the creativity of the designer. Moreover, these tasks are interconnected and hence warrant an iterative design process. To reduce the design time, the time taken for each execution of the steps and the number of iterations themselves are to be minimized. The advent of computers and consequently tools such as analysis, optimization and 3D parametric modeling has greatly reduced the design time by reducing the time taken to perform each step. The objective of this work is to develop a mathematical representation of a

product to enable rapid design through configuration as shown in

In this way, engineering design can most effectively utilize advances in data storage, computation speed and knowledge sharing repositories. A configurator, for product design purposes, is defined as a software element that instantiates a concept to meet requirements. The concept representation, developed here, forms the basis for product design through configuration. A concept is a generic representation of a product. Instantiation of a concept typically involves (a) defining the architecture of the product, (b) choice of values for its various parameters, (c) definition of its form, and, (d) evaluation of its performance.

Figure 1: Relationships between Concept, Configurator and Product

THE DESIGN PROCESS

The design process can be considered as a map, albeit an iterative one, from the perceived Customer Requirements (CRs) to the final design of the product that satisfies those

¹ Corresponding author: ramani@purdue.edu

available in components, their associated constraints, functional specifications, and geometric data (CAD). To enable rapid product design, the configurator must handle products that span multiple product families and domains. Hence, the attributes need to be defined for each domain and the constraints on those attributes need to be represented appropriately. Several languages and representations have been developed for abstracting knowledge; however the ontological approach is the most widely used.

Until recently, database systems did not address the issue of storing information-rich, relational data structures (graphs and trees) due to high time-complexity. However, as information becomes more complicated, graph structures will provide significant advantages for representation. In the product domain, assemblies can be represented as graphs through the topology thereby various graph matching algorithms can be used to search for concepts.

Previous work has been performed on the influence of high level target specifications on the sub levels of a hierarchical system [16, 17]. In order to propagate these specifications to the lower levels, a well defined methodology of targets, responses, coupling variables, and shared design variables must be established. The goals of one such methodology, Analytical Target Cascading (ATC), are to enable the design of complex systems to occur at a much earlier stage and to generate prototypes that more effectively satisfy functional requirements.

The area of product configuration has stimulated a large amount of interest in academic areas such as Artificial Intelligence (AI), optimization [18-20], and simulation-based design [21-24]. In the concept and structure hierarchy approach, a frame-based representation and an object-oriented approach are mainly used to represent the conceptual level knowledge and compositional structure of configuration systems [25, 26]. Case-based problem solving methods use the knowledge from solved tasks for generating solutions to new designs, with the inherent assumption that similar tasks lead to similar solutions. However, product design involves complex tasks and constraints that cannot be simplified to obtain useful solutions through techniques such as case-based reasoning and rules-based configuration. A product is designed to meet functional requirements, achieve desired performance and satisfy constraints.

Constraint-based configuration systems represent and evaluate the relations among components and their properties with the help of constraints. These representations are specified in a knowledge base and can be evaluated by constraint propagation [27-30]. Lin and Chen [31] integrate the performance criteria and functional coupling of concepts in an optimization framework for concept selection.

Felfernig et al. [32-36] have made an attempt towards distributed configurators that can handle requests on a web-based platform in a supply chain setting. There is a limited amount of literature that deals with these types of distributed configurators. Configurators that cater to distributed product design do not exist currently. Analogous applications assume a central location of all knowledge. However, in most practical situations, the knowledge is spread across the organization and, in the case of complex products, across suppliers. Since configuration is a dynamic task, it cannot be represented as a

traditional Constraint Satisfaction Problem (CSP). Hence conventional CSP techniques are not directly applicable. In other generic CSP representations [37], configuration with distributed knowledge sources has not been addressed

MATHEMATICAL FRAMEWORK FOR CONCEPT REPRESENTATION

As mentioned previously, products can be represented through a topological framework, such as a graph. Each product component is represented as vertices of a graph, and the constraints between them as the arcs of the graph [14], and in general, hyper-arcs. Each node of the product concept graph, called a "sub-concept," can itself be composed of sub-components thereby allowing a recursive definition. For example, a motor which is a sub-concept of a cordless electric screw driver is made up of a rotor, housing, brushes, etc. Each product concept graph represents a product family whose members can be generated with components that are specific instances of the respective sub-concepts. By rapidly generating components with different inputs, an understanding of the sensitivity of one subsystem parameter on the global performance of the entire system can be understood and used to more efficiently drive new designs.

