Computer-Aided Design 44 (2012) 879–900

Contents lists available at SciVerse ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad

Cognitive, collaborative, conceptual and creative — Four characteristics of the next generation of knowledge-based CAD systems: A study in biologically inspired design

Ashok K. Goel [a,b], Swaroop Vattam [a], Bryan Wiltgen [a,*], Michael Helms [a,b]

[a] Design & Intelligence Laboratory, School of Interactive Computing, 85 Fifth Street NW, Atlanta, GA 30308, USA

[b] Center for Biologically Inspired Design, Georgia Institute of Technology, 85 Fifth Street NW, Atlanta, GA 30308, USA

ARTICLE INFO

Keywords:
AI in design
Biologically inspired design
Biomimcry
Bionics
Conceptual design
Collaborative design
Creative design
Design cognition
Computer-aided design
Knowledge-based design

ABSTRACT

We envision that the next generation of knowledge-based CAD systems will be characterized by four features: they will be based on cognitive accounts of design, and they will support collaborative design, conceptual design, and creative design. In this paper, we first analyze these four dimensions of CAD. We then report on a study in the design, development and deployment of a knowledge-based CAD system for supporting biologically inspired design that illustrates these four characteristics. This system, called DANE for Design by Analogy to Nature Engine, provides access to functional models of biological systems. Initial results from *in situ* deployment of DANE in a senior-level interdisciplinary class on biologically inspired design indicates its usefulness in helping designers conceptualize design of complex systems, thus promising enough to motivate continued work on knowledge-based CAD for biologically inspired design. More importantly from our perspective, DANE illustrates how cognitive studies of design can inform the development of CAD systems for collaborative, conceptual, and creative design, help assess their use in practice, and provide new insights into human interaction with knowledge-based CAD systems.

Published by Elsevier Ltd

1. Next generation CAD systems

Computer-aided design (CAD) encompasses a broad area of scholarship focused on supporting design processes that shift and adapt even as the underlying computational technology is evolving. Given the challenge of defining the next generation of CAD, we prefer to focus on intelligent CAD, i.e., CAD that develops and deploys artificial intelligence (AI) techniques. In particular, we want to focus on knowledge-based CAD that investigates the content, representation, organization, access, use, acquisition, communication, and sharing of knowledge in CAD. We use the term "knowledge-based CAD systems" to contrast and separate it from other CAD paradigms based, for example, on computational geometry, computer graphics, connectionist networks, evolutionary computing, numerical analysis, optimization, simulation, etc. We propose that the next generation of CAD in general, and knowledge-based CAD in particular, will be defined by four characteristics: they will be based on cognitive accounts of design, and they will support collaborative design, conceptual design, and creative design. Thus, we will call the next generation the 4C generation, where the four C's in 4C stand for Cognition, Collaboration, Concepts, and Creativity.

1.1. Three generations of knowledge-based CAD systems

Before we describe the four C's of the next generation of knowledge-based CAD systems, let us briefly review the previous generations. From the perspective of AI in Design, the first generation of knowledge-based CAD systems, developed between the late 1970s and the early 1990s, typically focused on the task of configuration design [1] in engineering domains, e.g., [2–4]. Some first generation knowledge-based CAD systems became quite influential, e.g., R1 [5], AIR_CYL [6], PRIDE [7], VEXED [8], VT [9], and XCON [10]. The rule-based R1 that configured computer systems, and the industrial version of R1 called XCON, are especially noteworthy for their impact. Finger and Dixon [11,12] provide useful summaries of early work on knowledge-based design. Tong and Sriram's [13] three-volume anthology describes many early knowledge-based systems in detail. Dym [14] provides a new synthesis of engineering design from the perspective of this generation of knowledge-based CAD systems.

* Corresponding address: School of Interactive Computing, Georgia Institute of Technology, Technology Square Research Building, 85 Fifth Street NW, Atlanta, GA 30332, USA. Tel.: +1 404 894 4994; fax: +1 404 894 0673.

E-mail addresses: goel@cc.gatech.edu (A.K. Goel), svattam@cc.gatech.edu (S. Vattam), bwiltgen@cc.gatech.edu (B. Wiltgen), mhelms3@cc.gatech.edu (M. Helms).

0010-4485/$ – see front matter. Published by Elsevier Ltd
doi:10.1016/j.cad.2011.03.010

The second generation of knowledge-based CAD systems, developed between the late 1980s and the early 2000s, advanced four major research themes: design databases, case-based reasoning, model-based reasoning, and visual reasoning in design. Research on CAD databases led to the development of techniques for constructing and maintaining digital design libraries (e.g., [15–17]). Research on development of case-based reasoning in design led to many case-based design systems including (alphabetically) Archie [18,19], ARGO [20], AskJef [21], BOGART [22], CAB-Assembly [23], CADET [24]; CADRE [25]; CASCAD and CADSYN [26]; CYCLOPS [27]; Déjá vu [28]; FABEL [29]; IDIOM [30], KRITIK [31,32], and STRUPLES [33]. These case-based systems addressed a variety of design tasks in domains ranging from engineering to architecture to software design. Maher and Pu's [34] anthology of case-based systems contains descriptions of many case-based design systems. Goel and Craw [35] review early case-based design systems. Note that this research theme also included analogical design systems such as IDeAL [36,37]. In case-based design, the target design problem and the source design case are so similar that the design case can be adapted to meet the requirements of the target problem; in analogical design in general, the target design problem and the source design case are different enough that addressing the target problem requires abstraction of design principles and patterns from the source case for transfer to the target problem. Research on development of model-based reasoning in design led to several schemes for functional modeling of engineering systems including Functional Representation [38,39], Function-Behavior-Structure [40,41], Function-Behavior-State [42–44], and Structure-Behavior-Function [45–47]. Some systems, such as CADET [24], KRITIK [31,32], and IDeAL [36,37] even combined case/analogy-based and model-based reasoning. Finally, research on visual reasoning in design developed techniques for analysis of shapes and motion of kinematic mechanisms on a plane [48–50] as well as methods for accessing design drawings similar to an input drawing [51,52]. Stahovich [53] and Tomiyama [54] provide reviews of the second generation of knowledge-based CAD systems.

It is a little harder to identify similar research themes in the third generation of knowledge-based CAD systems developed between the late 1990s and the present (2011) in part because we are still in the middle of it and partly because there are apparently several such themes. One theme clearly is the development of sharable design repositories [55,56] as well as ontologies for building such repositories (e.g., [57–61]). Another research theme is the development of multimodal reasoning, entailing both visual/analogical and symbolic/propositional representations, for example, for deriving behavioral models from design sketches [62] or design drawings [63], and for analogical transfer of design plans [64]. Yet another research theme is the use of machine learning techniques for a wide variety of design tasks and subtasks [37,65–69]. Looking at the landscape of third-generation knowledge-based CAD systems, two important patterns stand out. First, the techniques used in these systems have so deeply permeated into the design disciplines that often we do not even recall that many of these ideas, theories and methods originated in AI. Second, important contributions are coming from many disciplines, such as architecture and engineering, not just from computer science or cognitive science. These two patterns highlight how the development of knowledge-based CAD systems is increasingly becoming a multi-disciplinary activity.

1.2. Four characteristics of next generation knowledge-based CAD systems

We now examine each of the four C's that we believe will characterize the next generation of knowledge-based CAD systems. It is important to note that while the first C refers to a methodology for developing CAD systems, namely, grounding the design, development and deployment of CAD systems in cognitive studies, the other three C's more properly refer to characteristics of design that CAD systems may support. The first C, then, deals with cognitive studies of design. Since design is a wide-ranging and open-ended cognitive activity, the design literature contains a large number of cognitive studies of design, for example [70–90], to mention just a few. These cognitive studies use a variety of methods ranging from protocol analysis to neuroimaging, from controlled experiments to ethnographic studies. The goals of these studies also vary from studying brain activations to memory processes and knowledge structures to design behaviors of individual designers and design teams. All these different paradigms add much value to our understanding of design cognition. In our own lab, we are specifically interested in observational and behavioral studies of designers and designing. We use *in situ* studies to provide insights into situated and distributed design behaviors in naturalistic settings.

It is unclear if any of the previous three generations of knowledge-based CAD systems were informed by any cognitive study of design or evaluated through a systematic study of their use, let alone validated through cognitive models that explain behavioral data—we expect the proportion to be quite small. We conjecture that in general the first generation of knowledge-based CAD systems were almost completely unrelated to cognitive studies of design; that some second generation projects at least partially and informally evaluated the knowledge-based CAD systems through studies of their use; and that even third generation systems were not typically informed directly by cognitive studies of design.

We certainly do not mean to be overly critical of previous generations of knowledge-based CAD systems (which include our own work). The goal of the pioneering systems of the first-, second-, and third-generation knowledge-based CAD systems was to *explore* the *development* of knowledge-based CAD technology. Few of these knowledge-based CAD systems were actually *deployed* in practice. While the next generation of knowledge-based CAD systems surely will continue to explore new methods and technologies, we believe that it will also *exploit* what we already know and will learn to deploy CAD tools in communities of design practice. For these tools to succeed in practice, they will need to blend into the natural flow of designing and designers. It follows that cognitive studies of design practice, conducted before, during, and after the development and deployment of CAD systems, are powerful techniques to improve the success of knowledge-based CAD systems in this regard.

The second C is about collaboration. Design in general is collaborative (e.g., [91,92]). In fact, design is collaborative in at least four dimensions: (1) design is collaborative by time: new design knowledge builds on inherited knowledge, and new designs typically are generated by adapting, transferring and composing old designs, (2) design is collaborative by space: design teams often are geographically distributed, (3) design is collaborative by discipline: design of complex engineering systems, such as deep sea oil drilling, is fundamentally multidisciplinary, and (4) increasingly, design is also collaborative by culture: members of a multidisciplinary, geographically-distributed design team may come from many societies and cultures. Not surprisingly, collaborative design has received significant attention in CAD (e.g., [93–99]). We believe that collaborative design will continue to be a key characteristic of the next generation of knowledge-based CAD systems both because of the fundamentally collaborative nature of design and because of the advent of the Internet and the World Wide Web. We have already noted the development of web-based design repositories and ontologies as part of the third generation knowledge-based systems. While CAD support for design collaboration is already well underway, design is becoming increasingly virtual and global [100]. The next generation of CAD systems must support not only collaboration across space, time, different disciplines, methods, and vocabularies, but also collaboration across different cultures characterized by different perspectives, values, and norms.

The third C is conceptual design. Although there is little agreement in the design literature on the number of design phases in the design process, let alone the precise specification of the design phases, there is a broad consensus (e.g., [9,101,102]) that (1) there in fact are distinct design phases such as preliminary design, conceptual design, geometric modeling, simulation, optimization, prototyping, and so on [103], (2) design is iterative and situated, and thus there is no linear, or even fixed, ordering among the design phases, and (3) design ideation and conceptual design occur early in the design process and thus are especially important.

Since the design literature contains many different characterizations of design ideation and conceptual design, few of which are very precise, it may be useful to note that our characterizations generally are based on [9,102]. For the purposes of the present discussion, it is sufficient to note that the task of design ideation has been described as taking as input the specification of a desired function and giving as output a design concept that specifies the causal mechanisms for achieving the function. Similarly, for the present discussion, it is sufficient to describe the task of conceptual design as taking as input a specification of functional requirements (and possibly other constraints), and giving as output a specification of the structure of one or more designs that purportedly meets some set of functional requirements and constraints. Note that design ideation may either precede or overlap conceptual design. The main point here again is that these tasks typically occur early in the design process.

Although there has been a significant amount of research on CAD for design ideation (e.g., [86,104]) and conceptual design (e.g., [105–107]), we believe that they have not yet received the sustained and focused attention they deserve. This probably is in part because of rapid advances in technologies well-suited for other design phases such as geometric and solid modeling, and numerical analysis simulation, and optimization. However, the tasks of design ideation and conceptual design are critical to the design process: in general, a small change in an upstream task such as conceptual design may have a much larger economic and ecological impact than a small change in a downstream task such as, say, assembly. Thus, as economic and environmental pressures on design continue to mount, we expect that conceptual design will become a primary focus of the next generation of knowledge-based CAD systems.

Finally, the fourth C is creativity. It has been said many times in the design literature that design can be routine, innovative, or creative [108], even though these categories often are imprecise. Brown and Chandrasekaran [6], for example, suggested that (1) in routine design, both the basic structure of the desired system and the plans for selecting the parametric values of each component were known, (2) in innovative design, only the structure of the system was known and the plans for selecting component parameter values were unknown, and (3) in creative design, the structure of the design itself was unknown. In our own earlier work on case-based design [31,32], we have proposed that (1) in routine design, the modifications needed to adapt a known design into the desired design are limited to values of parameters of components in the design, (2) in innovative design, the needed modifications pertain to the components of the design, and (3) in creative design, the modifications entail changes to the topology of the design itself. Accordingly, it seems fair to say that the first generation of knowledge-based CAD exemplified by, say, AIR_CYL [6] in general addressed routine design problems; the second generation exemplified by, say, KRITIK [31,32] started addressing innovative design problems; and IDeAL [36] at the intersection of the second and third generations started addressing creative design problems. Creativity, including design creativity, is a precious intellectual resource of all countries and cultures. Thus, building on the past trend, we believe that although the next generation of knowledge-based CAD systems will support all kinds of routine, innovative and creative design problems, it will increasingly emphasize creative design.

1.3. Operationalizing the Four C's

Thus far we have talked about some characteristics of the next generation of knowledge-based CAD systems: collaborative, conceptual, and creative design tools based on cognitive accounts of design. While this discussion provided some guidance on the required methodology and functionality of a next generation CAD system, operationalizing these requirements is a different matter altogether. Putting these requirements into practice depends on several things, such as a design context that allows for cognitive studies to be conducted.

We acknowledge that CAD will surely explore many more topics and themes (as some reviewers of this paper pointed out). For example, CAD paradigms other than knowledge-based AI will continue to play a major role. This includes not only existing CAD paradigms such as computational geometry and computer graphics, but also new paradigms of computing such as tangible computing, smart matter, and cyber-physical systems. Further, even in the knowledge-based AI paradigm, research on CAD surely will investigate other issues such as inception, intuition, and invention. We focus on the four C's of cognition, collaboration, conception, and creativity, not only because these are the themes we have been studying, but also because they appear to us to be the most natural and logical extensions of the current state in knowledge-based CAD in the near future.

Our goal here is to show how the 4Cs can be implemented in a pilot system for which biologically inspired design, a design paradigm we describe in the following section, presents an excellent context. Biologically inspired design, by definition, is collaborative: it entails collaboration between biology and a design discipline such as engineering. Biologically inspired design is conceptual: biological analogies typically are most useful in the design ideation phase of design. Biologically inspired design is usually characterized as creative: it engages cross-domain analogies from biology to a design domain such as engineering. And with many years of *in situ* observations and design cases from courses taught by Georgia Tech's Center for Biologically Inspired Design (www.cbid.gatech.edu), we have a strong cognitive foundation for biologically inspired design. In the remainder of this paper, then, we describe a case study in the design, development, and deployment of a knowledge-based CAD system for supporting biologically inspired design. This case study represents both a prototype CAD system and a problem domain that we feel embody our vision of next generation design. As we will see below, the deployment of the prototype interactive CAD tool called DANE (for Design by Analogy to Nature Engine) yielded new insights into the problems of supporting the four C's.

2. Cognitive studies of biologically inspired design

Biologically inspired design, also sometimes referred to as biomimicry or bionics, is a design paradigm that uses analogies to biological systems to suggest creative design ideas for difficult engineering problems [109,110]. The paradigm attempts to leverage the billions of design solutions already existing in nature by exposing engineers to the biological world. Examples of biologically inspired designs range from bio-inspired clothing to biomimetic robots [111,112]. One specific example developed by scientists affiliated with Georgia Institute of Technology's Center for Biologically Inspired Design is a new material for iridescent computer screen surfaces based on the optical properties of nano-scale structures of morpho butterfly wings [113]. Another similar example (though in the domain of computing, not engineering) is a scheme for dynamic server allocation for the Internet by analogy to the foraging behavior of honeybees [114]. Since biological organisms often are robust, efficient, and multifunctional, the paradigm is rapidly gaining popularity with designers who need to produce innovative, environmentally conscious, and sustainable designs. From our perspective, this makes biologically inspired design a design paradigm of significant technological, economic and societal value.

Despite its many successes (e.g., [115]), the *practice* of biologically inspired design remains largely *ad hoc*, with little systematization of either the biological knowledge or the design process. Thus, a major challenge in research on biologically inspired design is how to transform a promising paradigm into a principled methodology.

Note that the examples of biologically inspired design presented here provide strong evidence that biologically inspired design is an instance of the 4C generation of design. Practitioners clearly are collaborating across the disciplines of biology and engineering; they are performing conceptual design; and they are generating creative design ideas.

As mentioned above, in order to develop CAD systems that blend into the workflow of designers, the tools should be grounded in cognitive studies of their design practices. In particular, development of a biologically inspired design as a design methodology requires understanding and organizing biological knowledge from an engineering perspective as well as understanding the content and processes of analogical retrieval and transfer of biological knowledge to address design problems in engineering. To these ends, we have conducted *in situ* studies of biologically inspired design in practice [116]. We have also participated in extended biologically inspired design projects [117]. Our studies add to the small but growing body of literature on cognitive studies of biologically inspired design. Mak and Shu [118], for example, report that designers engaged in biologically inspired design have design fixation problems as well as difficulties with analogical mapping during idea generation from biological phenomena. Both of these findings are similar to our own findings [116]. Mak and Shu also found that functional descriptions of biological systems in the form of flow of substances among components improve the quantity and quality of the generated design ideas. Similarly, Linsey et al. [119] found that learning about analogous products with general linguistic terms that apply across the problem and target domains improves an engineer's ability to use the analogous product. They also found that functional annotations on diagrams increase the chances of successful biological analogies.

