An Approach to Capturing Structure, Behavior, and Function of Artifacts in Computer-Aided Design

Lisa Anthony  
e-mail: lpa22@drexel.edu

William C. Reglia)  
e-mail: regli@drexel.edu

Jon E. John  
e-mail: jej22@drexel.edu

Santiago V. Lombeyda)  
e-mail: slombey@caltech.edu

Geometric and Intelligent Computing Laboratory,  
Department of Mathematics and Computer Science,  
Drexel University, 3141 Chestnut Street, Philadelphia, PA 19104

This paper presents an approach to computer-aided design (CAD) that unites ideas from design with three-dimensional layouts and knowledge engineering. Our goal is to capture the structure, behavior, and function of CAD artifacts. We describe a software tool based on this approach, the conceptual understanding and prototyping (CUP) environment, for capturing the design intent inherent in the design process and authoring design semantics in previously created artifacts. CUP records design ideas, based on functional, geometric, and knowledge-based relationships among components in an electromechanical assembly. This design knowledge is stored using ontologies defined in XML. The goal of this work is to enable users to navigate intricate product data and design knowledge bases. [DOI: 10.1115/1.1385826]

Introduction

Over the past two decades, Solid modeling and three-dimensional (3D) computer-aided design (CAD) have become critical elements of the product realization process, leading to successive generations of increasingly robust software tools for detailed design, simulation, and engineering analysis. This paper presents our approach and software system to support archival of artifact semantics: conceptual understanding and prototyping (CUP).

With CUP, users specify a spatial layout of components and subassemblies, as well as structural, behavioral, and functional (SBF) information about components and subassemblies. CUP also provides mechanisms for capturing textual information about the designer’s intent and preferences. In this way,

1. CUP is an environment that enables users to document structural information in a top-down fashion and record the function, structure, and behavior of the intended artifact and its subcomponents; and
2. CUP provides a tool for design teams to digitally describe the high-level semantics of an artifact, simultaneously creating the logical description of the device using domain ontologies and the 3D spatial layout of the individual components.

CUP combines several new trends in information technology from the areas of product data representation and product knowledge sharing with Java, Java 3D, and extensible markup language (XML)—providing a knowledge-based approach to support small teams of engineers in defining the semantics and layout for new products. CUP creates a structured way in which product data semantics can be described as designs evolve, extending computer support to the very early phases of product development, as well as improving interfaces between downstream CAx tools and applications.

Related Research

Pahl and Beitz, often considered the definitive treatment of engineering design [1], present conceptual design as an iterative, evolutionary, top-down process that ultimately yields a principle solution or concept for the product. The most effective way of developing the functional requirements of a product in great enough depth is by modularizing the problem via a process known as “functional decomposition” [2]. Given an overall function of the product, designers and engineers will break this into multiple subfunctions to be treated as individual subproblems.

Functional modeling systems most often take a graph-based (nodes and edges) [3–5] approach to describing the engineering relationships among the elements in a mechanical assembly model, or other CAD-based structure [6]. Sketch and layout-based approaches to conceptual design cover a diverse collection of domains and involve research contributions from CAD, computer vision, computer graphics, and a number of engineering disciplines. Conceptual sketching has been explored as a way to develop user interfaces [7], architectural drawings [8,9], and 3D solid models for CAD [8,10–15]. One approach is to make these systems feature based, where domain-specific features can be exploited to drive the recognition and refinement of the conceptual design into a detailed 3D model [7,16–18].

Relationship of CUP to Existing Research. While there are many tools for detailed design (e.g., the major commercial CAD

a)URL: http://gicl.mcs.drexel.edu/regli.
b)Present Address: Center for Advanced Computing Research, Mailstop 158-79, 1200 E. California Blvd., California Institute of Technology, Pasadena, CA 91125.

Contributed by the Computer Aided Product Development Committee for publication in the JOURNAL OF COMPUTING AND INFORMATION SCIENCE IN ENGINEERING. Manuscript received January 2001; revised May 2001. Associate Editor: P. Wright.
semantics were usually imposed a priori, before the development of agents could begin. In the expert systems era, semantics were often hard coded—resulting in brittle systems and narrow problem solutions. The need for common knowledge-sharing standards has been cited as a major obstacle to achieving true concurrent engineering [24].