Similarly, Process and Resource knowledge can also be captured through *concept representation* obtained from a unified representation of Product, Process and Resource of an enterprise. Various graph-based reasoning methods (e.g. graph-partitioning and graph-matching) can be used to extract useful knowledge about the enterprise, in particular, for reusing design, process and resource knowledge associated with the past projects. Now a designer could search and reuse an aerodynamic system, instead of just a turbine blade. This paper focuses on the product concept representation, and its advantages in design.

In this formulation, a concept Φ is represented as a set consisting of (1) Concept graph (Ψ), (2) Parameters (R), (3) Form definition (ξ), (4) Process definition (S), (5) Maps (M) and (6) Constraints (N), i.e.:

$$\Phi = \{\Psi, \xi, S, M, N\}$$

The concept graph is represented as a set consisting of the list of sub-concepts and their connectivity. This connectivity may include functional, operational, informational as well as physical interactions. The concept graph can be considered as a generalization of the product topology to include the design knowledge. The connectivity in the concept graph signifies the constraints posed by the interactions between the sub-concepts in fulfilling a functional requirement.

Let $\Phi = \{\Phi_1, \Phi_2, \dots, \Phi_n\}$, the set of n sub-concepts and $C = \{\Gamma, \Phi\}$, the set of constraints each of which is represented as an expression Γ involving sub-concepts Φ_c with $\Phi_c \subseteq \Phi$, then $\Psi = \{\Phi, C\}$.

Further, $R = \{R_1, R_2, \dots, R_k\}$, the set of k parameters are defined by the maps M as $R_i = M_j(\Phi, R_c)$, where $\Phi_c \subseteq \Phi$, $R_c \subseteq R$ and $M_j \in I$. Figure 5 represents an example product concept with maps, concept

graph, and parameters. A product concept graph includes components and linkages representing constraints between components. Each component may itself be a design or a sub-concept. The concepts and sub-concepts may also be recursive. It is noted that the maps and expressions could involve algorithmic evaluations such as structural analysis, or tasks involving humans.

This framework captures the constraints that need to be satisfied to create a viable product based on the concept. The generic nature of the constraints and the above framework enable rapid automated redesign of a product within the same concept. The representation of maps as a type of concept graph yields a constraint linked nodal interpretation that can be represented in a computer readable and searchable format. These representations can be called Analysis Concepts.

The constraints N are similar to C in expression, but operate only on the Concept, Φ . For example, the set N may include constraints such as $\Phi_{color} \neq Green$ or $\Phi_{Supplier1.Catalog} \vee \Phi_{Supplier2.Catalog}$ in case of a Product Concept, and constraints such as $\Phi_{iterations} \leq 500$ in the case of Analysis Concept. The process description S may include the supply chain representation for product concepts and the evaluation method for analysis concepts. This process description for analysis concepts represents the Solution Method Model developed by Peak et al. [38].

The form of a concept can be defined using a Boundary Representation (B-Rep) that consists of a tuple, $\xi = (\gamma, \tau)$ where: $\gamma = \{\gamma_1, \gamma_2, \dots, \gamma_n\}$ is the set of geometric features composing a shape, and τ is a $m \times m$ adjacency matrix that defines the topological connectivity between elements of γ . The set γ that describes the form of a concept is a set of geometric entities (surface or volume elements) that satisfy the (geometric) constraints and functional requirements of the particular concept. The adjacency matrix is defined as

$$\tau = [\tau_{uv}]_{m \times m} \text{ where: } \tau_{uv} = \begin{cases} 1 & \text{if } \gamma_u \text{ is adjacent to } \gamma_v \\ 0 & \text{if not} \end{cases}$$

For a form to exist, both γ and τ must be non-zero.