While earlier cognitive studies of biologically inspired design typically have been done in controlled experiments in laboratory settings with small numbers of design students as human subjects, we have conducted our cognitive studies *in situ* through observation of an undergraduate, senior-level interdisciplinary class on biologically inspired design as well as design teams working on extended projects as part of the class. This course, ME/ISyE/MSE/PTFe/BIOL 4740, is taught by Georgia Institute of Technology's Center for Biology Inspired Design. We conducted the *in situ* studies during Fall 2006, Fall 2007, and Fall 2008. We know from Dunbar [120,121] that in general the analogy making behavior of humans in naturalistic and laboratory settings is quite different: people make more, and more interesting, analogies in their natural environments.

We have observed and documented three frequently occurring problem solving and representational activities of designers that served to inform construction of our CAD technology, DANE. These are: problem-driven and solution-driven design, compound analogical reasoning, and multi-modal analogical reasoning. A brief summary of the main findings is given below.

First, we identified the existence of two high-level processes for biologically inspired design based on two different starting points—*problem-driven* and *solution-driven process* [116]. Kruger and Cross [122] similarly have observed problem-driven and solution-driven strategies in industrial design, though, the similarity in labels notwithstanding, our characterization of the two strategies is quite different from theirs. In the problem-driven approach we observed, designers identified a problem, which formed the starting point for subsequent problem solving. They usually formulated their problem in functional terms (e.g., stopping a bullet). In order to find biological sources for inspiration, designers "biologized" the given problem, i.e., they abstracted and reframed the function in more broadly applicable biological terms (e.g., what characteristics do organisms have that enable them to prevent, withstand and heal damage due to impact?). They used a number of strategies for finding biological sources relevant to the design problem at hand, ranging from searching on functions to searching on champion adapters (see [116]). They then researched the biological sources in greater detail. Important principles and mechanisms that are applicable to the target problem were extracted to a solution-neutral abstraction, and then applied to arrive at a trial design solution.

In the solution-driven approach, on the other hand, designers began with a biological source of interest. They understood (or researched) this source to a sufficient depth to support extraction of deep principles from the source. The next step was to find human problems to which the principle could be applied. Finally the designers applied the source principle to generate a design solution to the identified problem.

Second, we found that biologically inspired design often involved compound analogies in which a new design concept was generated by composing the results of multiple cross-domain analogies [117]. This process of compound analogical design relies on an opportunistic interaction between two processes: *problem decomposition* and *analogy*. Nersessian [123,124] has described similar processes of compound analogies in her study of scientific discoveries. In case-based design, case composition is a basic process of creative case-based design (e.g., [26,28,30,125,126]). In engineering design, problem decomposition and solution composition are fundamental design processes (e.g., [101]).

Third, we observed that designers consistently used a combination of textual descriptions, pictures, graphs, and mathematical representations throughout the design process. These representations span not only multiple modalities (textual, diagrammatic, and pictorial) but also multiple levels of abstraction (pictures and diagrams of specific structures or parts of a biological system, to graphs and mathematical equations representing more abstract processes). Further, the use of multi-modal representations extended across disciplinary and level-of-experience boundaries.

Our observations suggest that the mental representations that designers use are rich and multimodal in nature, and are organized at different levels of abstraction. The use of multimodal knowledge representations is common practice in interactive case-based design aids such as ARCHIE [18], AskJef [21], CADET [24], CADRE [25], CASCAD and CADSYN [26], IDIOM [30], InteractiveKritik [127], and FABEL [29].

In addition to the published findings listed above, our cognitive studies identified five issues in biologically inspired design for which it may be productive to explore applications of CAD:

(1) The notion of functions of a system plays a central role in engineering design. However, biologists typically interpret function in terms of evolutionary benefits. The issue of functional representation is central to both organization of biological knowledge as well as collaboration and communication among biologists and engineers.
(2) The number of known biological systems is very large (in the billions). The issues of scale, complexity, generality and knowledge engineering need to be addressed to provide versatile, real-world application support.
(3) Biologically inspired design is fundamentally analogical. Current theories for accessing, mapping, abstracting and transferring information from biological source cases to engineering target problem may provide a starting point for aiding designers. However, it is not clear if the existing theories are sufficient for what is a fundamentally very challenging cognitive task.
(4) The generation of a solution in biologically inspired design often requires finding and understanding relevant biological information available in an ever-growing corpus of biological research. Designers often are faced with either too much or too little information relevant to the problem they are facing. Again current theories of human-information interaction may provide a starting point for aiding designers, but it is not clear that they are sufficient.
(5) Problem definition in biologically inspired design can be both ill structured and ill understood. Experts in one field (e.g., engineering) are confronted with large knowledge gaps in disciplines outside their own (e.g., biology). Furthermore, conceptual changes evoked from such diverse domains often require restructuring of large portions of problem descriptions. Frameworks for coping with dramatic shifts in problem structure must be developed to address the cognitive dissonance designers face when shifting between such diverse domains and across such gaps in knowledge.

3. From cognitive studies to design of CAD tools

From the observations made in our *in situ* cognitive studies, we abstracted functional requirements for a knowledge-based CAD tool called DANE for supporting the process of biologically inspired design. We then designed features for the software tool iteratively to meet these functional requirements. For example, we used variations of Structure-Behavior-Function models [45–47] to represent knowledge of biological systems. As is always the case, there are many more functional requirements that we wish we could have supported; some being limited by available technology, others simply victims of prioritization and resource constraints. A list of all functional requirements implied by our observations is beyond the scope of this paper; however, Table A.1 maps findings from our cognitive studies to functional requirements to features of the current tool. Note that each software feature in DANE could be regarded as representing the hypothesis that the feature is an aid for designers in overcoming the cognitive challenge described in our *in situ* study.

4. Functional modeling of biological systems

To represent functional models of biological and technological systems in our prototype CAD software, DANE, we used the Structure-Behavior-Function (SBF) knowledge representation scheme [45–47]. Briefly, (1) the structure portion of an SBF model of a complex system specifies the "what" of the system, namely, the components of the system as well as the connections among them. (2) Behaviors specify the "how" of the complex system, namely, the causal processes or mechanisms occurring in the system. A behavior is typically comprised of multiple states and transitions among them. The transitions are annotated by causal explanations for them. (3) Functions specify understanding of the "why" of the system. Note that a function is a particular interpretation of a system and its processes, not something inherent to that system. Functions act as indices to organize knowledge of structural components and causal behaviors. (4) A component of a complex system can itself comprise a system and thus have its own SBF model. The behavior of a system specifies the composition of the functional abstractions of its subsystems into the system functions. Thus, SBF models organize knowledge of a complex system in a $F \rightarrow B \rightarrow F \rightarrow B \rightarrow \cdots \rightarrow F(S)$ abstraction hierarchy.

Other researchers have described similar functional models of complex systems, e.g., the Function-Behavior-Structure models of Gero et al. [40] and Gero and Kannengiesser [41], the Function-Behavior-State models of Umeda et al. [42,43], and Umeda and Tomiyama [44], and the Function-Behavior models of Kitamura et al. [57,58]. The origin of our SBF models lies in Chandrasekaran's Functional Representation scheme [38,39]. Although the various functional representation schemes differ in many features, they typically share some key characteristics, viz., explicit representation of function, use of functional representations to organize knowledge of causal behaviors and structural components, a hierarchical system-subsystem organization of knowledge, a view of causal behavior as an intermediate abstraction between structure and function, and domain-independent vocabularies for representing structure, behaviors and functions of complex systems. Erden et al. [128] provide a recent survey of functional models of complex systems and their use in design.

The number of SBF models of complex systems in DANE is growing. When we first introduced DANE into a class on biologically inspired design in early fall of 2009, it contained a library of SBF models of 7 biological systems, with approximately 15 associated sub-function models, and 5 engineering systems. Many partially completed models were also included, for a total of about 40 functional models. The biological systems in DANE that we modeled occur at several levels of scale from the sub-cellular to organ function to organism sub-system (e.g. locomotion) levels. While this shows that the SBF knowledge representation scheme can capture the functioning of biological systems across multiple levels of scale, abstraction, and across at least two domains (engineering and biology), the issue of knowledge engineering of SBF models remains problematic. Historically, a key challenge for knowledge based AI systems, and specifically knowledge-based CAD systems, is that building a knowledge base takes deep knowledge of the domain as well as familiarity with the knowledge representation language (e.g., [2,18,21,32,36]). In the construction of our 7 biological models, including the 15 sub-functions, we noted that each model took between 40 and 100 h to complete. The process of understanding the functioning of underlying system (e.g. the kidney), modeling it, discovering faults in the model or in the modeler's understanding, and iterating over this process consumed the majority of the time. However, we estimate that the task of entering a completed SBF model into DANE required something less than 25% of this overall time cost.

Systems in DANE are indexed by system-function pairs, allowing them to be accessed by *name* (e.g., "Lotus Leaf Cleans Self"), by *subject* (e.g., "Lotus Leaf"), and/or by *verb* (e.g., "Clean"). Upon selecting a system-function pair, users are presented with a multi-modal representation of the paired system-function (e.g. the Lotus Leaf SBF model). A system can be represented in text descriptions and images, as well as through visualizations of behavior and structure models. Example text and image modalities for the "Lotus Leaf Cleans Self" model can be seen in Fig. B.1. Behavior and structure models are themselves represented as directed graphs, which may be annotated with text descriptions and images. The nodes and edges represent either structural

Table A.1
Functional requirements derived from cognitive studies.

Observation	Functional requirement	Supporting feature
Problem-driven/Solution-based design processes	For problem-driven design: enable search for biological solutions using function matching.	Library of available solutions is indexed by function, or verb e.g. "kidney filters blood" or "filters"
		Library is organized around function and subfunctions for optimal query capability Provide descriptions of the mechanism of the biological system (called behavior) for ease of transfer to human engineering problem
	For solution-based design: describe solutions with sufficient functional abstraction for transfer to engineering problems	For a known biological solution, establish a rigorous method of describing a function, e.g. (subject, verb, object*, preposition*, adverb*) that minimizes biological terminology in order to facilitate indexing into engineering problems. Example: the term "transpiration" is "plant transports water passively from soil to leaves"
Compound Analogy	Provide a simple means of showing the decomposition of a solution, so that the decomposition can be transferred to the problem	A graphical representation of the function decomposition of the biological solution is provided in the lower left corner of the tool. The decomposition is automatically generated from the SBF model of the selected system-function. It serves a dual role as a functionally oriented solution navigation tool
Multi-modal Representations	Provide text, pictorial, and model representations of biological systems	For each first class data object (e.g. function, state, object, behavior, structure) we provide a consistent interface that enables text and pictorial representation on one tab, and model-based representations on another. (This is not available for some secondary data objects, such as transition, object property, and principle.)
Centrality of Function	Emphasize the primacy of function for each system	As mentioned above, function is a primary indexing and searching mechanism, a rigorously defined data structure, an organizing theme for the database, and a navigation vehicle
Scale of Biological Sources	Enable biological systems to be represented at multiple levels of physical scale	The SBF representations have been shown capable of representing molecular, cellular, organ and human-scale systems effectively
	Enable arbitrarily complex biological organisms to be represented	System – sub-system relationships are hierarchically organized through function – sub-function relationships to any arbitrary depth, such that any functionally decomposable system can be represented to an arbitrary level of complexity
	Enable a wide range of access to the technology to "crowd-source" the massive numbers of biological systems that could be added to the system	DANE uses an internet enabled client–server application that enables a geographically varied access, in a java-based architecture that is independent of client side hardware. Participants are currently able to access and edit all content, stored in a centrally located database.
	Enable the fundamental processes of "identification", "mapping" and "transfer" inherent in analogical processing	Identification of applicable solutions as analogy candidates is enabled primary through functional indexing and search
		Mapping is supported through (a) the structured functional vocabulary, (b) maximizing terminology reuse through a common library of objects and terms, SBF models that explicitly describe how a system accomplishes a function; also applications for transfer of working principles from one design solution to another
Information Foraging	Enable the designer to quickly access research papers and text about the biological system	Text descriptions of biological systems in DANE may be annotated with footnotes to references
Problem Definition	Enable the design to track the development of their design problem as it unfolds over time	While we note that problem understanding and description is major gap in student capabilities, no support for problem description is provided in the current version of DANE

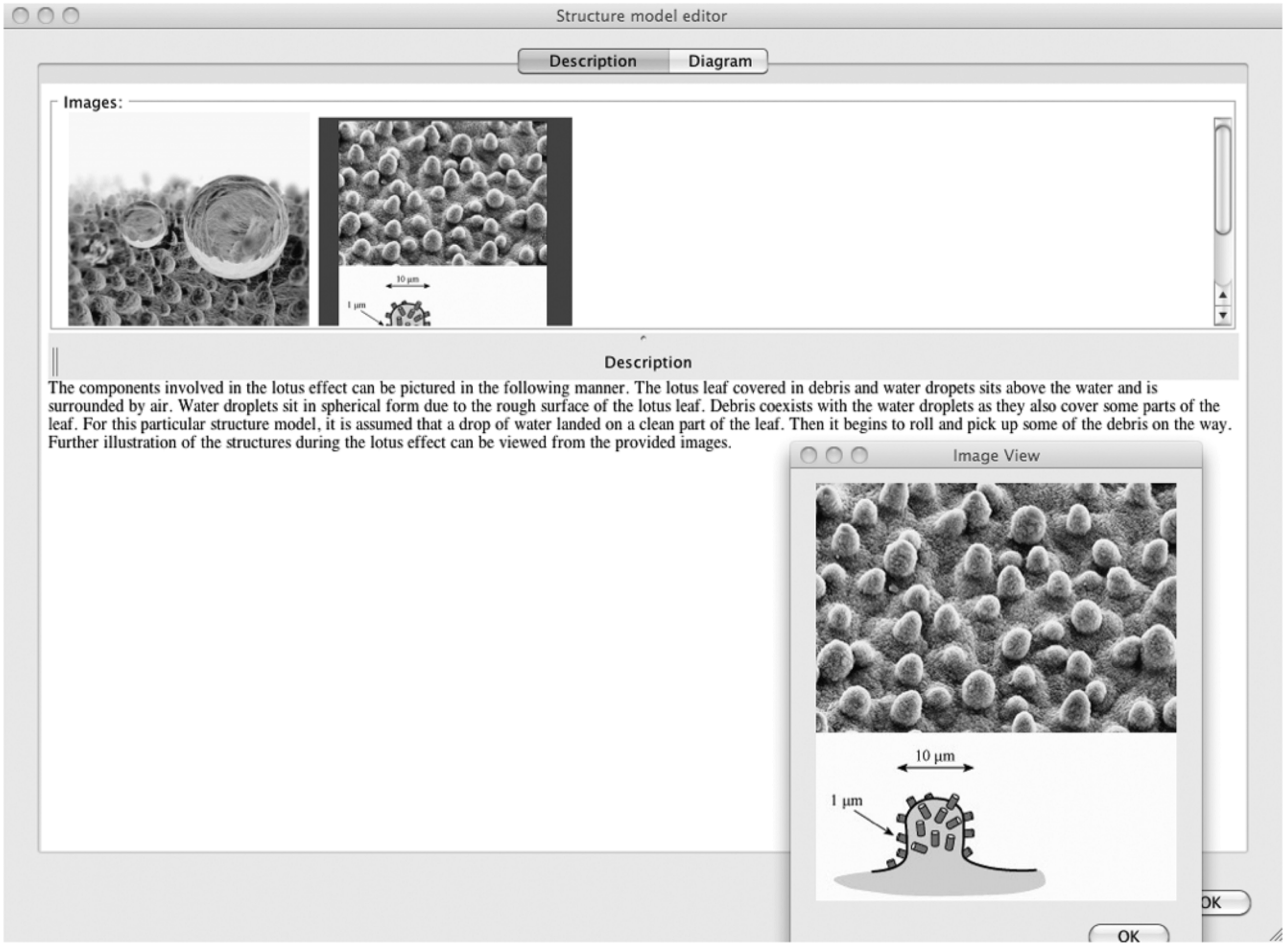
elements and connections (for structure models) or states and transitions (for behavior models), respectively. An example behavior model for the system "Lotus Leaf Cleans Self" (the same model as in Fig. B.1) can be seen in Fig. B.2. Additionally, each system is visually connected to other systems with which it shares a sub or super-function relationship. This functional hierarchy is represented as an interactive graph (see Fig. B.3) with nodes representing systems and edges representing the sub/super relationships. Users may navigate between systems by double-clicking on a node. Note that we go into greater detail about DANE in section five.

4.1. Illustrative examples of SBF models of biological systems

In this section we describe two biological systems, the lotus leaf and the basilisk lizard. These examples are indicative of the kind and complexity of models within DANE. There are two important things to note regarding these examples. First, while we present only biological examples, DANE is also fully capable of modeling engineering systems. Second, the models presented here look visually different than those within DANE (e.g., Figs. B.1–B.3). This is intentional. The SBF modeling language is not tied to a specific visualization scheme, and although we have chosen a particular visual syntax with which to show models in DANE, these examples illustrate other ways in which one could depict an SBF model.

4.1.1. Lotus leaf

For our first example, we will model the lotus leaf (see Fig. B.4). Note that this model has slightly different information from what is presented in Fig. B.2 although both represent the same function. In particular, Fig. B.2 displays the behavior model as it exists in DANE at present, while Fig. B.4 presents a more complete version in a slightly different visual style. We note that current model in



The components involved in the lotus effect can be pictured in the following manner. The lotus leaf covered in debris and water dropets sits above the water and is surrounded by air. Water droplets sit in spherical form due to the rough surface of the lotus leaf. Debris coexists with the water droplets as they also cover some parts of the leaf. For this particular structure model, it is assumed that a drop of water landed on a clean part of the leaf. Then it begins to roll and pick up some of the debris on the way. Further illustration of the structures during the lotus effect can be viewed from the provided images.