The current trend is to support the development of shared semantics. Paralleling the evolution of the Internet and its protocols, contemporary work in agent-based computing and knowledge representation seeks to create formal frameworks and tools for the rapid formation, modification, and integration of ontologies and knowledge bases. In much the same manner as a team of digital agents works toward a common goal, a team of designers progresses toward a final design in part by achieving collective group semantics.

CUP takes an analogous approach: a team of designers engages in a collaborative problem-solving exercise in order to define the semantics of a new product. With CUP the design process is one in which designers author an assembly design, specifying its structure and layout in 3D while annotating the relationships among all the entities in the assembly with reference to base ontologies. The base, or upper-level, ontologies CUP employs describe fundamental structure–behavior–function properties common in mechanical design systems. Hence, as designers create the 3D “back of the envelope” sketch of their design, they also are authoring the set of logical sentences that describe the semantics of the assembly with reference to these base ontologies. It is important to note that this approach, and the CUP system, is not specific to any one design domain. To modify CUP to perform design for the architecture/engineering/construction (AEC) domain, one would simply need to provide suitable 3D conceptual design primitives and references to appropriate base ontologies for describing relationships in the AEC world. This approach is similar to that favored by the knowledge engineering community for development of large, shared knowledge bases [25].

Shown in Fig. 1, users of CUP author design knowledge using 3D primitives and existing components from design repositories while describing engineering semantics with a domain ontology and a shared upper-level ontology. The output of a completed CUP session is a product knowledge base that includes an XML-based description of product structure, behavior and function with respect to the shape primitives and their relationships as described with the ontologies.

Technical Background. Designers often model preliminary designs without having detailed specifications for every component and feature, being more concerned with specifying design intent and the interrelationships between the components and fea-
CUP’s basic approach to the support of this activity involves some preliminary CAD capabilities: a user defines the assembly’s major components in three dimensions, arranging the primitive shapes in a layout which makes logical sense at this early stage of design. As discussed in the Related Research section, the emphasis is on the function of the components in the assembly and/or the flow of a substance or material from one or more components to some other. That is, the laying out and defining of the structural, behavioral, and functional relationships in the assembly are more salient at this point in the design process than geometry, size, shape, rotation, even relative orientation. It is the structure–behavior–function knowledge, more than the geometry and topology, that encodes the designers’ intent. By capturing this intent, we can search design knowledge bases for related information and create proactive design tools that can guide this search of the design space.

To represent the concept that lies behind the preliminary layout the user constructs, users “mark up” the design as they create the 3D layout of physical components. In this way, CUP captures a semiformal notion of design intent. For example, why does the shaft connect to the motor? What does the link between the axle and wheel 1 represent? The semantics of these queries and their answers can be expressed in terms of the incorporated ontologies.

A perfunctory survey of the literature regarding functional mapping and representation in assembly design [20,26–32] presented
three basic, common entities: function, that is, what an object (component, group of components, or subassembly) does within the overall assembly; behavior, the observable actions by which it accomplishes this function; and structure, specifically, what attributes of the physical object (e.g., physical constraints, materials, and so on) are related to helping it achieve this behavior. With function, some literature [26,32] also mentioned an entity known as flow, the substances or materials which are the inputs and/or outputs of the object’s function.

System Overview and User Interface. CUP consists of a very elemental CAD system in which the user designs without focusing on details, yet still defines enough information that a full design can evolve from this work. For this purpose, the CUP environment supports the creation and manipulation of a set of primitive three-dimensional objects (i.e., blocks, cylinders, spheres, and frustums) and the importation of more complex shape entities (i.e., VRML97 data structures, such as indexed facets). The user can assign further characteristics of these objects textually, as well as correlate relationships between these objects. This is accomplished by offering to the user a limited CAD environment with the abilities to tag, link, and group the objects. A simple example of an object created in CUP and the relationships among its components is shown in Fig. 2(a). In Fig. 2(b), the internal data model of the assembly is depicted. Arrowed-edge connections indicate function–flow relationships among the components; plain-edge connections show the internal structure of the 3D world and its objects.