Figure 5: Example representation of a concept

The Maps M_i in general for product concepts, involve concurrent analyses across multiple domains. For example, the map $R_1 = M_1(\Phi_{CAD}, R_2)$ takes the parameter R_2 and the form definition of the sub-concept Φ as input to evaluate the parameter R_1 . Figure 6 represents the map M_2 of Figure 5 that

involves analyses A_1, A_2 , and A_3 , as well as expressions Γ and Γ . For example, if the concept in Figure 5 represents a gas turbine engine, then, Φ represents the combustor while Φ represents the turbine blade. In this way, the thermal analysis A_1 interacts with the combustor Φ , the operating temperature R_4 , and the CFD (Computational Fluid Dynamics) analysis A_2 to evaluate thermodynamic parameters. Similarly, the CFD analysis A_2 interacts with the turbine-blade Φ , structural analysis A_3 and thermal analysis A_1 to evaluate the power generated. The expressions Γ and Γ evaluate the efficiency R_2 and thrust R_5 . The input to the maps typically consists of parameters and form.

Figure 6: Analysis concept graph in the map M_2

DESIGN PROCESS IN THE CONCEPT FRAMEWORK

In this framework, the design process can be considered to follow a top-down approach during the configuration design and bottom-up approach during optimization. The design process begins by deciding the functional requirement (FRs) on sub-concepts to satisfy the required FR. This involves techniques such as functional decomposition. Many concepts exist to satisfy each of the decomposed functional requirements resulting in different product architectures. These concepts can be quickly substituted for each of the possible sub-concepts and the constraints are then formulated among them, allowing the designer rapid access to all design possibilities.

A particular choice of sub-concepts results in a single product architecture with the product concept graph describing the interactions between them. It is now required to find instances of each sub-concept that do not violate any constraints and satisfy the functional requirement. The performance parameters can now be mapped from the choice of sub-concepts and the nature of interaction. Once the choice of the sub-concepts is achieved, the form and process descriptions of the product can be designed. The form of the product could be an assembly of the forms of the components from the sub-concepts. It is noted that only the general nature of the form is initially decided. The complete set of parameters, including geometric parameters, is now available to optimize a particular instance of the concept. The optimal choice of parameter values is now used (or possibly reused) to generate the form for this instance of the Concept Architecture. In situations where the concept consists of multiple sub-concepts, the form is composed from the individual forms of the sub-concepts.

Hence, the product details are available for the manufacture of this instance of the concept.

Automating the above process can help the designer to quickly design and evaluate various architectures for a set of FRs.

Figure 7: Design process through configuration

Reuse as an Enabler to Rapid Exploration

Reuse is not well understood or practiced today due to the lack of an effective mechanism to capture and search product, process and resource knowledge. For a generic product configuration framework, an assembly-based search is critical in development of new products. The “concept search” activity retrieves all design concepts that are “similar” to a user-designed concept thereby reducing design and development time. The graph based representation facilitates search of similar concepts and topologies in a repository. Various graph matching algorithms have been proposed in the past for comparing graph structures [41-43]. Iyer et al. [39, 40] have implemented graph-matching algorithms for comparing and searching for 3D models of product components. The time complexity of graph-matching, processing graph-based indexing, and clustering techniques have not been addressed in literature for quick search. These graph matching algorithms, can be applied to concept graphs for both reuse of analysis as well as reuse of concepts.

In order to more effectively drive design it is necessary to generate and access knowledge in a timely fashion. Due to the increase in popularity of CAE packages, computerized analysis has come to represent the preferred repository for design knowledge. The ability to reuse knowledge allows for much quicker synthesis of a product design and drastically lowers the perceived amount of redesign required. This application has the advantage that it can be

scaled to fit almost any application from simple catalog queries by customers to advanced representation and searching for existing analysis models. The key enabler to knowledge extraction is the reusable, extensible representation of the knowledge.