Fig. B.1. *An example of a multi-modal representation in DANE.* The user is investigating the structure of the Lotus Leaf in the "Lotus Leaf Cleans Self" model. The system presents images (which may be expanded as in the bottom-right), a textual description, and a diagram, viewable by selecting the appropriate tab.

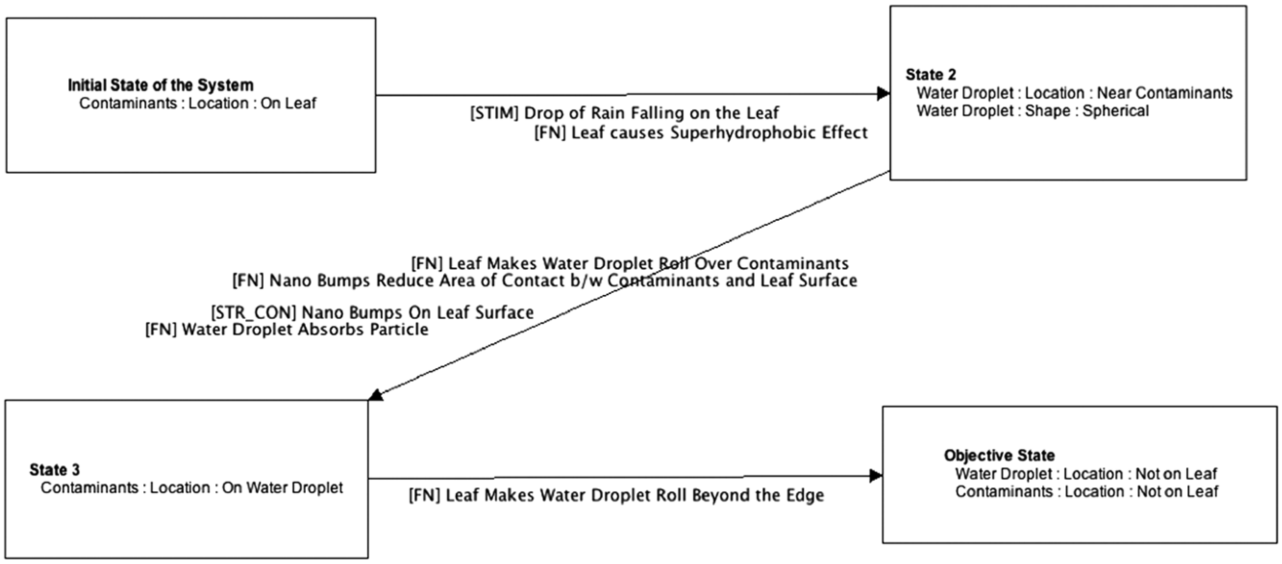
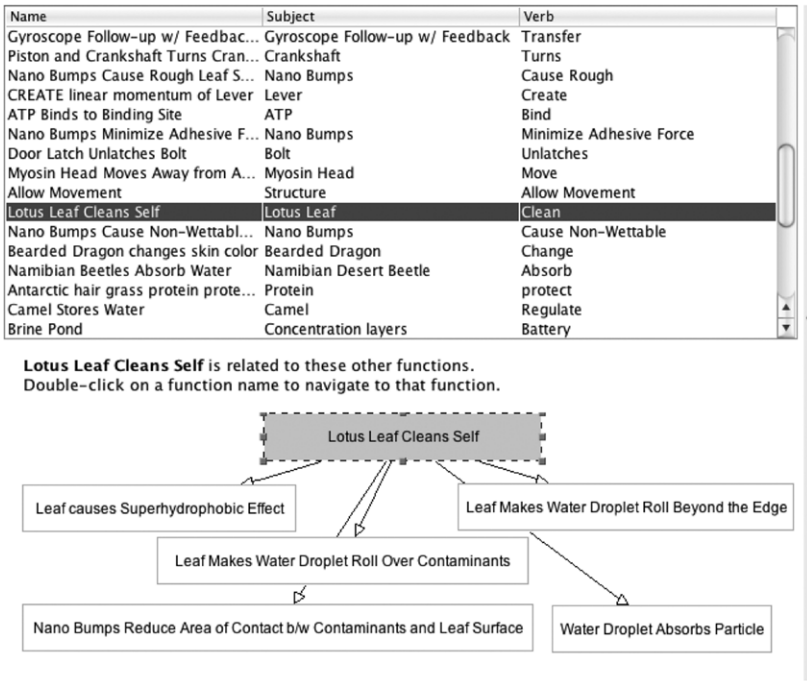

Initial State of the System
Contaminants : Location : On Leaf
[STIM] Drop of Rain Falling on the Leaf
[FN] Leaf causes Superhydrophobic Effect
State 2
Water Droplet : Location : Near Contaminants
Water Droplet : Shape : Spherical
[FN] Leaf Makes Water Droplet Roll Over Contaminants
[FN] Nano Bumps Reduce Area of Contact b/w Contaminants and Leaf Surface
[STR_CON] Nano Bumps On Leaf Surface
[FN] Water Droplet Absorbs Particle
State 3
Contaminants : Location : On Water Droplet
[FN] Leaf Makes Water Droplet Roll Beyond the Edge
Objective State
Water Droplet : Location : Not on Leaf
Contaminants : Location : Not on Leaf

Fig. B.2. *DANE behavior model of "Lotus Leaf Cleans Self".* In this diagram, each node represents a state and each edge represents a transition from one state to the next. Annotations on the edges describe why the system performs that particular transition.

DANE (Fig. B.2) could be easily extended to become like the more complete model (Fig. B.4).

The lotus leaf is a system with properties that allow it to clean itself of debris. Each of the dashed boxes in the figure represents a particular function and its corresponding behavior. The largest box, on the left-hand side, represents the primary function of the Lotus Leaf.

We will begin by examining the primary function. In the initial state, placed at the top of the diagram, there are contaminants on the leaf. Next, because a drop of rain falls on the leaf and because

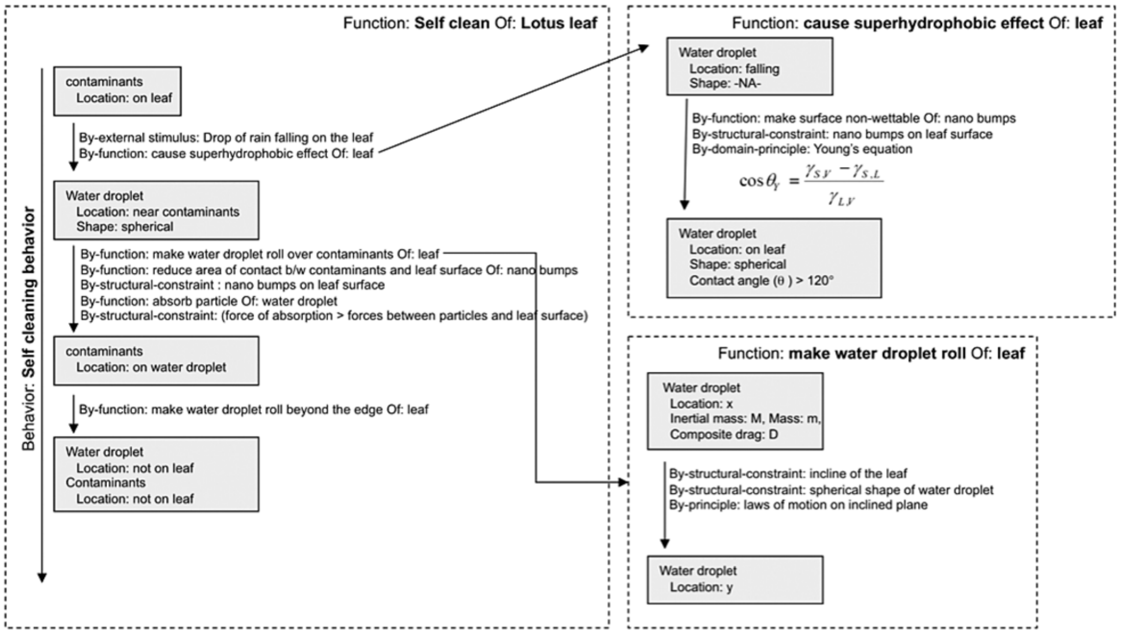


Name	Subject	Verb
Gyroscope Follow-up w/ Feedbac...	Gyroscope Follow-up w/ Feedback	Transfer
Piston and Crankshaft Turns Cran...	Crankshaft	Turns
Nano Bumps Cause Rough Leaf S...	Nano Bumps	Cause Rough
CREATE linear momentum of Lever	Lever	Create
ATP Binds to Binding Site	ATP	Bind
Nano Bumps Minimize Adhesive F...	Nano Bumps	Minimize Adhesive Force
Door Latch Unlatches Bolt	Bolt	Unlatches
Myosin Head Moves Away from A...	Myosin Head	Move
Allow Movement	Structure	Allow Movement
Lotus Leaf Cleans Self	Lotus Leaf	Clean
Nano Bumps Cause Non-Wettabl...	Nano Bumps	Cause Non-Wettable
Bearded Dragon changes skin color	Bearded Dragon	Change
Namibian Beetles Absorb Water	Namibian Desert Beetle	Absorb
Antarctic hair grass protein prote...	Protein	protect
Camel Stores Water	Camel	Regulate
Brine Pond	Concentration layers	Battery

Lotus Leaf Cleans Self is related to these other functions.
Double-click on a function name to navigate to that function.

Lotus Leaf Cleans Self
Leaf causes Superhydrophobic Effect
Leaf Makes Water Droplet Roll Beyond the Edge
Leaf Makes Water Droplet Roll Over Contaminants
Nano Bumps Reduce Area of Contact b/w Contaminants and Leaf Surface
Water Droplet Absorbs Particle

Fig. B.3. *Function browser (top half) and a functional hierarchy for "Lotus Leaf Cleans Self" in DANE (bottom half).* Functions are sortable by their name, their subject, or a basic verb describing the function. The function hierarchy highlights the currently viewed function and places super-functions above and sub-functions below.



Function: Self clean Of: Lotus leaf
Behavior: Self cleaning behavior
contaminants Location: on leaf
By-external stimulus: Drop of rain falling on the leaf
By-function: cause superhydrophobic effect Of: leaf
Water droplet Location: near contaminants Shape: spherical
By-function: make water droplet roll over contaminants Of: leaf
By-function: reduce area of contact b/w contaminants and leaf surface Of: nano bumps
By-structural-constraint : nano bumps on leaf surface
By-function: absorb particle Of: water droplet
By-structural-constraint: (force of absorption > forces between particles and leaf surface)
contaminants Location: on water droplet
By-function: make water droplet roll beyond the edge Of: leaf
Water droplet Location: not on leaf Contaminants Location: not on leaf
Function: cause superhydrophobic effect Of: leaf
Water droplet Location: falling Shape: -NA-
By-function: make surface non-wettable Of: nano bumps
By-structural-constraint: nano bumps on leaf surface
By-domain-principle: Young's equation
$\cos\theta_Y = \frac{\gamma_{SV} - \gamma_{SL}}{\gamma_{LV}}$
Water droplet Location: on leaf Shape: spherical Contact angle (θ) > 120°
Function: make water droplet roll Of: leaf
Water droplet Location: x Inertial mass: M, Mass: m, Composite drag: D
By-structural-constraint: incline of the leaf
By-structural-constraint: spherical shape of water droplet
By-principle: laws of motion on inclined plane
Water droplet Location: y

Fig. B.4. *Function/behavior model of the Lotus Leaf.* As in Fig. B.2, nodes represent states, edges represent transitions, and annotations on edges describe why the transition occurred. Arrows pointing to another function are used to describe the sub-function relationship between one behavior and a separate function.

of the leaf's surface causes the superhydrophobic effect, the water droplet forms a spherical shape. Next, the water droplet rolls over the contaminants on the leaf and absorbs them because the nano-bumps on the surface of the leaf reduce the area of contact between it and the contaminants, which causes the force of absorption from the water to be greater than the forces binding the contaminants to the leaf's surface. The system transitions to the final state where the water droplet and the contaminants (now contained within the water droplet) are no longer on the leaf because the water droplet continues rolling until it falls off the edge of the leaf.

Two sub-functions of the Lotus Leaf's primary function are also represented in this example. The first sub-function describes how the leaf causes the superhydrophobic effect. This function only contains two states: the initial state, where the water droplet is falling towards the leaf, and the final or objective state, where the water droplet is contacting the leaf at an angle greater than 120° and its shape is spherical. The system achieves this objective state because the nano-bumps on the surface of the leaf make its surface non-wettable and cause a high contact angle between the droplet and the leaf's surface.

The second sub-function describes how the water droplet rolls off the leaf. Like before, this sub-function only has initial and objective states. The initial state describes the water droplet existing in some location *X* on the leaf with an associated inertial

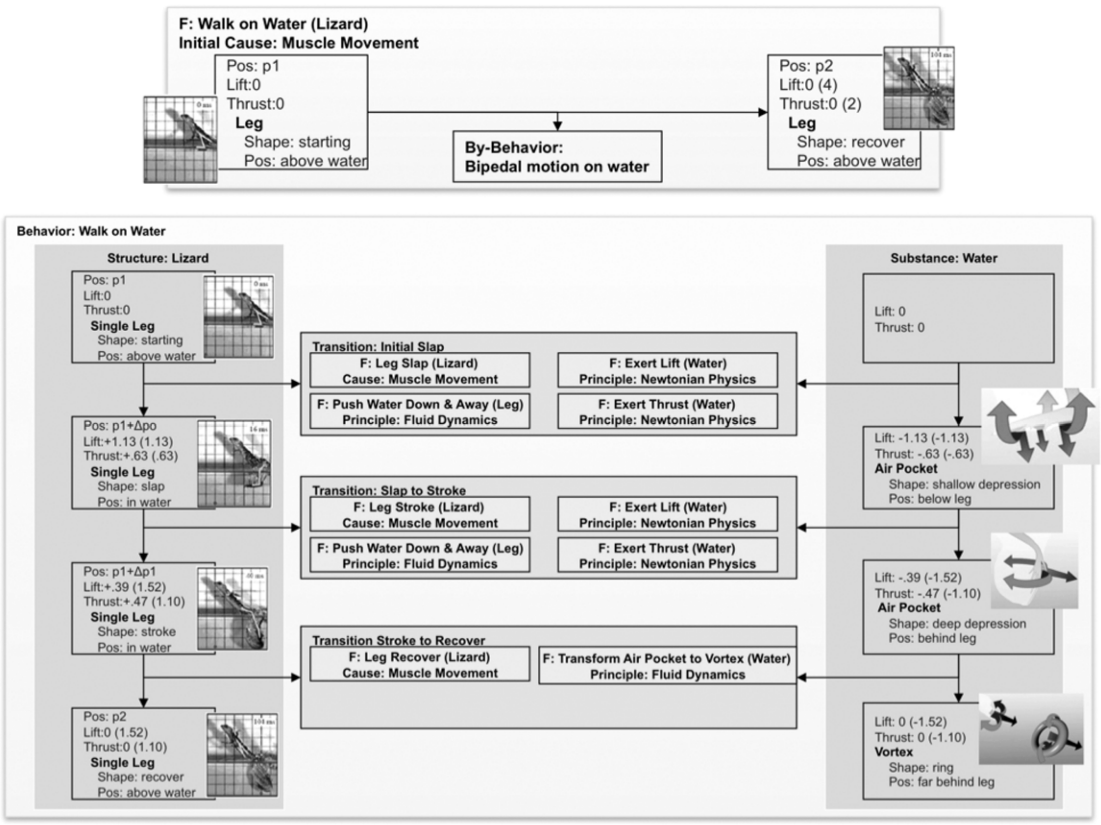


Fig. B.5. *Function/behavior model of the Basilisk Lizard*. Nodes with pictures (and the top-right node in the Walk on Water function) represent states, and edges represent transitions between states. Nodes pointed to in-between states are transition annotations.

mass and composite drag. The objective state shows the water droplet now at some other location Y. This change is state is achieved because the lotus leaf sits at an inclined plane, the water droplet is spherical, and the laws of motion of a spherical object on an incline plane dictate that the spherical object will roll down the plane.

4.1.2. Basilisk lizard

Next we will discuss the Basilisk Lizard (see Fig. B.5). This lizard has the unique function of walking on water and has even earned the nickname of the "Jesus lizard" for this feat. Note that because the walking itself is a repetition of the same set of leg movements, we have only modeled a single complete motion of one of the lizard's legs in this example as opposed to the complete set of movements required to walk across a body of water.

In this particular visualization of an SBF model, we show the function as an initial state and objective state, and we say that the reason the system is able to transition between these states is due to its behavior, which is shown below the function. Although in previous visualizations we did not explicitly state that the behavior provided an explanation for a function, this connection is implicit in our representation schema.

The behavior diagram is subdivided into two vertical columns. The left column shows the behavior of the lizard (specifically, the lizard's leg), and the right column shows the behavior of the water with which the lizard is interacting. In prior examples, our visualization would combine the states in these two columns such that one state would show both the lizard's leg and the water. However, here we have decided to split them because it allows the reader to separately consider and focus on the behavior of either component.

In the beginning state, the lizard's leg is above the water and in an initial configuration or shape, and the water initially exerts no lift or thrust on the lizard. The leg then slaps the water, causing water to be pushed away. Newtonian physics tells us that every action has an equal and opposition reaction, so we transition to a state where there is now positive lift and thrust exerted on the lizard by the water. Additionally, an air pocket is created in the water below the leg. Next, the lizard's leg, now underwater, uses its muscles to stroke backwards, which pushes the water and air pocket down and away from itself. Again, due to the laws of physics, more force is exerted against the lizard, causing even more positive lift and thrust to be applied to it. Finally, the lizard recovers its leg from the water, which transforms the air pocket into a water vortex, allowing the lizard to maintain its positive thrust and lift.