The CUP interface consists of the central pane and a Java 3D-based viewer capable of displaying a design (the CUP “world”). There are controls in menus and toolbars for creation and deletion of primitives, groups, and links; editing of object properties such as scale, rotation, color, and position; and editing of object tags. Also available is a two-dimensional tree view of the current world. This tree view, shown in Fig. 3, displays the hierarchical structure of the world’s objects and groups, and offers the capability to view and edit tags from this window.

All objects in the design space have a one-to-one relationship with a tag containing information about their structure, behavior, and function, as well as text-based descriptions about the components. Objects may be linked to each other with either unidirectional or bidirectional links that establish functional ties among components. An example of the linking and tagging facility is shown as part of Fig. 4. Finally, a selection of multiple objects may be grouped to create a conglomerate which has a tag of its own.
Representation of Structure, Behavior, and Function. CUP has been developed so that users can “plug in” their own representation schema. Our implementation was tested using a taxonomy proposed by Szykman et al. [26] of the National Institute of Standards and Technology (NIST) regarding the representation of function in engineering design. In this prestandard, function and flow are defined as the two main quantities that are necessary in order to represent design intent properly. The novel aspect of the NIST work is that it defines a representation based on the separation of function and flow into two distinct categories. Reasons given in their work for this decision are to eliminate duplicate entities within the taxonomy itself, to facilitate changing any component of an artifact’s function independently of the others, and to promote the inclusion of functions which are not associated with a flow. These three reasons are referenced in their work as weaknesses in taxonomies which appeared in the literature prior to their own work.

NIST has specified a relatively complete schema for both function and flow within the mechatronic domain. Function is the actual function performed by any given component of an assembly, or by a link between two or more components. Most functions have an associated flow. Flow is the definition of what material or quantity is being transferred by its associated function, and in which direction it is being transferred (i.e., from which component to which component).

At present, we use NIST’s taxonomies for both function and flow, which provide over 130 functions and over 100 flows, respectively. In our system’s data representation, we have included

Fig. 7 Design process of the missile seeker within CUP
two additional quantities: structure, or the construction or shape of the component, and behavior, or the observable action of the component. Thus, users of CUP specify any or all of these quantities in a SBF representation of their designs.

Once an assembly is created and its pertinent SBF information defined, it becomes necessary to consider how this information could be stored. Our system uses the XML [33], a dynamic language similar to HTML and SGML for the structured representation of data with user-defined tags. The robust extensibility of XML makes it a perfect fit with our need for a hierarchical semantically embedded representation of design data. Figure 6 shows two pieces of XML code that were created by CUP in reference to the drive train model shown in Fig. 2. In Fig. 5(a), the code shown pertains to the circled primitive, a cylindrical shape whose function is to perform rotational motion. In Fig. 5(b), the circled link is the representation of the rotational motion transferred by the relationship from the axle to the rear wheel.

Implementation Specifics. The Java 3D internals, which are based on the virtual reality modeling language (VRML) structure, are shown in Fig. 6. At the top level there is a root branch group with all of the “children” in the VRML world attached to it. These children may be other branch groups, transform groups, and various leaf nodes, which may contain CUP shapes, CUP groups, and CUP links.

CUP is based on cross-platform technologies: the virtual reality modeling language and Sun Microsystems’ Java language. The intention is to create a downloadable tool which the user can run inside the Java Runtime Environment software from Sun on a wide variety of platforms. CUP currently runs on Pentium II/III-based machines with Microsoft Windows NT 4.0 and Sun Ultra workstations with Solaris 2.6/7 that are equipped with Java Runtime Environment version 1.2 and the VRML97, Java3D, Java Help, and XML 1.0 extensions.

CUP model worlds are saved in XML format (.xml), structuring the SBF information about each component of the model within the model’s file. CUP can also import geometries defined in VRML format (.wrl)—such as those created by other CAD systems—and act as a SBF editor for CAD models which otherwise would not store this information. This import functionality is shown in the later examples, and in Fig. 8.