Figure 8: Components of a Configurator

SOFTWARE IMPLEMENTATION ARCHITECTURE

In this work, a preliminary software architecture consisting of a transaction manager, unit configurator, form manager, and an analysis manager was developed. The following is a description of each component of the architecture:

1. *Transaction Manager*: The transaction manager regulates the information flow between the user and the configurators of the sub-concepts. The configurators that handle each of the sub-concepts are called ‘unit configurators’. The transaction Manager provides access to the appropriate unit configurator through a search on its database. This database contains the list of all the unit-configurators in the system along with a brief description of their product line.
2. *Unit Configurator*: The unit configurator interacts with the transaction manager to offer valid solutions to the user. It may interface with a catalog to offer products, or could generate new knowledge based on the requirements of the transaction manager. The unit configurator also provides a solution for its concept that meets the specification given as input. The logic of configuration is stored in the unit configurator that uses the internal transaction manager to generate solutions from those of its siblings. In this work, three basic types of unit configurators are identified:
 - a) *Dedicated Configurators*: The logic of configuring just a single product concept is programmed into the configurator. Any change in the product concept requires reprogramming the configurator. This type of configurator is particularly useful for designing product concepts that change very slowly with time, however require fine resolution in the solution set.
 - b) *Catalog Based Configurators*: These configurators handle standardized product concepts such as fasteners. All the product concept instances are available in a repository that is queried to provide the configuration solution.
 - c) *Concept Definition Based Configurators*: These are generic configurators that use the concept to provide configuration solutions. These configurators have internal transaction managers to connect to other

configurators for the sub-concepts to provide configuration solutions.

3. *Analysis Manager*: In general, the constraints among sub-concepts are algorithmic in nature. The analysis manager interfaces with dedicated third party solvers to execute and propagate such constraints. The analysis manager provides mechanisms for reuse of analysis to expedite the configuration process.
4. *Concept Editor*: The concept editor provides a tool for abstracting the design process. The product architecture, the topology, and analysis concepts are designed in the concept editor. The concept editor also provides a mechanism for the designer to impose/relax instance-specific constraints.
5. *Form Manager*: The Form Manager acts as a liaison between the CAD systems of the user to the configurator. The form manager helps extract physical connectivity between the sub-concepts and create a preliminary topology to which the designer can add functional constraints. The form manager also acts as a map between the parameter definitions in the concept to the solid model of the product. In this implementation of the framework, the assembly and geometric constraints are delegated to the CAD system through a mechanism provided by the form manager.

EXAMPLE CASE

Figure 9 depicts an example module of a Formula SAE (FSAE) vehicle. The designer of this car desires to achieve an objective associated with: “maximum dynamic performance from the vehicle in terms of its handling and acceleration.” However, these are competing and often mutually exclusive objectives; meaning that any attempt to increase handling through techniques such as a stiffer chassis or uniform weight distribution will lead to increased mass in the vehicle which will limit the acceleration.

For rapid design of the vehicle, all the sub-systems are to be configured simultaneously. Figure 10 shows the complete concept graph of the vehicle. In addressing this problem, a smaller subset of the design problem shown in

travel. The tie rod transfers the motion of the wheel to the shock absorber through the rocker. Various constraints ($c_1, c_2, \dots, c_{12}$) are formulated between the sub-concepts of the vehicle. For example, the constraint c_7 in

Figure 11 represents the bump-steer requirement while c_{10} represents the co-linearity constraint between the roll axes of front and rear suspensions. The chassis configurator, ergonomic configurator, and the suspension configurator are used by the vehicle configurator to provide configuration solutions. The coupling between the chassis and the suspension configurators exists as suspension mounting points on the chassis, suspension loads on the chassis, and the torsional rigidity of the chassis on the suspension.

Figure 9: Chassis and Suspension of the Purdue Formula SAE vehicle

Chassis configurator

This unit configurator is structured as a topological optimization coupled with a translator based on skeletonization to a space frame structure coupled with a space frame optimization. This configurator interacts with the packaging designer supporting various subsystems at different locations within the vehicle volume using a solid modeler package. The input geometry is initially partitioned and reduced by removing areas where the main components of the car will be attached to the chassis including the position of the driver. The constraints are geometric in nature where a set of volumes are given that

Figure 11 is configured. The chassis includes several tubes welded together as shown in Figure 9. The spatial arrangement of the tubes and the specification of each of the tubes determine its performance (weight and stiffness). The chassis is designed to consist of two parts (cockpit and the rear box) that bolt together at the roll hoop, between the driver and the engine. The suspension includes two upper control arms and two lower control arms that control the geometry of wheel

should not intersect with the chassis volume. These volumes are the envelopes of the driver seating (entry and exit), the engine and other sub-systems.