5. Design and development of DANE

In this section we describe our prototype CAD technology. Not only is DANE designed to address the issues our cognitive studies identified in biologically inspired design, but it is also represents our first attempt at building a next generation CAD system. As the name implies, the Design by Analogy to Nature Engine (DANE) is intended in the long term to be a semi-automated engine for design. However, at present, DANE interactively facilitates biologically inspired design by (1) helping designers find biological systems that might be relevant to a given engineering design problem, (2) aiding designers in understanding the functioning of biological systems so that they can extract and transfer the appropriate principles to engineering design problems, and (3) enabling designers and design teams to construct and enter functional models of biological and engineering systems into DANE. In addition to describing the tool itself, we report later in this work on the deployment of DANE in an undergraduate multi-disciplinary course on design.

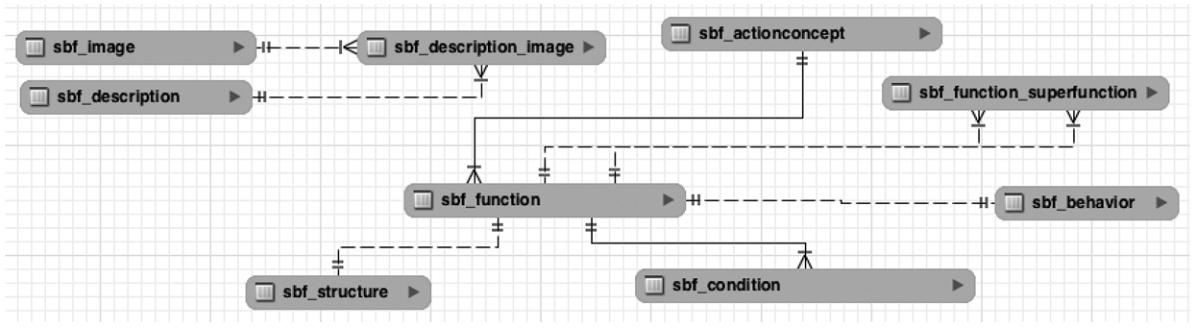


Fig. B.6. *Top-level ER diagram of DANE's database schema.* This diagram represents a high-level look at the entities and relationships in the schema of the server-side database for DANE.

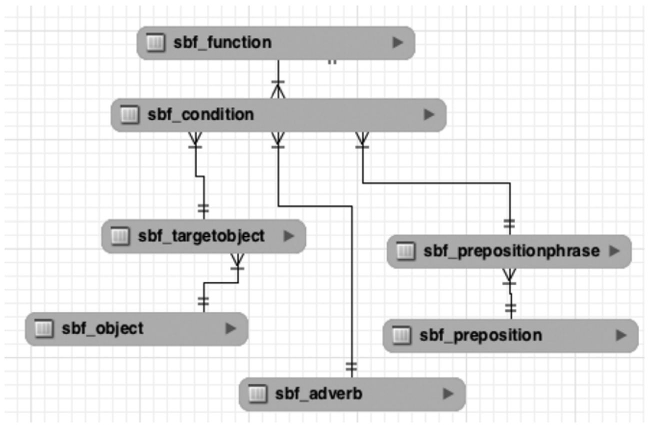


Fig. B.7. *ER diagram for condition and related tables in DANE's database schema.* This diagram expands on the condition table's relationships to other tables in the database. Functions are partially defined through a set of conditions.

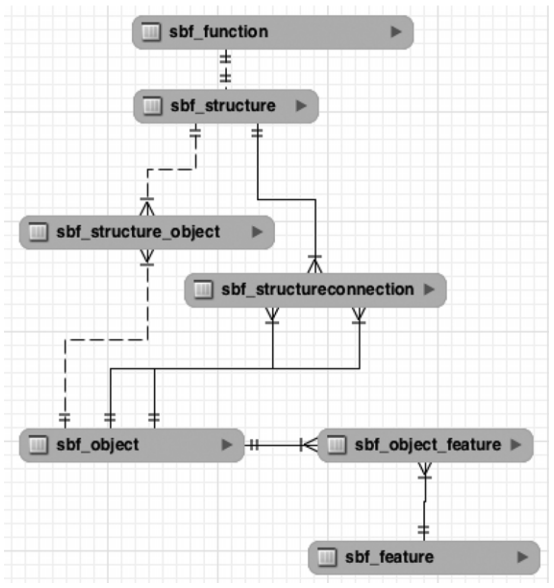


Fig. B.8. *ER diagram for structure and related tables in DANE's database schema.* This diagram expands on the structure table and its related sub-tables. combined, these tables form the data backend for structure models in DANE.

5.1. DANE's architecture

DANE is a Java Web Start application that employs a client–server architecture. Our server is a standard desktop PC configuration that utilizes off-the-shelf hardware. The server is running Red Hat Linux, version 2 of the Glassfish application server, a MySQL database. Glassfish and MySQL work in concert to provide server-side support for our application. The server is also running Apache, which services only our project's website. The website is a simple static page that provides a description of the project and a link to launch the client application.

In our architecture, the server acts as an intermediary between the client and the database. All transactions are done through the framework provided by Enterprise Java Beans 3 technology with the Glassfish application server handling all communication details. Through this architecture, each database table is mapped to a Java object, and relationships are expressed as objects that point to other objects. For example, in a one-to-many situation such as one Structure diagram containing many Structural Components ("Objects" in our database parlance), the EJB technology would create a Structure Java object that contains an array of pointers to all the Structural Component objects of which it is associated. Essentially, this architecture enables the client to treat all server-side database content as Java objects. It also provides the software developer with a straightforward API for requesting objects from the server and committing any changes made to those objects. The interested reader is encouraged to peruse the copious EJB3 documentation that exists on the Internet.

5.2. DANE's knowledge base

Our database is organized in terms of the SBF modeling framework. In Fig. B.6, you can see a top-level entity-relationship diagram. Note that in this and the following ER diagrams, there is no functional difference between a solid and a dashed line. This diagram shows how a Function is related to a single Structure, a single Behavior, a single verb (an "action concept"), and many conditions. This diagram also shows the tables we utilize to annotate several of our objects (Structures, Behaviors, Functions, Structural Components, and Behavior States) with textual descriptions and images. For the sake of clarity, these connections are not depicted on the diagram.

A Condition (Fig. B.7) describes the context of a Function. A Function may have many Conditions associated with it. Each Condition may be one of several types: (a) a Target Object, which describes what the Function affects; (b) an Adverb, which describes how the target object expresses its action; or (c) a Preposition Phrase, which positions the function relative to some external environment. For example, the Function might be the following: Basilisk Lizard propels self swiftly across water. Self is a target object; swiftly is an adverb; across water is a preposition phrase.

A Structure is itself a complex series of tables, as can be seen in Fig. B.8. Again, a Structure is related to a single Function. A Structure contains many Objects ("Structural Components" in SBF terminology). An Object may belong to more than one Structure (e.g., the Structures of both the Lotus Leaf and Transpiration models contain the "Leaf" Structural Component). An Object is related to many Object-Feature-Value triples. These triples relate an Object (such as a "Leaf") to a Feature ("Color") and a value ("Green").

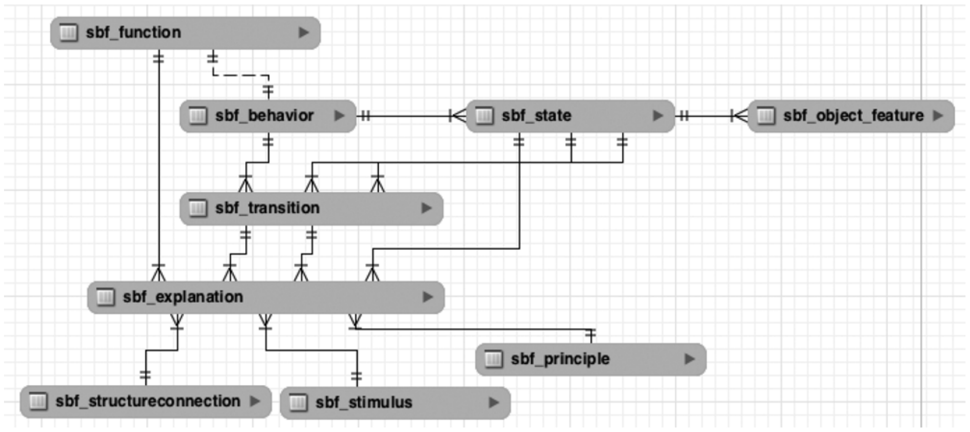


Fig. B.9. *ER diagram for behavior and related tables in the DANE database schema*. This diagram expands on the behavior table and its related sub-tables. combined, these tables form the data backend for the behavior models in DANE.

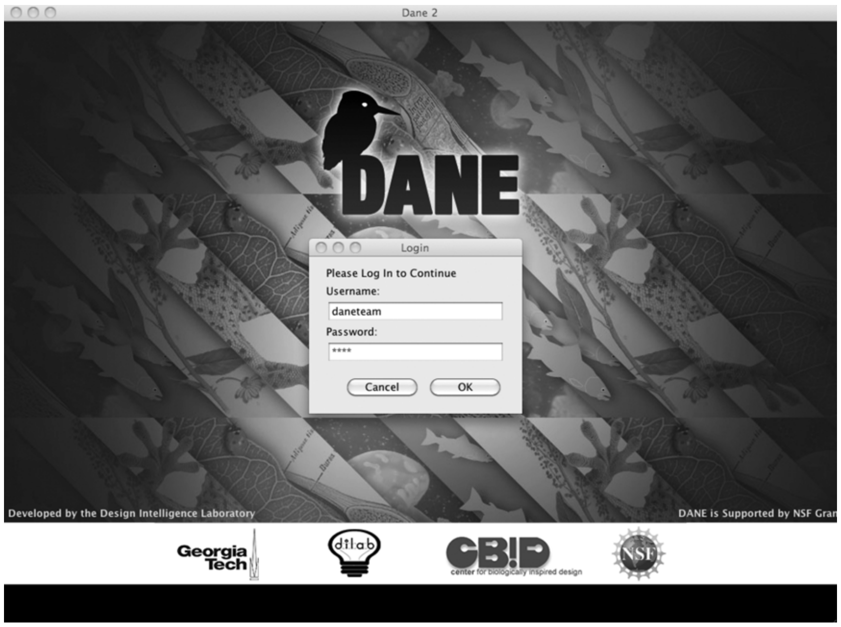

Dane 2
DANE
Login
Please Log In to Continue
Username:
daneteam
Password:
****
Cancel
OK
Developed by the Design Intelligence Laboratory
DANE is Supported by NSF Gran
Georgia Tech
dilab
CB!D
center for biologically inspired design
NSF

Fig. B.10. *The DANE log-in screen*. This is the first screen a user sees when booting DANE. He or she is prompted to enter his or her credentials before continuing to the main screen (Fig. B.11).

Object-Feature-Value records are used to describe the value of particular Object Features in a Behavior State, but in the case where the value is kept blank, an Object-Feature-Value record may simply be created to enumerate the relevant features of a particular Object. A Structure also contains several Structure Connections, which create a relationship between two Structural Components.

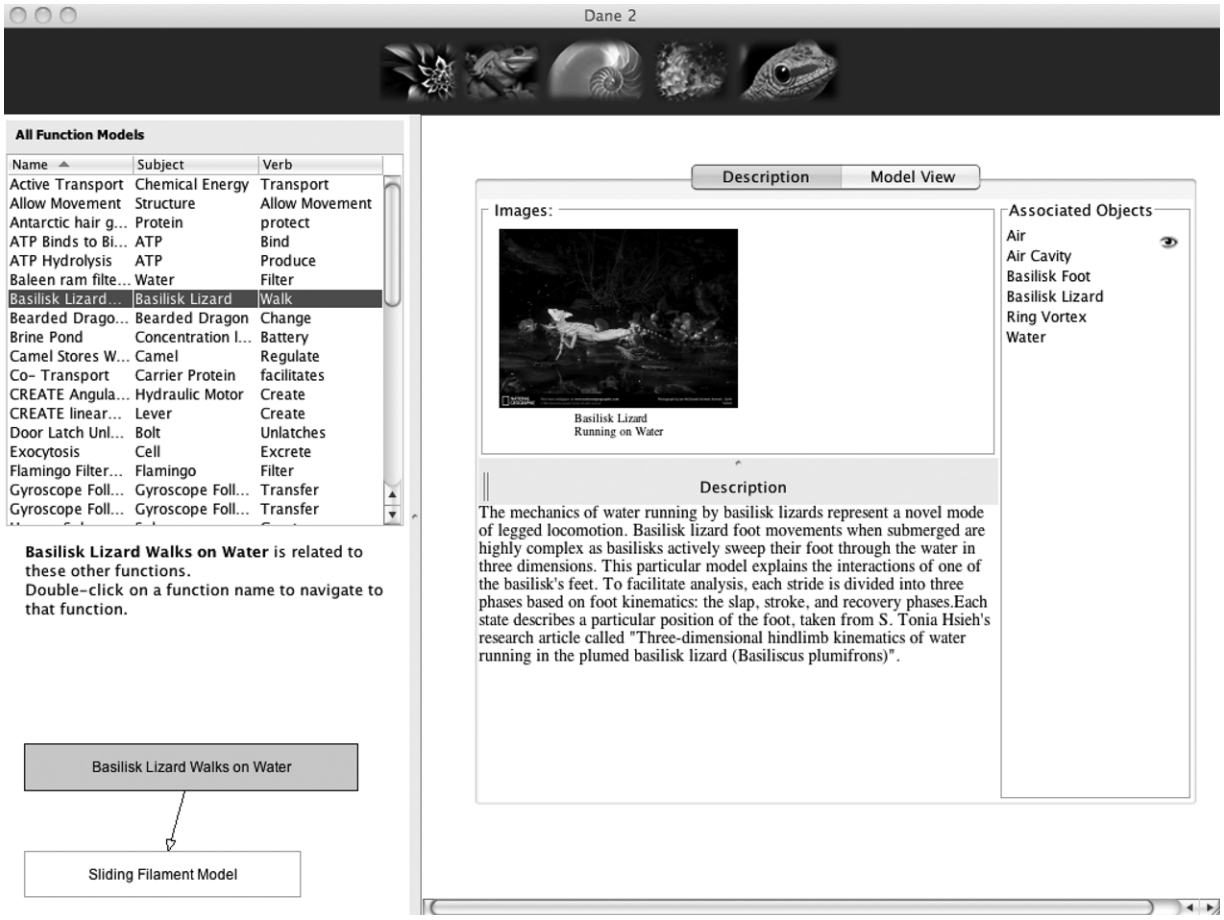
A Behavior is also a series of tables, which can be viewed in Fig. B.9. Like a Structure, a Behavior is related to a single Function. The Behavior contains many States, which themselves contain many Object-Feature-Value triples (described in the previous paragraph). A Behavior also contains several Transitions. A Transition creates a relationship between two States and also has several Explanations associated with it. An Explanation may be one of several types: "By function" points to a Function, "by Structural Connection" points to a Structural Connection, etc.

As previously mentioned, the client-side application is a Java program that launches via the Java Web Start technology. This technology enables users to click on a link at our project web site that will automatically download and launch the application. If a user had previously downloaded the application, Java Web Start will also automatically detect and update the application if newer code is available on the server. In the following paragraphs, we will provide the reader with a brief run-through of all DANE's user-interface screens. Note that all graphs are drawn using the JGraph graphics library, and all other forms were created using standard Java GUI components.

5.3. User interaction in DANE

When a user launches DANE, he or she will first log in by entering his or her username and password, shown in Fig. B.10. Note that henceforth we will be viewing DANE from the perspective of a user with read-only access. A user with edit privileges would simply see more interactive features that provide simple dialogs and lists for the creation and editing of content within models and our database. The actual display of the models is exactly the same in both read-only and editor modes.

Once the user has logged in, he or she will see Fig. B.11, although initially the right- and bottom-left panels will be empty. This screen is composed of three parts. The left-hand panel displays a list of models in the database, which can be sorted by the function of the system being described, the subject of the function, or the verb that characterizes the function. Once the user has clicked on



All Function Models

Name	Subject	Verb
Active Transport	Chemical Energy	Transport
Allow Movement	Structure	Allow Movement
Antarctic hair g...	Protein	protect
ATP Binds to Bi...	ATP	Bind
ATP Hydrolysis	ATP	Produce
Baleen ram filte...	Water	Filter
Basilisk Lizard...	Basilisk Lizard	Walk
Bearded Drago...	Bearded Dragon	Change
Brine Pond	Concentration I...	Battery
Camel Stores W...	Camel	Regulate
Co- Transport	Carrier Protein	facilitates
CREATE Angula...	Hydraulic Motor	Create
CREATE linear...	Lever	Create
Door Latch Unl...	Bolt	Unlatches
Exocytosis	Cell	Excrete
Flamingo Filter...	Flamingo	Filter
Gyroscope Foll...	Gyroscope Foll...	Transfer
Gyroscope Foll...	Gyroscope Foll...	Transfer

Basilisk Lizard Walks on Water is related to these other functions.
Double-click on a function name to navigate to that function.

Basilisk Lizard Walks on Water → Sliding Filament Model

Associated Objects
Air
Air Cavity
Basilisk Foot
Basilisk Lizard
Ring Vortex
Water

Description

The mechanics of water running by basilisk lizards represent a novel mode of legged locomotion. Basilisk lizard foot movements when submerged are highly complex as basilisks actively sweep their foot through the water in three dimensions. This particular model explains the interactions of one of the basilisk's feet. To facilitate analysis, each stride is divided into three phases based on foot kinematics: the slap, stroke, and recovery phases.Each state describes a particular position of the foot, taken from S. Tonia Hsieh's research article called "Three-dimensional hindlimb kinematics of water running in the plumed basilisk lizard (Basiliscus plumifrons)".