Using CUP: Potential Scenarios

The following scenarios demonstrate the applications of CUP to the design world.

Using CUP During the Design Process. A simplified missile seeker assembly [34] might be one of many dozens of designs stored in a corporate design database or knowledge base. A design team, faced with the task of creating a new seeker, might want to interrogate the CAD knowledge base and examine previous design cases that might be relevant to this new problem—starting points for variant design.

The design scenario shown in Fig. 7 demonstrates how CUP might be used by a designer to describe quickly in 3D the major components and structural relationships in the assembly. Figure 7 lays out the step-by-step process from component design to functional description to final assembly. The resultant conceptual design then becomes the basis for detailed CAD (in this case, in Bentley’s Microstation Modeler). Rather than performing detailed CAD modeling to create a draft design (which for this model took several days), designers can build a design in a matter of minutes. In this way, CUP could help designers capture the design intent and build a structure—behavior—function model of the artifact to be created. Finally, this CUP design can be used as a starting point for further refinement or as a query by example to a design knowledge base, in which the components and relationships in the query are compared against those digitally archived [35].

Using CUP to Augment Traditional CAD. Most traditional CAD environments support detailed design, where the precise product shape is specified. For these environments, CUP can be used to augment shape and assembly data with engineering semantics. CUP is able to import complex geometries (i.e., in VRML format), and annotate them with descriptions of structure, behavior, and function. In this way, CUP is used to describe the engineering knowledge implicit in traditional CAD tools. Shown in Fig. 8 is an example of annotating the design of a computer mouse from SDRC’s CAD suite, I-DEAS Master Series. The output of I-DEAS is a detailed solid model of the design, along with assembly relationships and kinematic constraints. CUP can then be used to import the shape models and allow users to author the knowledge-level SBF information for these designs. In Fig. 8(a), the electromechanical design of a computer mouse is shown in SDRC’s I-DEAS. The SDRC system encodes the shape and assembly design knowledge; when imported into CUP [Fig. 8(b)] the semantics of this design can be described [Fig. 8(c)] using the domain ontologies loaded into CUP.

Conclusions

This paper described and demonstrated an approach to knowledge-based design that supports small teams of engineers in defining the semantics and layout for new products. We have implemented a tool to demonstrate this approach: CUP, a system for the computer-aided design and semantic annotation of assemblies. In the development of this tool, we have created a 3D modeling system and introduced methods for integrating the descri-

---

3CUP is available for download and use at http://edge.mcs.drexel.edu/CUP. Some example models are also available.
tion of formally represented engineering knowledge (function and behavior) with 3D graphical modeling. We believe that CUP, and tools like it, will become essential components of future design environments, enabling users to create a knowledge-level description of the design without having to perform detailed CAD and solid modeling [36].

We are extending our approach and the CUP system in several ways. Future work includes enabling multiuser design with CUP, with shared 3D JavaSpaces; and enabling access to design repositories [35,37–39], helping designers navigate large-scale engineering digital libraries and knowledge bases of engineering information and solid models. It is an eventual goal to integrate CUP with the National Design Repository (http://www.designdepository.org) and provide facilities for users to add their own ontology information to the Repository’s knowledge base.

While continuing our research, it is important to note that there are many open research issues in conceptual design that will require input from the wider community. One problem in particular is that of design validation: given the high-level functional description, such as that provided by a tool like CUP, can we create design validation or compilation software that will check the composed system with respect to design requirements? A second major community-wide problem is the lack of shared semantics and knowledge representations [36]. While standards such as STEP, and provide facilities for users to add their own ontology information to the Repository’s knowledge base.

Acknowledgments
This work was supported in part by National Science Foundation (NSF) CAREER Award No. CISE/IRIS-9733545 and Grant No. ENG/DMI-9713718; additional support was provided by AT&T Laboratories. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation or the other supporting government and corporate organizations. The authors would also like to thank the ASME JCISE reviewers for their many helpful comments.

References