This reduced model and loads from the suspension are then automatically fed to a topological optimization using ANSYS™, with a specified percentage reduction in density and a convergence condition. The resultant densities are further processed via the Voronoi skeletonization scheme [44] which thins the density model until only lines and vertices remain. A designer is involved to further refine the generated space frame. This provides a semi-automated form-synthesis mechanism for the design of the chassis. These vertices and the choice of the tube sections are then optimized to complete the chassis configuration. Specifically, the objective is to maximize:

Stiffness (P, T)

Given: ψ , the topology connecting n points with m members, i.e.,

$$\psi(i, j) = \begin{cases} 1 & \text{if there is a member between nodes } i \text{ and } j \\ 0 & \text{otherwise} \end{cases}$$

$P^0 = \{p_1^0, p_2^0, p_3^0, \dots, p_n^0\}$, are the initial location of nodes Where:

$P = \{p_1, p_2, p_3, \dots, p_n\}$ are the locations of each node, $T = \{t_1, t_2, t_3, \dots, t_m\}$ are the selection for each member, and $S = \{s_1, s_2, s_3, \dots, s_k\}$ are the stock tubes available

Figure 10: Exploded partial concept graph of Purdue FSAE vehicle

Figure 11: Simplified concept graph of Purdue FSAE Vehicle

Typical Constraints

C_5 :-

$Natural_Frequency(FULLCARMOD\ EL, Front_Suspension, RearSuspension, Chassis) > 0^3$

C_9 :-

$(Front_suspension.UCA_F = Chassis.p12) \cap (Front_Suspension.UCA_R = Chassis.p16) \cap$

...

$(Chassis.LoadSet1 =$

$Front_Suspension.LaneChangeForces)$

Subject To:

$$t_i \in$$

$$\left| p_i - p_i^0 \right| \leq \epsilon, i = \{1, 2, 3, \dots n\}$$

$$Weight(P, T) \leq W_0, \text{ the given weight}$$

Also there exists some nodes in the structure where the chassis interfaces with the other subsystems in the vehicle such as suspension, power-train and the engine. These points are considered fixed and cannot be modified.

The loads on this structure consist of two components:

1. The static component which includes the weights of all sub-systems in the car.
2. The dynamic component that appears at the node that is attached to the suspension.

The structure is to be maximized for torsional stiffness. In this implementation, the problem is posed as minimization of torsional deflection.

A Genetic Algorithm was then implemented in Matlab™ to find the near optimal values of $\{r, \theta, \phi\}$ for each node and the index t_i of the tube selection for each member. Here r is the distance from the initial node location. θ and ϕ are the angles from the x axis (in the X-Y plane) and the Z axis respectively. The ability to execute each of these analyses is discussed towards the end of this section.

Ergonomics Configurator

The ergonomics configurator is implemented as a dedicated configurator which provides the human kinematic envelope to allow the driver to enter, exit and operate the vehicle in accordance with ergonomic constraints. This configurator interacts with the chassis configurator to minimize the center of gravity. Constraints such as driver comfort and exiting requirements in terms of time for emergency evacuation, as stipulated by the Formula SAE rules, are enforced in this configurator.

Suspension Configurator

The Suspension Configurator is designed as a concept based configurator that interacts with three other configurators to provide solutions, namely: rocker configurator, control-arms configurator, and tire configurator.

The rocker configurator is an analysis based dedicated configurator of the rocker arm, which controls the motion ratio of the FSAE car suspension between wheel travel and shock travel. The configuration begins by specifying the desired motion ratio, based on individual characteristics of the shocks. A CAD model is used to evaluate several key suspension points. These points are then fed into a kinematics model of the suspension. A single wheel travel analysis with a 2 inch range of motion is then simulated to extract the motion of the shock versus wheel travel. The derivative of this data set is then taken to determine the ratio. Optimization schemes are used to minimize the error between the desired ratio and the current one based on the inputs.

A similar dedicated configurator is used for designing the kinematics of the suspension (Control-arms configurator). The tire configurator is a catalog based configurator that uses the Pacejka Tire model to provide tire solutions from a database

of available tires. The configuration solutions provided by the chassis and the suspension configurators would then be used to provide several overall vehicle solutions through the concept based car configurator.