Fig. B.11. *The main DANE screen*. On the left side is a sortable list of all models in DANE. Once a model is selected, such as "Basilisk Lizard Walks on Water" in this figure, the bottom left panel shows how the function relates to other functions in DANE, and the right panel displays information on that model in three different modalities: text, image, and a structured view (shown in Fig. B.12).

a model (the state shown in the figure), two areas of the screen fill with content: the bottom-left panel and the right panel. The bottom-left panel shows a function/sub-function hierarchy. This hierarchy is built by linking to any model that is referenced in a By Function (displayed as "[FN]" in DANE) transition in the Behavior model of the currently selected system. Any box in this hierarchy can be double-clicked on to jump to the model to which it refers. The right panel of the screen displays the details of the model. In this figure we see a description and several images for the model of Basilisk Lizard Walks on Water. We also see a list of "Associated Objects". This list displays objects that appear in the Structure model, the Function model, or the Behavior model. One can be inspect an object by double-clicking on it or by selecting an object and clicking on the eyeball icon. The screen that appears is the same that shows when clicking on "Explore Object", which is described below.

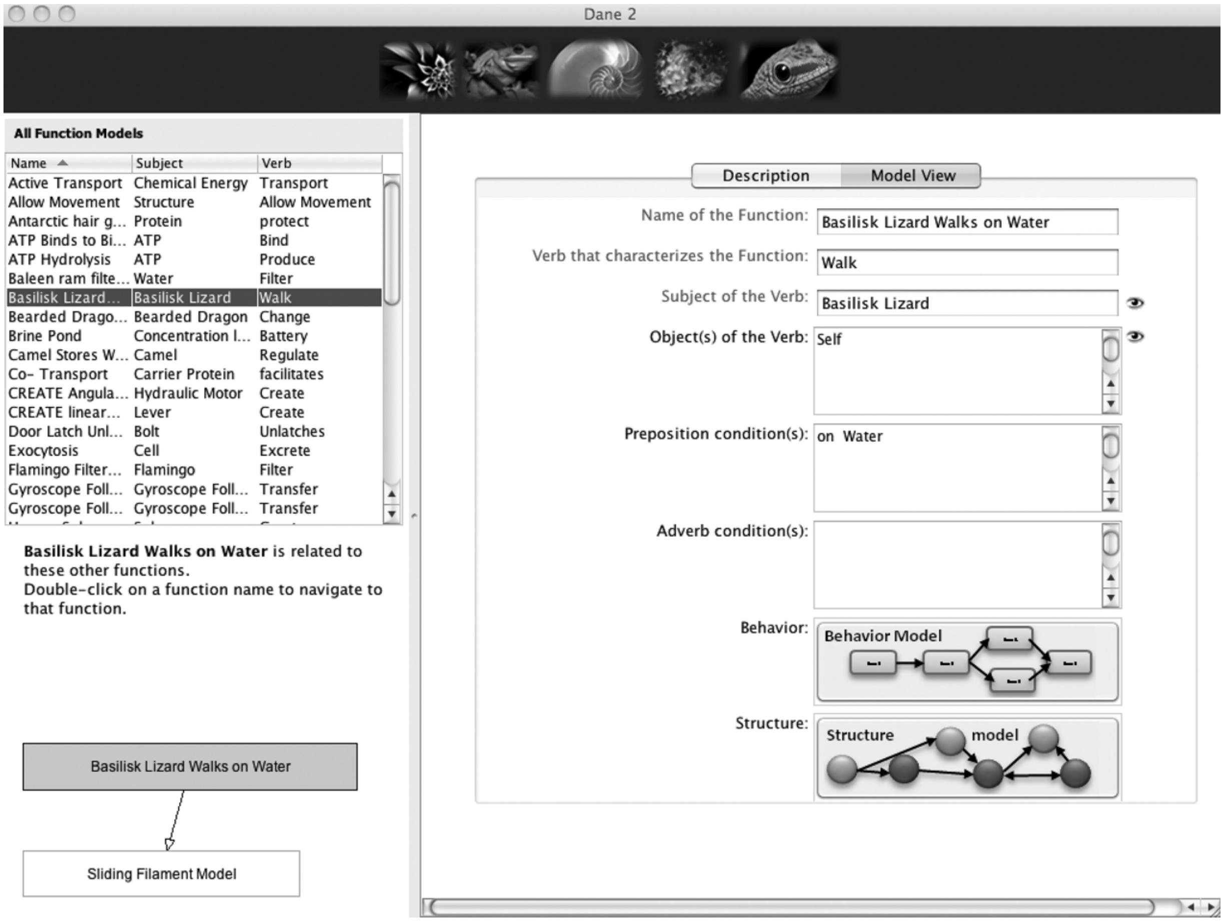
Clicking on the "Diagram" tab for the same system would transition to Fig. B.12. This screen explicitly explains the Function by breaking it up into its subject, verb, target object, and condition aspects. Additionally, the user is presented with buttons to display the Behavior and Structure models associated with this Function.

Fig. B.13 illustrates what the user would see as a Behavior for this function. Note that this textual description and image visual layout is the same as what was shown in Fig. B.11, except that here we have double-clicked on an image and see an enlarged version of it in a new window. Whenever we speak of textual descriptions and images in DANE, they are all presented in this same visual layout. The only exception is the textual description and image layout that displays when first clicking on a model, for this incorporates the "Associated Objects" list.

If the user then clicked on the "Diagram" tab, he/she would see Fig. B.14. The user can interact with this graph, moving nodes around and reorganizing it spatially to meet his/her visual preferences. If the user clicks on a particular state in the Behavior and selects the "Explore State" button, the user would see Fig. B.15. This screen provides a textual description and images. The "Overview" tab simply reiterates the information that appeared in the diagram.

Next, the user might close the Behavior window and choose to inspect the Structure, which would appear as Fig. B.16. Note we have skipped the "Description", as it is the same visually (although has separate content from) the "Description" tab in Behavior. Similar to Behavior, the spatial layout of the Structure graph is interactive and can be modified by users if they desire. The user may choose to inspect a particular Structural Component by clicking on a node and pressing the "Explore Object" button, as shown in Fig. B.17. Again, we skip the "Description" tab because it is visually the same as the one displayed earlier. The "Overview" tab displays information about the domain of the object and any properties that have been associated with it or its parent objects (e.g., a parent of "Basilisk Foot" might be an abstract object called "Foot"). Note that at the time of writing, our object hierarchy feature was not fully functional, so no inherited properties are displayed in the figure.

After viewing this model, the user may choose to browse to another model. This may be done either through the model list shown in Fig. B.11 or by double-clicking on a related model through the functional hierarchy, also shown in Fig. B.11.


Dane 2

All Function Models

Name	Subject	Verb
Active Transport	Chemical Energy	Transport
Allow Movement	Structure	Allow Movement
Antarctic hair g...	Protein	protect
ATP Binds to Bi...	ATP	Bind
ATP Hydrolysis	ATP	Produce
Baleen ram filte...	Water	Filter
Basilisk Lizard...	Basilisk Lizard	Walk
Bearded Drago...	Bearded Dragon	Change
Brine Pond	Concentration I...	Battery
Camel Stores W...	Camel	Regulate
Co- Transport	Carrier Protein	facilitates
CREATE Angula...	Hydraulic Motor	Create
CREATE linear...	Lever	Create
Door Latch Unl...	Bolt	Unlatches
Exocytosis	Cell	Excrete
Flamingo Filter...	Flamingo	Filter
Gyroscope Foll...	Gyroscope Foll...	Transfer
Gyroscope Foll...	Gyroscope Foll...	Transfer

Basilisk Lizard Walks on Water is related to these other functions.
Double-click on a function name to navigate to that function.

Basilisk Lizard Walks on Water → Sliding Filament Model

Description | Model View

Name of the Function: Basilisk Lizard Walks on Water
Verb that characterizes the Function: Walk
Subject of the Verb: Basilisk Lizard
Object(s) of the Verb: Self
Preposition condition(s): on Water
Adverb condition(s):
Behavior: Behavior Model
Structure: Structure model

Fig. B.12. *Structured information for a selected function in DANE.* The user has selected a model in the left-hand panel and then clicked the "Model View" tab in the right-hand panel. This view displays structured information about the subject, verb, direct object(s), preposition phrase(s) or operating environments, and adverb(s) of the function related to this model.

6. Deployment and assessment of DANE

We deployed DANE in the Fall 2009 semester session of the project-based, senior-level, undergraduate ME/ISyE/MSE/PTFe/BIOL 4740 course on biologically inspired design taught by Georgia Institute of Technology's Center for Biologically Inspired Design. Although student teams were offered extra credit for adding a model to DANE, we did not connect the software with any specific learning objective. Our goal was exploratory: how, if at all, would teams integrate our CAD system into their design workflows?

The class composition was multi-disciplinary, comprising of 15 biology students, 11 mechanical engineering students, and 14 other students from a variety of backgrounds including biomedical engineering, chemical engineering, industrial engineering, material science, mathematics, and a few other engineering fields.

6.1. The biologically inspired design class

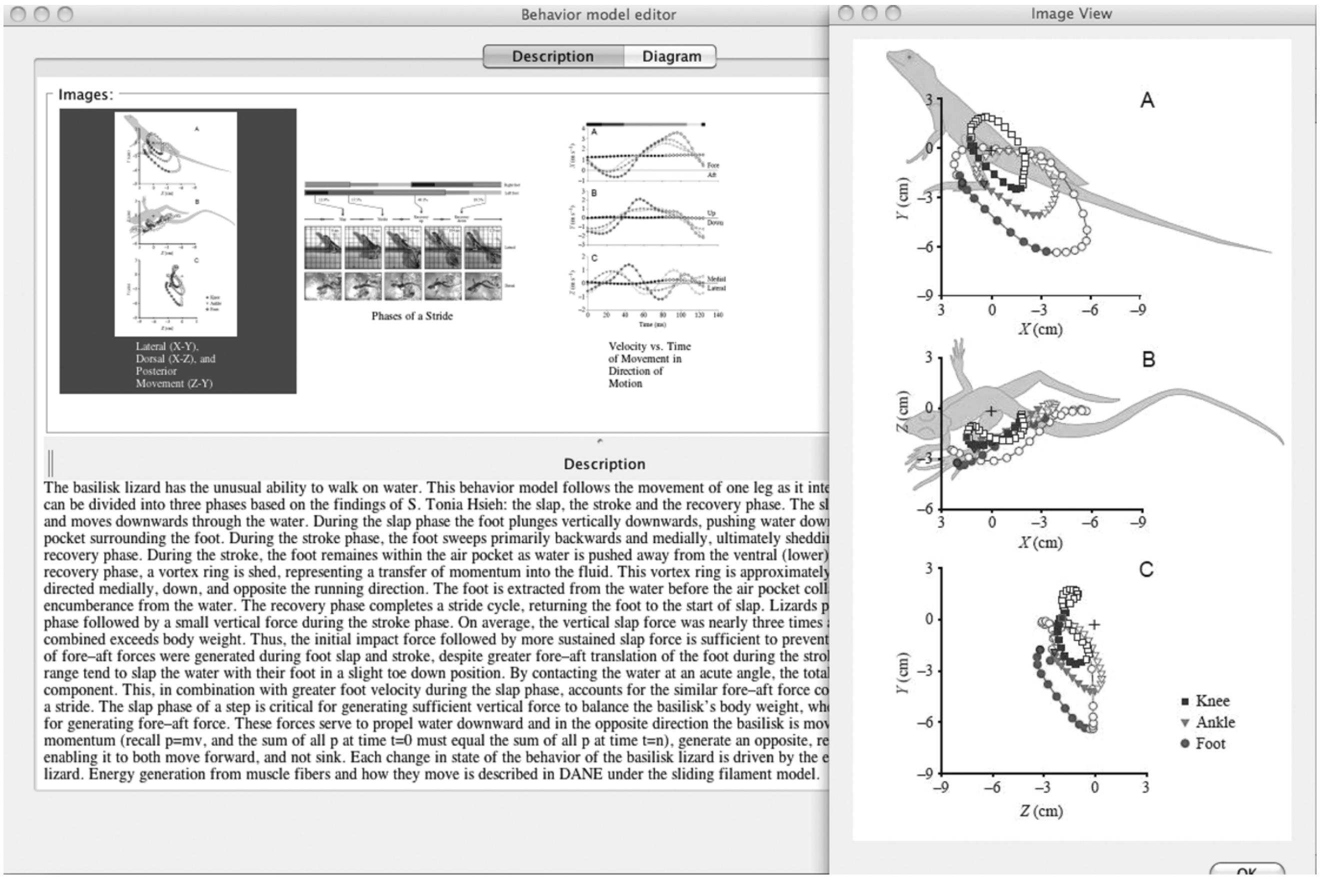
Georgia Tech's ME/ISyE/MSE/PTFe/BIOL 4740 course is broken up into three components: lectures, found object exercises, and a semester-long biologically inspired design team project. The lectures are of many types, for example: exposing the designers to case-studies in biologically inspired design, explaining the design processes for biologically inspired design, reframing engineering problems in a biological language, breaking a problem up into functional components, the identification and use of analogy in design, and quantitative analysis.

The found object exercises tasked students to bring in biological objects from their environments and analyze the functions of these objects. For example, a student might bring in an acorn and discuss how it performs the function of seed protection and dispersal. These exercises are meant to both expand the student's understanding of biology and encourage them to think deeply about the functions of biological systems.

Lastly, the students are engaged throughout the duration of the course in a team-based design project. Groups of 4–6 students are formed so that the design teams have at least one biology student and students from different schools of engineering. Each team was given a broad problem in the domain of dynamic, adaptable, sustainable housing such as heating or energy use. Teams were expected to refine the problem and then design a biologically inspired solution based on one or more biological sources to solve it. All teams presented their final designs during the end of the class and submitted a final design report.

The biologically inspired design class typically is taught without any special computational aids for design or research. Students are encouraged to perform their own research on biological systems through publicly available resources such as Google Scholar [129], Encyclopedia of Life [130], Web of Science [131], and Ask Nature [132]. While these sources contain quality information, they often return an overwhelming number of results, and results are often presented in a scientific language that is especially challenging for the non-biologists in the class to understand.

Further, students transmit information about their research to one another via PDF copies of scientific articles, meaning that all members of a team must read the raw sources. This perpetuates the problem of source materials being difficult to understand.


Behavior model editor
Description
Diagram
Images:
Lateral (X-Y), Dorsal (X-Z), and Posterior Movement (Z-Y)
Phases of a Stride
Velocity vs. Time of Movement in Direction of Motion
Image View
A
B
C
Y (cm)
X (cm)
Z (cm)
Knee
Ankle
Foot

Description

The basilisk lizard has the unusual ability to walk on water. This behavior model follows the movement of one leg as it inte
can be divided into three phases based on the findings of S. Tonia Hsieh: the slap, the stroke and the recovery phase. The sl
and moves downwards through the water. During the slap phase the foot plunges vertically downwards, pushing water dow
pocket surrounding the foot. During the stroke phase, the foot sweeps primarily backwards and medially, ultimately sheddi
recovery phase. During the stroke, the foot remaines within the air pocket as water is pushed away from the ventral (lower)
recovery phase, a vortex ring is shed, representing a transfer of momentum into the fluid. This vortex ring is approximately
directed medially, down, and opposite the running direction. The foot is extracted from the water before the air pocket coll
encumberance from the water. The recovery phase completes a stride cycle, returning the foot to the start of slap. Lizards p
phase followed by a small vertical force during the stroke phase. On average, the vertical slap force was nearly three times
combined exceeds body weight. Thus, the initial impact force followed by more sustained slap force is sufficient to prevent
of fore–aft forces were generated during foot slap and stroke, despite greater fore–aft translation of the foot during the strol
range tend to slap the water with their foot in a slight toe down position. By contacting the water at an acute angle, the total
component. This, in combination with greater foot velocity during the slap phase, accounts for the similar fore–aft force co
a stride. The slap phase of a step is critical for generating sufficient vertical force to balance the basilisk's body weight, wh
for generating fore–aft force. These forces serve to propel water downward and in the opposite direction the basilisk is mov
momentum (recall p=mv, and the sum of all p at time t=0 must equal the sum of all p at time t=n), generate an opposite, re
enabling it to both move forward, and not sink. Each change in state of the behavior of the basilisk lizard is driven by the e
lizard. Energy generation from muscle fibers and how they move is described in DANE under the sliding filament model.

Fig. B.13. *Textual description and image modalities in DANE.* Functions, behaviors, structures, behavior states, and objects or structural components all contain textual descriptions and images, and they are all displayed in the format presented in this figure. Here the user has double-clicked on the top-left picture and popped up an enlarged version for his or her inspection.

Our motivation for deploying DANE in this class was to test it in a real-world situation. Although this setting does not easily allow for formal controlled experiments and does not permit collection of certain types of data, it does enable observation of problem solving by real teams of people, as well as problem solving over an extended period of time. In our case, we felt that placing DANE *in situ* would provide a more accurate depiction of its usefulness, strengths, and weaknesses, as students might use it in ways that we did not anticipate. Moreover, students would only use DANE if they saw clear benefits to do so.

6.2. Training and instructions

At the end of the third week of the class, our tool was introduced during class-time through an hour long tutorial session presented by us. The goal of this tutorial was to familiarize students with DANE such that they would be capable of using it without our intervention.

At the time of our tutorial, students were already comfortable with the idea of biologically inspired design, were grouped in their semester design teams, and were aware of their semester-long project. Each team had brought at least one computer available to them from which the application was accessible.

The lesson began with a short discussion on the goal of DANE and an overview of SBF models. The point of this initial presentation was to motivate DANE and get students acquainted to the kind of representations that exist within the software.

We then instructed the students to go to our deployment website to launch DANE. We provided students with temporary credentials so that they could log in to the application. After the tutorial session, individual credentials and a copy of the URL where they could launch DANE on their home computers were provided to the students via individual email communications.

The rest of the hour proceeded as follows: the presenter discussed an aspect of DANE, for example how to build a function model, and then demonstrated its use. Students were then encouraged to try the same functionality on their copies of DANE. The students chose a organism they were familiar with from their own projects to act as the topic for entry during this exercise; we were not involved in the content of their models unless they specifically asked for help regarding the content. Those of us that were not presenting walked the room, observing the students performing these actions and answering any questions that arose.

Once the tutorial session concluded, the students were told to direct any additional questions to an online web forum, accessible through the class portal that all students were familiar with using. We did not provide any more instructions to the students. However, we encouraged them to use the application whenever they felt it would be appropriate throughout the semester.

6.3. Assessment and results

The following five kinds of data were obtained during the deployment of DANE. (1) An online traffic counter recorded how many people used our application-launching web site, which gave us rough information on how often DANE was used. (2) We kept a record of the models that were built in DANE by the users, and (3) we kept a log of the online troubleshooting forum. (4) After the class, we interviewed a student from the class about her opinions and experiences with DANE. (5) The course instructor

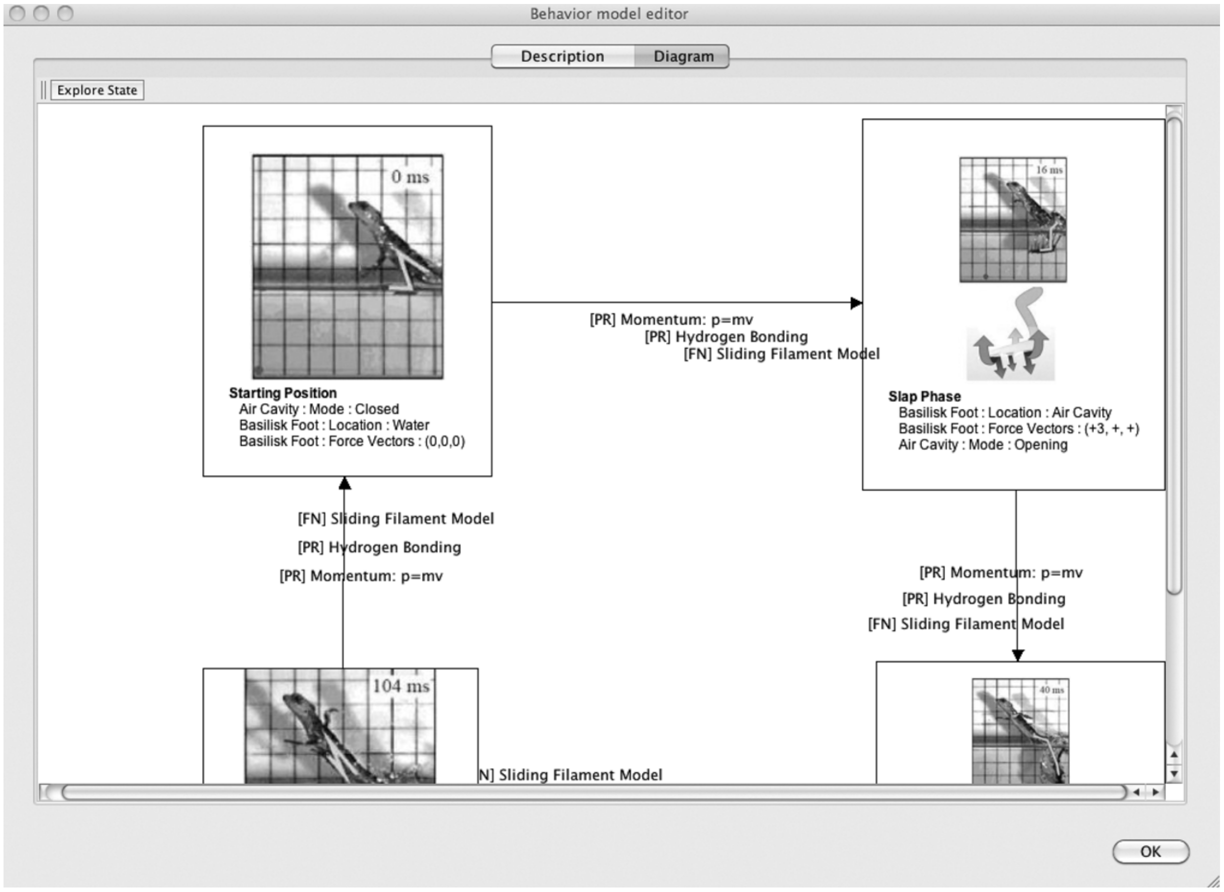

Behavior model editor
Description
Diagram
Explore State
0 ms
Starting Position
Air Cavity : Mode : Closed
Basilisk Foot : Location : Water
Basilisk Foot : Force Vectors : (0,0,0)
[PR] Momentum: p=mv
[PR] Hydrogen Bonding
[FN] Sliding Filament Model
16 ms
Slap Phase
Basilisk Foot : Location : Air Cavity
Basilisk Foot : Force Vectors : (+3, +, +)
Air Cavity : Mode : Opening
[FN] Sliding Filament Model
[PR] Hydrogen Bonding
[PR] Momentum: p=mv
[PR] Momentum: p=mv
[PR] Hydrogen Bonding
[FN] Sliding Filament Model
104 ms
N] Sliding Filament Model
40 ms
OK

Fig. B.14. *Example of a behavior model in DANE*. This figure displays a segment of a behavior diagram, presented as states and transitions, in DANE of the model "Basilisk Lizard Walks on Water". Note that, in the transitions, "[FN]" describes a sub-function and "[PR]" describes a principle.

made available to us the final project reflections, where students discussed the process by which they researched and designed their projects. This data formed the basis for determining if and how DANE was useful to end users in the context of biologically inspired design.

We observed that 9 new models were entered into the system. All models were related to some biological system (e.g., "Baleen ram filter feeding apparatus") or design idea (e.g., "Recycle Graywater"). Recall that a full system model in DANE contains a function, a behavior model, a structure model, and textual descriptions and images for function, structure, and behavior. Of the models entered by students, all had functions, three had behavior models, two had structure models, and two had textual descriptions for their functions. None had textual descriptions for their behaviors or structures, and none had images. Qualitatively speaking, all the models entered by students were incomplete by our standards. However, this incompleteness did not necessarily mean the students found their own models unhelpful, which will become apparent by the student interview.

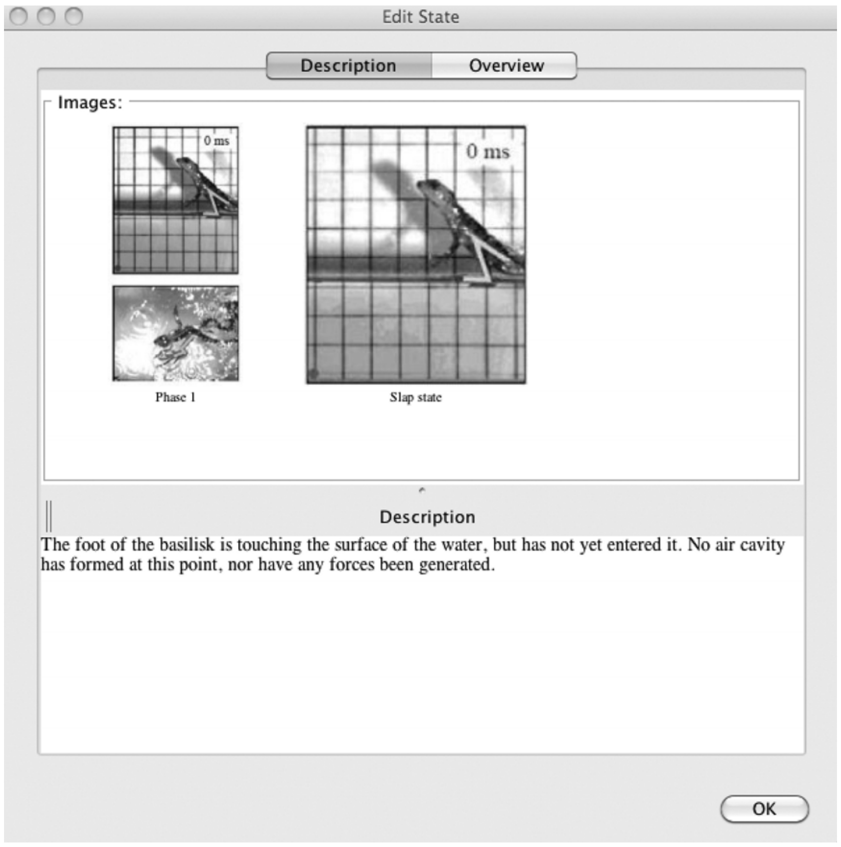
Our online troubleshooting forums contained four sub-sections: "Usability and Interface Issues" received 1 question; "Suggestions" also received 1 question; "How to Build Content" received 3 questions; and "DANE Bugs" received 2 questions. All the questions in the forum were technical in nature. No question was targeted at our representation schemata. Interestingly, the same student posted all the questions.

A 14-question interview about DANE was conducted after the semester was over with the student that posted the questions in our online forum. Although a single student obviously is not a sufficient sample for how the entire class felt about our tool, we felt this student in particular (due to her apparent engagement with DANE) could provide valuable feedback about our deployment. The interview was taped and then transcribed with permission of the interviewee. Questions were both subjective (e.g., "Did DANE improve your understanding of biological systems?") and objective (e.g., "Approximately how many hours, if any, did you use DANE?").

When asked how she would rate the DANE training session from 1 to 10 (with 10 being completely effective) and why, the student said she would rate it a 9 because the presenter "went over all the basic things" and "it was reasonable that, like, everybody in the class would understand how to use DANE in that training session".

Regarding her use of the tool, the student reported that she used it for approximately 20 h and mainly before class presentations because the professor gave extra credit if the team built a model on one of their 25 "inspired objects", which were objects in nature from which they drew analogies. This answer correlates with the usage patterns. Students were encouraged before presentation dates to use DANE for extra credit, so they did, causing usage to peak during those times.

When asked how she would rate the importance of DANE to her semester-long project on a scale from 1 to 10 (with 10 being of vital importance), the student gave a rating of 5, stating "it wasn't extremely, crucially vital, but it wasn't something that was not necessary" and "in the end we could've probably done without it, but I think it helped us to conceptualize". Later in the interview when probed about what she meant by "conceptualize", the student responded, "I mean, like, conceptualize, like, I think in boxes. Only because I'm in industrial engineering so I think in a lot of—I mean they look like flow charts. So that's what I like about DANE so I could, like...build a flow chart, essentially. From, like, the beginning stage to the end stage of a process".



The foot of the basilisk is touching the surface of the water, but has not yet entered it. No air cavity has formed at this point, nor have any forces been generated.

Fig. B.15. *Expanded view of a behavior state in DANE.* By double-clicking on a state, a user can see a text description and set of images related to that state. The "Overview" tab re-displays the information one would see about the state in a behavior diagram (i.e., Fig. B.14).

Not all responses were positive. When asked if DANE improved her understanding of biological systems, the student said no because, according to her, "I wasn't looking up information. I was trying to input information into the database".

Finally, when asked if she would recommend that other students use DANE, she answered yes, stating it is a "good resource" for "trying to build the analogies. And for like visualizing the connections, like the different properties. Like when my team first looked at it our overall function was regulate, and from regulate we had like regulate water, regulate energy, regulate heat, and you could just like break that up and you could go into DANE and see which- like we all independently like came up with objects in nature that had these properties and see if they were tied to each other". In addition to analogy-making, she said that DANE would save herself and other students work if it contained a small set of systems that were relevant to the topic of the class, as this would be an easier database to browse than Google or Web of Science.

Students in the class were asked to write a final paper that reflected upon their experiences in the class. 36 such reflection papers were submitted. In six of those, DANE was mentioned by name. In two papers the comments were explicitly positive (e.g., "I thought that DANE was a very useful tool to help decompose our system into its parts" and "A resource database (DANE!) would be VERY helpful in this class"). In one paper the comments were explicitly negative (e.g., "DANE did not really help in our communication" and "it had good intentions, but I did not feel that it had great potential as an aide"). Three of the six reflections mentioned DANE as a research repository, two described it as a modeling tool, and one described it in terms of aiding communication.

6.4. *Lessons learned and the fifth C: knowledge cost*

Based on the observed results of our deployment, we have drawn several lessons. The first is overcoming the cost/benefit hurdle of systems requiring intensive knowledge engineering. Students were not willing to invest the time and effort to build models because they saw no personal benefit. Likewise, without a sufficient number of models, students found the system of little use as a reference resource. However, at 40–100 h per model, a small group of isolated researchers has little hope of creating sufficient breadth for general usability. Thus, we believe it is important that research groups building small libraries of biological and/or technological systems for supporting biologically inspired design form collaborative efforts that combine their knowledge bases in productive ways.

The primary value to students of DANE was as an organizational framework to (1) organize their understanding of systems and (2) test their own ability to represent a design case. In our student interview, the student mentions that DANE was a useful tool for conceptualizing systems and in making analogies. Additionally, she said that the repository would improve her research process if enhanced with models that were relevant to the topic of the class. While we developed DANE with both of these benefits in mind, we incorrectly assumed students would build and share models, which would incrementally enhance the value of the tool.

Although DANE only explicitly appeared in one-sixth of the final reflections, the perspectives provided, if only anecdotal due to our sample size, are illuminating. Comparing these results to the four C's of next generation CAD systems, we can see that some students used the tool to help conceptualize their design, to enhance creativity via analogical reasoning, and, perhaps to a lesser

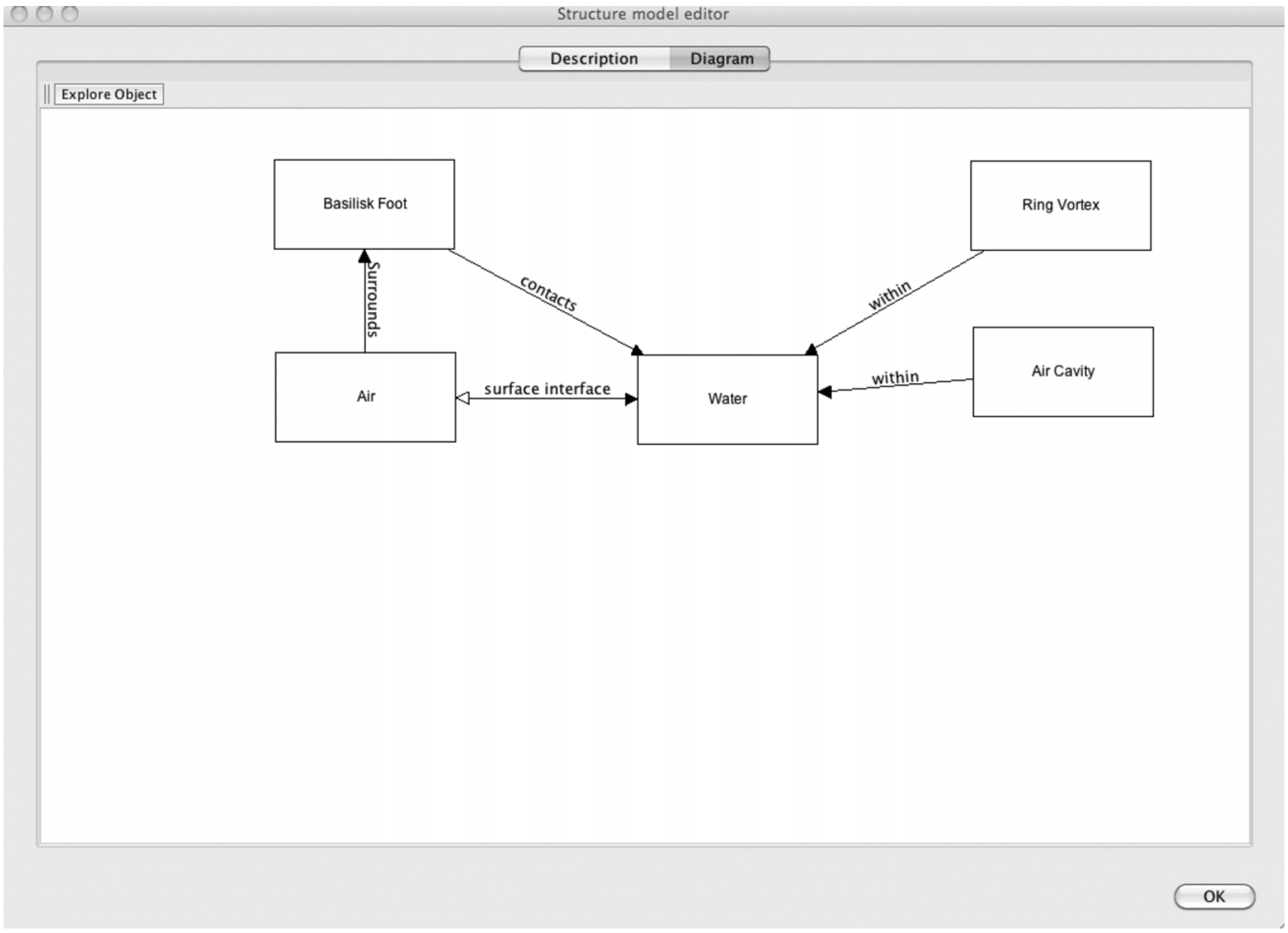


Fig. B.16. *Structure diagram in DANE.* This figure displays a structure diagram in DANE for the model "Basilisk Lizard Walks on Water". Nodes are structural components, and edges represent physical connections between those components.

extent, to collaborate across disciplines by browsing its knowledge repository. Moreover, the fact that students wrote about DANE in their final reflections without being prompted to do so acts as evidence that, four months after the application's deployment, some students were still aware of DANE and thinking about it in terms that align with how we hoped they would think about it. Note that we go into further details about DANE's relation to the four C's in Section 7.2.