Configurator Characteristics and Requirements

In order to access and represent unit configurators to the analysis manager, common representations and access methods should be established. This work utilizes Isight™, an optimization package and process modeler from Engineous software [45] to execute each of the configurators and to provide optimal solutions. An example of this technique would be calling on the analysis manager to provide a valid suspension system, given input constraints, and possibly output requirements. The analysis manager would then instantiate each Isight model with the correct parameters and goals and would evaluate the individual design of each subsystem of the suspension (Rocker, A-Arms, etc). Each individual analysis code, e.g., Matlab, FEA solvers, or in-house legacy code, is then executed to provide valid solutions.

During configuration, each constraint involving analysis could be called on to: A.) provide a verification of a design where all inputs and outputs are provided, B.) to provide the output of a design given only inputs, or C.) to solve for an input with specified goals for the outputs. This demonstrates that any unit configurator should be able to evaluate, generate, or backwards solve for different combinations of inputs and outputs depending on where in the hierarchy of the design previous constraints have been imposed and on what the requirements of the analysis manager have been defined.

The large advances in computing speed and memory in the past decade have enabled large amounts of pre-computed data to be generated, sorted and stored. Once each unit configurator is defined, it can be used to generate an encapsulation of the design space using numerical exploratory methods. In this way, when the analysis manager calls for a solution, the unit configurator can determine if the solution has already been generated, and quickly reuse it

CONCLUSION

Just as mass production was crucial to manufacturing in the 20th century, rapid and reusable design methodologies will be the key to economic growth in the 21st century. Quick design, knowledge retention, and rapid product realization hold promise in realizing these goals. In this paper, a framework has been developed which supports the rapid design of complex, multi-disciplinary products through knowledge capture and reuse. Key to this framework is the unit representation of concepts, maps, forms and constraints within each concept and their associated mappings. Constraint solving methodologies combined with optimization techniques then enforce the representations and generate feasible designs. Our Purdue FSAE car example illustrates the need and application of the generic framework to solve complex engineering problems. Many individual models and solvers have been executed according to a hierarchical constraint solving methodology to propagate the customer requirements to each system. The incorporation of a process model and a supply-chain configuration and representation in the framework will further

expand the domain of our rapid product realization methodologies.

ACKNOWLEDGMENTS

We would like to thank the Purdue Formula SAE Team members for their cooperation in this work. We would also like to thank Prof. John Starkey for his valuable suggestions.