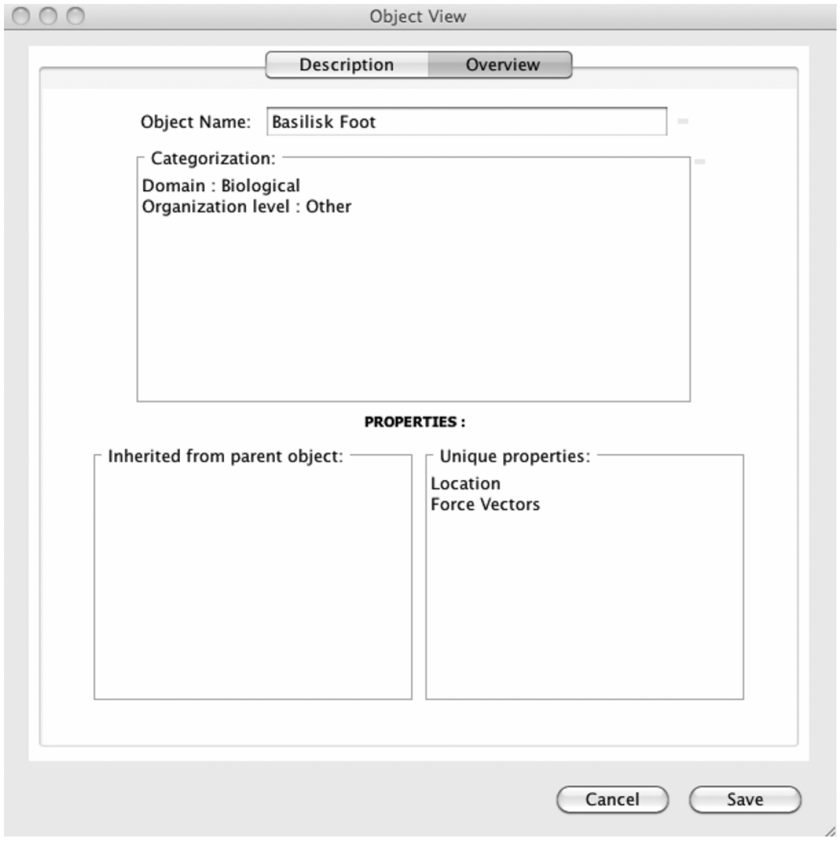
However, our other observations suggest that students were largely unconvinced of DANE's usefulness in whatever role they perceived it filling. Over half of the days the application was deployed received less than 10 hits; we had only one user engaged in our support forums; and our traffic peaks nearly always occurred during times when those peaks could be explained either by novelty (the peak right after the initial deployment/credential handout) or by an offer of extra credit (the peaks near the presentation times). Going forward, we will focus on how to better communicate our application's benefits to the students and encourage them to use the application.

Another lesson comes from the quality of the student-built models in DANE. The student models are incomplete, often lacking the important associated behavior and structure models. Although the student we interviewed described our training session as effective, the model sparseness might suggest that students did not understand the training session as well as the one we interviewed. Alternatively, the models could be the result of students being uninterested in DANE and doing only the minimal amount of work required to get their extra credit, which returns to the problem of motivation. The models could also be a symptom of not knowing their biological systems well enough to articulate them in a model.

Regardless of the reason for the sparse models, this situation illustrates the need for more intervention with regard to the students' grasp of DANE. Had we done interviews during the semester instead of only after the semester, we would be able to say with more certainty what caused the sparse models. In future deployments of DANE, we will remember this lesson.

7. DANE as a case study in the 4C's of next generation CAD systems

We presented in this paper a knowledge-based CAD system called DANE for functional modeling of biological systems in the context of biologically inspired design. The design literature describes a small but increasing number of interactive computational tools for supporting biologically inspired design (e.g., [132–137]). The Biomimicry Institute [132], for example, has developed an online web portal called AskNature for accessing a functionally-indexed database of scholarly articles relevant to biologically inspired design. Chakrabarti et al. [133] have developed an interactive tool called Idea-Inspire for supporting biomimetic idea generation in product design. The Idea-Inspire system represents the function, behavior, and structure of biological and engineered systems, and supports product designers with automated search for biological and engineering analogues. Sarkar and Chakrabarti [134] report on experiments with Idea-Inspire that show that the sources of inspiration suggested by Idea-Inspire, that range from text and diagrams to audio and video, have a significant influence on the representations, number, and quality of the generated ideas. In contrast, Chiu and Shu [135] describe a natural language approach



Object Name: Basilisk Foot

Categorization:
Domain : Biological
Organization level : Other

PROPERTIES :

Inherited from parent object:

Unique properties:
Location
Force Vectors

Fig. B.17. *Expanded view of a structural component or object in DANE*. By double-clicking on a structural component in Fig. B.16, the user gets this view. The description tab displays a text description and a set of images. The overview tab displays structured information about a Component that helps a user to better understand it.

that uses latent semantic indexing to retrieve biological systems related to technological functions by creating relationships between biological and engineering terms.

Our work on DANE differs from previous efforts in three ways. First, the design and development of DANE is based on our analysis of *in situ* studies of biologically inspired design [116]. Although Sarkar and Chakrabarti report on *post facto* experiments with Idea-Inspire, insofar as we know the design of Idea-Inspire was not based on any specific cognitive study biologically inspired design. Second, as described below, we have introduced DANE into a community of practice, viz., a classroom. Again insofar we know, Idea-Inspire so far has only been tested in laboratory settings. The distinction between evaluating in a community of practice versus a controlled environment is important in the context of biologically inspired design because literature suggests humans make different analogies in their natural environments than in laboratory settings [120,121]. Evaluation done in the setting of the design practice provides a more accurate picture of how the CAD system will actually perform. Third, DANE uses the Structure-Behavior-Function (SBF) scheme for modeling complex systems [45]. This is important because SBF models were developed in AI to support automated analogy-based design [32,36]. Thus, in the long term it should be possible to add automated inferences (e.g., to find relevant source analogs given a design problem) to DANE, increasing its value in supporting design. It is unclear at present whether this can be done in Idea-Inspire to the same degree of automation.

7.1. *Lessons from deployment of DANE in practice*

We also described in this paper the deployment of DANE to help designers performing biologically inspired design in a classroom setting. Although in retrospect it is clear that we struggled with properly motivating DANE's usage in the class and with gathering enough data to determine exactly how and why students were using it, deployment of DANE succeeded in the sense that the students were able to use the system when they wanted, and both the student we interviewed and two of the design journals said that DANE was a useful addition to the design teams' workflow.

Note that the results of our study with DANE are nowhere as neat or clean as those described by Sarkar and Chakrabarti [134] in their work on IDEA-INSPIRE. We believe this is primarily because Sarkar and Chakrabarti report on controlled experiments with individual designers working on selected problems for short durations in laboratory settings. In contrast, we deployed DANE in a large design class, the designers worked in teams, the teams selected their own problems, the problem solving unfolded over a semester, and we had access to only a small portion of the design teams' work. It is for this same reason that we could not measure the efficacy of DANE for design ideation using quantitative measures such as frequency, novelty, variety, and quality (e.g., [86]).

On the other hand, the *in situ* deployment of DANE in a naturalistic setting led us to the result about DANE's utility as a conceptualization tool. Although we had developed DANE largely as a library of SBF models of biological systems that designers may access to address their engineering problems, we found that at this stage of its development, designers found DANE more useful as a tool for conceptualizing a complex system, with the SBF scheme enabling the designers to organize their knowledge of complex systems. We conjecture the utility of DANE as a design library may grow with the size of the library.

The lessons we learned from the design, development and deployment of DANE emphasize the need for deployment to be an iterative process and for early deployment with your target users *in situ*. Had we developed DANE in isolation and only tested it in controlled situations, the problem of motivation and the insight into the importance of DANE as a conceptualization tool (as opposed to primarily as a repository) would have been difficult, if not impossible, to realize.

7.2. DANE as an illustration of the 4C's

DANE was envisioned to provide some level of direct support for each of the four C's in our characterization of next generation of CAD systems. For collaboration, DANE provides at least two kinds of support. The first is the ability for designers and design teams to share functional models of biological systems. Individual designers can investigate and add to the repository of models, or update models of others, sharing their knowledge of systems across time and space. The second, subtler, means is by supporting interdisciplinary communication via the underlying modeling of Structure-Behavior-Function models. The modeling framework itself provides a shared vocabulary about systems and a shared framework for organizing one's own internal knowledge of a system.

For conceptual design support, DANE provides a repository of designs, indexed hierarchically by function and system. The indexing support enables designers to search for a wide variety of conceptually related system analogues, at various levels of abstraction and detail. As conceptual design is neither linear nor monolithic, but rather a series of explorative sorties into many concepts at many levels of detail, a system must provide flexibility in order to support the process. Thus, as the sophistication of the conceptual design matures, designers are able to move up and down levels of abstraction within a system, to identify the most appropriate for the state of the current conceptual design. Multimodal representation capabilities allow for the most appropriate representation to be applied at any given design state.

Proponents of biologically inspired design assert that this design paradigm, in which DANE is embedded, gives rise to creative designs. In fact, a major point of biologically inspired design is to jump out of the well-trodden problem definitions in engineering, and to spawn new ways of thinking about problems. Unlike conventional methods such as undirected brainstorming, browsing the Internet, or the perusal of a biology textbook, DANE provides means to rapidly access systems functionally and at multiple levels of abstraction. The marriage of cross-domain analogy and functional indexing enables finding applicable systems for the development of creative design solutions to difficult problems.

Finally, the design, development, and deployment of DANE is firmly grounded in our cognitive studies. We started the DANE project with *in situ* studies of a community of practice in biologically inspired design, viz., ME/ISyE/MSE/PTFe/BIOL 4740 at Georgia Tech. Our observations then informed the design of the DANE knowledge-based CAD tool (cf. Table A.1). We ended this phase of the project by deploying DANE in the community of design practice and examining its use. Thus cognitive studies of design formed the basis for the design, development and deployment of DANE.

7.3. DANE as a methodology for next generation CAD systems

Perhaps more importantly, we posit that our work on DANE is an illustrative example of a cognitively grounded methodology for the next generation of CAD systems. As we noted in the introduction, we have witnessed over the last three decades a large number of cognitive studies of design as well as very large number of CAD systems. We believe the time has come to bridge the two lines of research, not only by grounding the development, deployment, and assessment CAD systems in prior cognitive research, but also by conducting specific cognitive studies that inform the design and assessment of specific CAD systems. After all, the phrase "computer-aided design" contains three words, but only two — computer and design — typically receive much attention. We must not forget the obvious point that we develop CAD tools to aid humans. Thus, realization of the promise implicit in the phrase computer-aided design demands that the next generation of CAD systems be grounded in theories and studies of design cognition.

Acknowledgements

We are grateful to Professor Jeannette Yen, Director of Georgia Tech's Center for Biologically Inspired Design (www.cbid.gatech.edu), for her strong support and encouragement for this work. Yen also coordinated the ME/ISyE/MSE/PTFe/BIOL 4740 class in Fall 2006, Fall 2007, Fall 2008 and Fall 2009, when we conducted our cognitive studies of design teams engaged in biologically inspired design. We developed the interactive knowledge-based design tool DANE during 2008–09, where the design of DANE was informed by our early cognitive studies (e.g., [116]), In Fall 2009, we deployed and evaluated DANE in the ME/ISyE/MSE/PTFe/BIOL 4740 class. We thank the other instructors of the course in Fall 2009, including Professors David Hu, Craig Tovey and Marc Weissburg, We also thank the students of ME/ISyE/MSE/PTFe/BIOL 4740 class in Fall 2009, including Jing Li, who later became a research assistant in our Design and Intelligence Laboratory. We are grateful to the US National Science Foundation that has generously supported this research through an NSF CreativeIT Grant (#0855916) entitled "MAJOR: Computational Tools for Enhancing Creativity in Biologically Inspired Engineering Design". Finally, we thank the anonymous reviewers whose critiques significantly helped us improve this paper, and Yong Zeng and Imre Horvath, who pushed us to make this more optimal.

Appendix A. Tables

See Table A.1.

Appendix B. Figures

See Figs. B.1–B.17.

References

[1] Stumptner M. An overview of knowledge-based configuration. AI Communications 1997;10(2):111–25.
[2] Sussman G. Electrical design: a problem for artificial intelligence research. In: Proceedings of the 5th international joint conference on artificial intelligence. IJCAI-77. vol. 2. 1977.
[3] McDermott D. Circuit design as problem solving. In: Latombe JC, editor. AI and pattern recognition. CAD, North-Holland; 1978. p. 227–45.
[4] Latombe J, editor. Artificial intelligence and pattern recognition in CAD, proceedings of the international federation for information processing (IFIP) working conference. Grenoble, France, March 17–19, Amsterdam (New York): North-Holland Publishing Company; 1978.
[5] McDermott J. R1: a rule-based configurer of computer systems. Artificial Intelligence 1982;19(1):39–88.
[6] Brown DC, Chandrasekaran B. Design problem solving: knowledge structures and control strategies. San Mateo (Calif.): Morgan Kaufmann; 1989.
[7] Mittal S, Dym C, Morjaria M. PRIDE: an expert system for the design of paper handling systems. IEEE Computer 1986;19(7):102–14.
[8] Steinberg L. Design as refinement plus constraint propagation: the VEXED experience. In: Proc. national conference on AI. AAAI-87. 1987.
[9] Marcus S, Stout J, McDermott J. VT: an expert elevator designer that uses knowledge-based backtracking. AI Magazine 1988;9(1):95–111.
[10] McDermott J. R1 (XCON) at age 12: lessons from an elementary school achiever. Artificial Intelligence 1993;59:241–8.
[11] Finger S, Dixon J. A review of mechanical engineering design—I. Research in Engineering Design 1989;1(1):51–67.
[12] Finger S, Dixon J. A review of mechanical engineering design—II. Research in Engineering Design 1989;1(2):121–37.