REFERENCES

- [1] Dieter, G. E., 1999, *Engineering Design: A Materials and Processing Approach*, Third edition, McGraw-Hill, New York, USA
- [2] Suh, N. P., 2001, *Axiomatic Design: Advances and Applications*, Oxford University Press, New York, USA
- [3] Boden, M. A., 2004, *The Creative Mind: Myths and Mechanisms*, Second Edition, Routledge, New York, USA.
- [4] Ko, H., and Lee, K., 1987, "Automatic Assembling Procedure Generation from Mating Conditions," *CAD*, **19**(1), pp. 3-10
- [5] Lee, K., and Gossard D.C., 1985, "A Hierarchical Data Structure for Representing Assemblies: Part 1," *CAD*, **17**(1), pp. 15-19
- [6] Lee, K., and Gossard, D.C., 1985, "A Hierarchical Data Structure for Representing Assemblies: Part 2," *CAD*, **17**(1), pp. 20-24
- [7] Takase, H., and Nakajima, N., 1985, "A Language for Describing Assembled Machines," *Design and Synthesis*, Elsevier Science Publishers B. V., North-Holland, pp. 471-476
- [8] Whitney, D. E., 1996, "The Potential for Assembly Modeling Product Development and Manufacturing," MIT, http://esd.mit.edu/esd_books/whitney/pdfs/assembly.pdf
- [9] Stuss, P., 1987, "Multiple Representation of Structure and Function," *Expert Systems in Computer-Aided Design*, pp. 57-84
- [10] Ulrich, K., 1995, "The Role of Product Architecture in the Manufacturing Firm," *Research Policy*, **24**(3), pp.419-440.
- [11] Simpson, T. W., *to appear*, "Product Platform Design and Customization: Status and Promise," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, **18**(1).
- [12] Yassine, A., Kim, K.C., Roemer, T., and Holweg, M., 2004, "Investigating the Role of Information Technology in Customized Product Design," *International Journal of Production Planning and Control*, **14**(4), pp. 422-434.
- [13] Du, X., Jiao, J. and Tseng, M.M., 2002, "Graph Grammar Based Product Family Modeling", *Concurrent Engineering: Research and Application*, **10**(2), pp. 113-128.
- [14] Jiao, J., Tseng, M.M., Duffy, V. G. and Lin F., 1998, "Product Family Modeling for Mass Customization," *Computers in Engineering*, **35**(3-4), pp. 495-498.
- [15] Mocko, G., Malak, R., Paredis, C., and Peak, R., 2004, "A Knowledge Repository for Behavioral Models in Engineering Design", *Proc. of DETC'04 : 24th Computers and Information Science in Engineering Conference*, September 28-October 3, 2440, Salt Lake City, Utah.
- [16] Michelena, N. F., Kim, H. M., Papalambros P. Y., and Jiang, T., 2000, "Target Cascading in Optimal System Design," *Proc. 2000 ASME Design Automation Conference*, DAC paper 14253, Baltimore, September 10-13, 2000
- [17] Kim, H. M., Rideout, G. D., Papalambros, P. Y., and Stein, A. J., 2001, "Analytical Target Cascading in Automotive Vehicle Design," *27th Design Automation Conference*, September 9-12, 2001, Pittsburgh, Pennsylvania.
- [18] Yin, S., and Cagan, J., 2000, "An Extended Pattern Search Algorithm for Three-Dimensional Component Layout," *Transactions of the ASME* **122**, pp. 102-108.
- [19] Cagan, J., Degentesh, D., and Yin, S., 1998, "A Simulated-Annealing Based Algorithm Using Hierarchical Models for General Three-Dimensional Component Layout," *Computer-Aided Design*, **30**(10), pp. 781-790.
- [20] Szykman, S., and Cagan, J., 1997, "Constrained Three-Dimensional Component Layout Using Simulated Annealing," *ASME Transactions* **119**, pp. 28-35.
- [21] Sinha, R., Paredis, C. J. J., Liang, V. C., and Khosla, P. K., 2001, "Modeling and Simulation Methods for Design of Engineering Systems," *Transactions of the ASME* (1), March, 2001.
- [22] Coutinho, M. G., Eleish, R., Kim, J., Kumar, V., Ling, R., Neches, R., and Will, P., 1998, "Active Catalogs: Integrated Support for Component Engineering," *Proceedings of DETC '98 - ASME Design Engineering Technical Conferences*, Atlanta, GA, September 13-16, 1998.
- [23] Ling, S. R., Kim, J., Will, P., and Luo, P., 1997, "Active Catalog: Searching and using Catalog Information in Internet-based Design," *Proceedings of DETC '97 - ASME Design Engineering Technical Conferences*, Sacramento, California, September 14-17, 1997.
- [24] Kim, J., Ling, S. R., Will, P., 1997, "Ontology Engineering for Active Catalog," *Using AI in Electronic Commerce, Virtual Organizations, and Enterprise Knowledge Management to Reengineer the Corporation: Papers from the 1997 AAAI Workshop*, July 28, 1997, Providence, RI. AAAI Press, Technical Report WS-97-02, 1997.
- [25] Gunter, A., and Kuhn, C., 1999, "Knowledge based Systems - Survey and Future Directions," *5th Biannual German Conference on Knowledge Based Systems*, Proceedings Springer, Lecture Notes in Artificial Intelligence 1570, 1999.
- [26] Schmidt, L. C., and Cagan, L., 1998, "Optimal Configuration Design: An Integrated Approach using Grammars," *Journal of Mechanical Design* **120**(1), pp. 2-9.
- [27] Yu, B., and MacCallum, K.J., 1995, "Decision Support for Configuration Management in Product