[13] Tong Christopher, Sriram Duvvuru. Artificial intelligence in engineering design. San Diego (CA, USA): Academic Press; 1992.
[14] Dym C. Engineering design: a synthesis of views. Cambridge University Press; 1994.
[15] Katz R, Chang E, Bhateja Rajiv. Version modeling concepts for computer-aided design databases. In: Procs. ACM SIGMOD international conference on management of data. 1986. p. 379–86.
[16] Eastman C, Bond A, Chase S. A data model for design dabeltabases. In: Proc. first international conference on AI in design. 1991.
[17] Ahmed R, Navathe S. Version management of composite objects in CAD databases. In: Procs. ACM SIGMOD international conference on management of data. 1991. p. 218–27.
[18] Pearce M, Goel A, Kolodner J, Zimring C, Sentosa L, Billington R. Case-based decision support: a case study in architectural design. IEEE Expert 1992;7(5): 14–20.
[19] Goel A, Kolodner J, Pearce M, Billington R, Zimring C. Towards a case-based tool for aiding conceptual design problem solving. In: Proc. third DARPA workshop on case-based reasoning. 1991.
[20] Huhns M, Acosta E. Argo: a system for design by analogy. IEEE Expert, Fall 1988;53–68.
[21] Barber J, Bhatta S, Goel A, Jacobson M, Pearce M, Penberthy L, et al. AskJef: integration of case-based and multimedia technologies for interface design support. In: Gero J, editor. Artificial intelligence in design'92. Dordrecht: Kluwer; 1992. p. 457–74.
[22] Mostow J, Barley M, Weinrich T. Automated reuse of design plans. Artificial Intelligence in Engineering 1989;4(4):181–96.
[23] Pu P, Reschberger M. Assembly sequence planning using case-based reasoning techniques. Knowledge-Based Systems 1991;4(3):123–30.
[24] Sycara K, Guttal R, Koning J, Narasimhan S, Navinchandra D. CADET: a case-based synthesis tool for engineering design. International Journal of Expert Systems 1991;4(2):157–88.
[25] Hua K, Faltings B, Smith I. CADRE: case-based geometric design. Artificial Intelligence in Engineering 1996;10(2):171–83.
[26] Maher M, Balachandran M, Zhang D. Case-based reasoning in design. Lawrence Erlbaum; 1995.
[27] Navinchandra D. Case-based reasoning in CYCLOPS, a design problem solver. In: Kolodner J, editor. Proc. first DARPA workshop on case-based reasoning. Morgan Kauffman; 1988. p. 286–301.
[28] Smyth B, Keane M, Cunningham P. Hierarchical case-based reasoning: integrating case-based and decompositional problem-solving techniques for plant-control software design. IEEE Transactions on Knowledge and Data Engineering 2001;13(5):793–812.
[29] Gebhardt F, Voß A, Gräther W, Schmidt-Belz B. Reasoning with complex cases. Norwell (MA): Kluwer; 1997.
[30] Smith I, Lottaz C, Faltings I. Spatial composition using cases: IDIOM. Lecture notes in computer science, vol. 1010. Springer; 1995. p. 88–97.
[31] Goel A, Chandrasekaran B. Integrating case-based and model-based reasoning for design problem solving. In: Proc. AAAI-88 workshop on AI in design. 1988.
[32] Goel A, Bhatta S, Stroulia E. Kritik: an early case-based design system. In: Maher M, Pu P, editors. Issues and applications of case-based reasoning in design. Mahwah (NJ): Erlbaum; 1997. p. 87–132.
[33] Zhao F, Maher M. Using analogical reasoning to design buildings. Engineering with Computers 1988;4:107–22.
[34] Maher M, Pu P, editors. Issues and applications of case-based reasoning in design. Mahwah (NJ): Lawrence Erlbaum Associates; 1997.
[35] Goel A, Craw S. Design, innovation and case-based reasoning. Knowledge Engineering Review 2005;20(3):271–6.
[36] Goel A, Bhatta S. Use of design patterns in analogy-based design. Advanced Engineering Informatics 2004;18(2):85–94.
[37] Bhatta S, Goel A. Model-based indexing and index learning in engineering design. Engineering Applications of Artificial Intelligence 1996;9(6):601–10. special issue on Machine Learning in Engineering.
[38] Chandrasekaran B. Functional representations and causal processes. In: Yovits M, editor. Advances in Computers. 1994. p. 73–143.
[39] Chandrasekaran B. Functional representation: a brief historical perespective. Applied Artificial Intelligence 1994;8(2):173–97.
[40] Gero J, Tham K, Lee H. Behavior: a link between function and structure in design. In: Brown D, Waldron M, Yoshikawa H, editors. Intelligent computer aided design. Amsterdam: North Holland; 1992. p. 193–225.
[41] Gero J, Kannengiesser U. The situated function-behavior-structure framework. Design Studies 2004.
[42] Umeda Y, Takeda H, Tomiyama T, Yoshikawa H. Function, behavior, and structure. In: AIENG'90 applications of AI in engineering. Southernpton (Berlin): Computational Mechanics Publications and Springer-Verlag; 1990. p. 177–93.
[43] Umeda Y, Ishii M, Yoshioka M, Shimomura Y, Tomiyama T. Supporting conceptual design based on the function-behavior-state modeler. (AI EDAM) Artificial Intelligence for Engineering Design, Analysis and Manufacturing 1996;10(44):275–88.
[44] Umeda Y, Tomiyama T. Functional reasoning in design. IEEE Expert 1997; 12(2):42–8.
[45] Goel A, Rugaber S, Vattam S. Structure, behavior and function of complex systems: the structure, behavior, function modeling language. (AI EDAM) Artificial Intelligence for Engineering Design, Analysis and Manufacturing 2009;23:23–35.
[46] Goel A, Stroulia E. Functional device models and model-based diagnosis in adaptive design. (AI EDAM) Artificial Intelligence for Engineering Design, Analysis and Manufacturing 1996;10:355–70. Special issue on Functional Representation and Reasoning.
[47] Prabhakar S, Goel A. Functional modeling for enabling adaptive design of devices for new environments. Artificial Intelligence in Engineering 1998;12: 417–44.
[48] Faltings B, Sun K. FAMING: supporting innovative mechanism shape design. Computer Aided Design 1995.
[49] Joskowicz L, Sacks E. Computational kinematics. Artificial Intelligence 1991; 51(1–3).
[50] Joskowicz L, Sacks E, Srinivasan V. Kinematics tolerance analysis. Computer-Aided Design 1998;29.
[51] Gross M, Do E. Drawing on the back of an envelope: a framework for interacting with application programs by freehand drawing. Computers & Graphics 2000;24(6):835–49.
[52] Yaner P, Goel A. Visual analogy: viewing retrieval and mapping as constraint satisfaction. Journal of Applied Intelligence 2006;25(1):91–105.
[53] Stahovich T. Artificial intelligence for design. In: Antonsson EK, Cagan J, editors. Formal engineering design synthesis. Cambridge University Press; 2001. p. 228–69.
[54] Tomiyama T. Intelligent computer-aided design systems: past 20 years and future 20 years? (AI EDAM) Artificial Intelligence for Engineering Design, Analysis and Manufacturing 2007;21(1):27–9.
[55] Szykman S, Bochenek C, Racz J, Senfaute J, Sriram R. Design repositories: engineering design's new knowledge base. IEEE Intelligent Systems 2000; 15(3):48–55.
[56] Szykman S, Sriram R, Regli W. The role of knowledge in next-generation product development systems. ASME Transactions, the Journal of Computing and Information Science in Engineering 2001;1(1):3–11.
[57] Kitamura Y, Sano T, Namba K, Mizoguchi R. A functional concept ontology and its application to automatic identification of functional structures. Advanced Engineering Informatics 2002.
[58] Kitamura Y, Kashiwase M, Fuse M, Mizoguchi R. Deployment of an ontological framework for functional design knowledge. Advanced Engineering Informatics 2004;18(2).
[59] Murdock J, Szykman S, Sriram R. An information modeling framework to support design databases and repositories. In: Proc ASME design engineering technical conference. 1997.
[60] Hirtz J, Stone RB, McAdams DA, Szykman S, Wood KL. A functional basis for engineering design: reconciling and evolving previous efforts. Research in Engineering Design 2002;13:65–82. 1728–33.
[61] Anthony L, Regli W, John J, Lombeyda S. An approach to capturing structure, behavior, and function of artifacts in computer-aided design. Journal of Computing and Information Science in Engineering 2001;1(2):186–93.
[62] Stahovich T, Davis R, Shrobe H. Generating multiple new designs from a sketch. Artificial Intelligence 1998;104(1–2):211–64.
[63] Yaner P, Goel A. From design drawings to structural models by compositional analogy. (AI EDAM) Artificial Intelligence for Engineering Design, Analysis and Manufacturing 2008;22(2):117–28. Special issue on Multimodal Design.
[64] Davies Jim, Goel Ashok, Nersessian Nancy. A computational model of visual analogies in design. Journal of Cognitive Systems Research 2009;10:204–15. special issue on Analogies—Integrating Cognitive Abilities.
[65] Arciszewski T, Bloedorn E, Michalski R, Mustafa M, Wnek J. Machine learning of design rules: methodology and case study. Journal of Computing in Civil Engineering, ASCE 1994;8(3):286–308.
[66] Bhatta S, Goel A. Discovery of physical principles from design experiences. (AI EDAM) Artificial Intelligence for Engineering Design, Analysis and Manufacturing 1994;8(2):113–23. Special issue on Machine Learning in Design.
[67] Sim S, Duffy A. A foundation for machine learning in design. (AI EDAM) Artificial Intelligence for Engineering Design, Analysis and Manufacturing 1998;12(2):193–209.
[68] Sim S, Duffy A. Evolving a model of learning in design. Research in Engineering Design 2004;15(1):40–61.
[69] Stahovich T. LearnIT: an instance-based approach learning and reusing design strategis. Journal of Mechanical Design, ASME 2000;122:149–56.
[70] Cross N, Christiaans H, Dorst K, editors. Analysing design activity. Chichester: Wiley; 1996.
[71] Christensen B, Schunn C. The relationship of analogical distance to analogical function and preinventive structure: the case of engineering design. Memory and Cognition 2008;35(1):29–38.
[72] Eastman C, Newstetter W, McCracken M, editors. Design knowing and learning: cognition in design education. 2001.
[73] Visser W. Two functions of analogical reasoning in design: a cognitive-psychology approach. Design Studies 1996;17:417–34.
[74] Ahmed S, Wallace K, Blessing L. Understanding the differences between how novice and experienced designers approach design tasks. Research in Engineering Design 2002;14(1):1–11.
[75] Akin O, Akin C, Athavankar U. Mental imagery as a design tool. Cybernetics and Systems 1996;28(1):25–42. 1997. Frames of reference in architectural design: analysing the hyperacclamation (A-h-a-!) Design Studies, 17 (4): 341–361.
[76] Bonnardel N. Towards understanding and supporting creativity: analogies in a constrained cognitive environment. Knowledge-Based Systems 2000; 505–13.

[77] Casakin H, Goldschmidt G. Expertise and the use of visual analogy: implications for design education. Design Studies 1999;20(2):153–75.
[78] Cross N. The nature and nurture of design ability. Design Studies 1990;11(3): 127–40.
[79] Gero J, McNeil T. An approach to the analysis of design protocols. Design Studies 1998;19(1):21–61.
[80] Goel V, Grafman J. The role of the right prefrontal cortex in ill-structured problem solving. Cognitive Neuropsychology 2000;17(5):415–36.
[81] Goel V, Pirolli P. The structure of design problem spaces. Cognitive Science 1992;16:395–429.
[82] Goldschmidt G. On visual design thinking: the vis. kids of architecture. Design Studies 1994;15(2):158–74.
[83] Jansson D, Smith S. Design fixation. Design Studies 1991;12(1):3–11.
[84] Moss J, Kotovsky K, Cagan J. Expertise differences in the mental representation of mechanical devices in engineering design. Cognitive Science 2005.
[85] Oxman R. Design by re-representation: a model of visual reasoning in design. Design Studies 1997;18(4):329–47.
[86] Shah J, Vargas-Hernandez N, Smith S. Metrics for measuring ideation effectiveness. Design Studies 2003;24(2):111–34.
[87] Smith S, Ward T, Schumacher J. Constraining effects of examples in creative generation tasks. Memory and Cognition 1993;21(6):837–45.
[88] Suwa M, Purcell T, Gero J. Macroscopic analysis of design processes based on a scheme for coding designers' cognitive actions. Design Studies 1998;19(4): 455–83.
[89] Suwa M, Tversky B. What do architects and students perceive in their design sketches? A protocol analysis. Design Studies 1997;18(4):385–403.
[90] Visser W. More or less following a plan during design: opportunistic deviations in specification. International Journal of Man-Machine Studies 1990;33(3):247–78.
[91] Bucciarelli L. Designing engineers. MIT Press; 1994.
[92] Bucciareilli L. An ethnographic perespective on engineering design. Design Studies 1988;9(3):159–68.
[93] Cutkosky M, Engelmore R, Fikes R, Gruber T, Genesereth M, Mark W, Tenenbaum J, Weber J. PACT: an experiment in integrating concurrent engineering systems. IEEE Computer 1993;26(1):28–37.
[94] Edmonds E, Candy L, Jones R, Soufi B. Support for collaborative design: agents and emergence. Communications of the ACM 1994;37:41–7.
[95] Fischer G, Grudin J, Lemke A, McCall R, Ostwald J, Reeves B, et al. Supporting indirect collaborative design with integrated knowledge-based design environments. Human-Computer Interactions 1992;7:281–314.
[96] Fischer G, Ostwald J. Knowledge management—problems, promises, realities, and challenges. IEEE Intelligent Systems 2001;60–72.
[97] Sriram D. Distributed and integrated collaborative engineering design. Sarven Publishers; 2002.
[98] Subrahmanian E, Monarch I, Konda S, Granger H, Collins M, Milliken R, Westerberg A. Boundary objects and prototypes at the interfaces of engineering design. Computer Supported Cooperative Work (CSCW) 2003; 12(2):185–203.
[99] Kvan T. Collaborative design: what is it? Automation in Construction 2000; 9(4):409–15.
[100] McDonough E, Kahn K, Barczak G. An investigation of the use of global, virtual and colocated new product development teams. Journal of Product Innovation Management 2001;18(2).
[101] Pahl G, Beitz W. In: Wallace K, editor. Engineering design: a systematic approach. 2nd ed. Springer; 1996. [English edition].
[102] French M. Conceptual design for engineers. 3rd ed. Springer-Verlag; 1999.
[103] Sim S, Duffy A. Towards an ontology of generic engineering design activities. Research in Engineering Design 2003.
[104] Shah J, Kulkarni S, Vargas-Hernandez N. Evaluating the effectiveness of idea generation techniques in design: metrics and experimental methodology. ASME Transactions, Journal of Mechanical Design 2000;122(4):377–84.
[105] Finger S, Dixon J. A review of research in mechanical engineering design. Part 1: descriptive, prescriptive and computer-based models of design processes. Representations, analysis, and design for the life cycle. Research in Engineering Design 1989;1:51–67.
[106] Welch R, Dixon J. Guiding conceptual design through behavioral reasoning. Research in Engineering Design 1994;6:169–88.
[107] Gorti S, Sriram R. From symbol to form: a framework for conceptual design. Computer Aided Design 1996;23(11):853–70.
[108] Goel A. Design, analogy, and creativity. IEEE Expert 1997;12(3):62–70.
[109] Benyus J. Biomimicry: innovation inspired by nature. New York: William Morrow; 1997.
[110] Vincent J, Mann D. Systematic transfer from biology to engineering. Philosophical Transactions of the Royal Society of London 2002;360:159–73.
[111] Bar-Cohen Y, editor. Biomimetics: biologically inspired technologies. Taylor & Francis; 2006.
[112] Yen J, Weissburg M. Perspectives on biologically inspired design: introduction to the collected contributions. Journal of Bioinspiration and Biomimetics 2007;2.
[113] Srinivasarao M, Padilla L. Biologically inspired design: color on wings. In: Proc. materials research society symposium. Spring. vol. 479. 1997.
[114] Nakrani S, Tovey C. On honey bees and dynamic server allocation in internet hosting centers. Adaptive Behavior 2004;12:223–40.
[115] Bonser R, Vincent J. Technology trajectories, innovation, and the growth of biomimetics. Proceedings of the Institution of Mechanical Engineers, Part C (Journal of Mechanical Engineering Science) 2007;1177–80.
[116] Helms M, Vattam S, Goel A. Biologically inspired design: process and products. Design Studies 2009;30(5):606–22.
[117] Vattam S, Helms M, Goel A. A content account of creative analogies in biologically inspired design. AI for Engineering Design, Analysis and Manufacturing 2010;24:467–81. Special issue on Biologically Inspired Design.
[118] Mak T, Shu L. Using descriptions of biological phenomena for idea generation. Research in Engineering Design 2008;19(1):21–8.
[119] Linsey JS, Wood KL, Markman AB. Modality and representation in analogy. (AI EDAM) Artificial Intelligence for Engineering Design, Analysis and Manufacturing 2008;22:85–100.
[120] Dunbar K. How scientists really reason: scientific reasoning in real-world laboratories. In: Sternberg RJ, Davidson J, editors. Mechanisms of insight. Cambridge (MA): MIT Press; 1995. p. 365–95.
[121] Dunbar K. The analogical paradox. In: Gentner D, Holyoak KJ, Kokinov BN, editors. The analogical mind: perspectives from cognitive science. MIT Press; 2001.
[122] Kruger C, Cross N. Solution-driven vs. problem-driven design: strategies and outcomes. Design Studies 2006;27(5):527–48.
[123] Nersessian NJ. How do scientists think? capturing the dynamics of conceptual change in science. In: Giere RN, editor. Cognitive models of science. Minneapolis (MN): University of Minnesota Press; 1992. p. 3–45.
[124] Nersessian NJ. Model-based reasoning in conceptual change. In: Magnani L, Nersessian NJ, Thagard P, editors. Model-based reasoning in scientific discovery. New York: Kluwer Academic, Plenum Publishers; 1999. p. 5–22.
[125] Wills L, Kolodner J. Towards more creative case-based design systems. In: Procs. twelveth national conference of the American association for artificial intelligence. 1994. p. 50–5.
[126] Wills L, Kolodner J. Explaining serendipitous recognition in design. In: Proc. 16th cognitive science conference. Lawrence Erlbaum; 1994. p. 940–5.
[127] Goel A, Gomez A, Grue N, Murdock M, Recker M, Govindaraj T. Explanatory interface in interactive design environments. In: Gero J, Sudweeks F, editors. Proc. fourth international conference on AI in design. Palo Alto, June 1996, Boston (MA): Kluwer Academic Press; 1996. p. 1–20.
[128] Erden M, Komoto H, van Beek T, D'Amelio V, Echavarria E, Tomiyama T. A review of function modeling: approaches and applications. (AI EDAM) Artificial Intelligence for Engineering Design, Analysis and Manufacturing 2008;22(2):147–69.
[129] Google scholar. http://scholar.google.com.
[130] Encyclopedia of life. http://www.eol.org.
[131] Web of science. http://wokinfo.com/.
[132] Biomimicry Institute. Ask nature—the biomimicty design portal. 2008. http://www.asknature.org.
[133] Chakrabarti A, Sarkar P, Leelavathamma B, Nataraju B. A functional representation for aiding biomimetic and artificial inspiration of new ideas. (AI EDAM) Artificial Intelligence for Engineering Design, Analysis and Manufacturing 2005;19:113–32.
[134] Sarkar P, Chakrabarti A. The effect of representation of triggers on design outcomes. (AI EDAM) Artificial Intelligence for Engineering Design, Analysis and Manufacturing 2008;22(02):101–16.
[135] Chiu I, Shu L. Biomimetic design through natural language analysis to facilitate cross-domain analysis. (AI EDAM) Artificial Intelligence for Engineering Design, Analysis and Manufacturing 2007;21:45–59.
[136] Shu L, Stone R, McAdams D, Greer J. Integrating function-based and biomimetic design for automatic concept generation. In: Proc. international conference on engineering design. ICED'07. 2007.
[137] Nagel R, Midha P, Tinsley A, Stone R, McAdams D, Shu L. Exploring the use of functional models in biomimetic concept design. Journal of Mechanical Design, ASME 2008;130(2).

Ashok K. Goel is an Associate Professor of Computer Science & Cognitive Science in the School of Interactive Computing at the Georgia Institute of Technology. He is Director of the School's Design & Intelligence Laboratory, and a Co-Director of Georgia Tech's Center for Biologically Inspired Design. He is an Associate Editor of IEEE Intelligent Systems and ASME Journal of Computing and Information Science in Engineering. His research has been supported by NSF, DARPA, ONR, DHS and IES, and he has been a technical consultant to NEC and NCR.

Swaroop Vattam is a candidate for Ph.D. in Computer Science at the Georgia Institute of Technology. He works in the Design & Intelligence Laboratory in the School of Interactive Computing, where he is investigating analogical reasoning and creativity in the context of biologically inspired design. His forthcoming Ph.D. thesis investigates mediated analogy, i.e., analogical problem-solving mediated by external information environments.

Bryan Wiltgen is a Ph.D. student in Computer Science at the Georgia Institute of Technology. He does research in the Design & Intelligence Laboratory in the School of Interactive Computing, studying creativity, analogical reasoning, and cognition in natural and social settings, such as biologically inspired design.

Michael Helms is a Ph.D. student in Computer Science at the Georgia Institute of Technology working in the Design & Intelligence Laboratory in the School of Interactive Computing. He has been an instructor in Biologically Inspired Design at Georgia Tech and at NASA, and is the graduate student coordinator at the Center for Biologically Inspired Design at Georgia Tech. Previous to his Ph.D. work, Michael worked as a business consultant with Teradata as well as holding various marketing and strategic planning positions with Zurich Financial Group.