- Development,” *Proceedings of the third International Conference on Computer Integrated Manufacturing*, Singapore, pp. 278-286.
- [28] Heinrich, M. and Jungst, E. W., 1996, “A Resource-Based Paradigm for the Configuring of Technical Systems from Modular Components,” *AAAI-96 Fall Sympos. Series: Configuration*. MIT, Cambridge, MA, November 9-11, pp. 19-27.
- [29] Nakayama, H., Arakawa, M. and Sasaki, R., 2002, “Simulation based Optimization Using Computational Intelligence,” *Optimization and Engineering*, 3, pp. 201-214.
- [30] Feldkamp, F., Heinrich, M. and Gramann, K., D., 1998, “SyDeR-System design for reusability,” *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 12, pp. 373-382.
- [31] Lin, L., and Chen, L. C., 2002, “Optimization of Product Configuration Design Using Functional Requirements and Constraints,” *Research in Engineering Design*, 13, pp. 167-182.
- [32] Felfernig, A., Friedrich, G., Jannach, D., and Stumptner, M., 2000, “Exploiting Structural Abstractions for Consistency-based Diagnosis of Configurator Knowledge Bases,” *14th European Conference on Artificial Intelligence (ECAI'2000) - configuration workshop*, Berlin, Germany, pp. 23-28.
- [33] Felfernig, A., Friedrich, G., Jannach, D., and Zanker, M., 2000, “Towards Distributed Configuration,” *Proceedings of the Joint German/Austrian Conference on AI: Advances in Artificial Intelligence*, Springer – Verlag, London, pp. 198 – 212.
- [34] Felfernig, A., Friedrich, G., Jannach, D., and Zanker, M., 2000, “Integration of Distributed Constraint-Based Configurators,” *14th European Conference on Artificial Intelligence (ECAI'2000) - configuration workshop*, Berlin, Germany, pp. 29-34.
- [35] Felfernig, A., Friedrich, G., Jannach, D., and Zanker, M., 2002, “A Joint Foundation for Configuration in the Semantic Web,” *Proceedings 15th European Conference on Artificial Intelligence - Configuration Workshop*, Lyon, France.
- [36] Felfernig, A., Friedrich, G., Jannach, D., and Zanker, M., 2002, “Configuration Knowledge Representation using UML/OCL,” *Proceedings Fifth International Conference on the Unified Modeling Language - the Language and its applications UML'2002*, Dresden, Germany.
- [37] Fleischanderl, G., Friedrich, G., Haselbock, A., Schreiner, H., and Stumptner, M., 1998, “Configuring Large Systems Using Generative Constraint Satisfaction”, *IEEE Intelligent Systems* 13(4): pp. 59-68.
- [38] Peak, R. S., Fulton, R. E., Nishigaki, I., and Okamoto, N., 1998, “Integrating Engineering Design and Analysis Using a Multi- Representation Approach,” *Engineering with Computers*, 14(2), pp. 93-114.
- [39] Iyer, N., Kalyanaraman, Y., Lou, K., Jayanti, S., and Ramani, K., 2003, “A Reconfigurable, Intelligent 3D Engineering Shape Search System Part I: Shape Representation,” *ASME DETC, Computers and Information in Engineering (CIE) Conference*, 2003.
- [40] Iyer, N., Kalyanaraman, Y., Lou, K., Jayanti, S., Ramani, and K., Prabhakar, S., 2003, “A Reconfigurable, Intelligent 3D Engineering Shape Search System Part II: Database Indexing, Retrieval and Clustering,” *ASME DETC, Computers and Information in Engineering (CIE) Conference*, 2003.
- [41] Ullman, J.R., 1976, “An Algorithm for Sub Graph Isomorphism,” *Journal of the Association for Computing Machinery* 23, pp. 31-42.
- [42] Messmer, B. and Bunke, H., 1995, “Subgraph Isomorphism in Polynomial Time,” *Technical Report TR-IAM-95-003*.
- [43] Ambler, A. P., Barrow, H. G., Brown, C. M., Burstall, R. M., and Popplestone, R. J., 1996, “A Versatile Computer-Controlled Assembly System,” *Proc. of International Joint Conference on Artificial Intelligence*, pp. 298-307.
- [44] Ogniewicz R. L., Kubler O., 1995, “Hierarchical Voronoi Skeletons,” *Pattern Recogn*, 28(3); pp. 343-359
- [45] <http://www.engineous.com/index.htm>