Proceedings of DETC '97
1997 ASME Design Engineering Technical Conference
September 14-17, 1997, Sacramento, California

DETC97/DTM-3873

THE *N*-DIM APPROACH TO CREATING DESIGN SUPPORT SYSTEMS

Eswaran Subrahmanian
Engineering Design Research Center
Carnegie Mellon University
Pittsburgh, PA 15213, USA
Phone: (412) 268 5221
Fax: (412) 268 5229
email: sub@globe.edrc.cmu.edu

Suresh L. Konda, Allen Dutoit*
Software Engineering Institute
Carnegie Mellon University

Yoram Reich
Department of Solid Mechanics, Materials and Structure
Faculty of Engineering, Tel Aviv University
Ramat Aviv 69978, Israel
email: yoram@eng.tau.ac.il

Douglas Cunningham, Robert Patrick, Mark Thomas, Arthur W. Westerberg
Engineering Design Research Center
Carnegie Mellon University

ABSTRACT

Creating practical design support systems is a complex design endeavor. We approach it with an evolutionary process, one that studies the design information flow then builds and tests information management support systems. Through our experience with industrial partners we have evolved this process into a set of methods and tools that support these methods. We have evolved an infrastructure called *n*-dim, that is composed of a small number of building blocks that can be composed in ways that match the complexity of design contexts and work. We have developed this infrastructure to be highly flexible so as to allow us to conduct this evolutionary process in a practical project setting.

INTRODUCTION

Our approach to creating design support systems is influenced by several well documented observations regarding the nature of modern engineering design. In this paper, we motivate our approach based on a considerable body of empirical work and on the exigencies of supporting engineering design practice. Our argument is that an engineer's work is characterized by features which make the design information very complex. The goal in supporting such work, then, is to help the engineer tame this complexity. This requires, in turn, a support system that is capable of representing the information in all its complexities and is comprehensible, usable, and maintainable. Of course, one must also be able to build the environment within a reasonable time frame and budget.

In order to achieve this goal, we iteratively apply the following steps: study the design work, develop systems to support the work, and evaluate these systems by studying the new work environment after system deployment. While these steps are almost obvious, carrying them out under pragmatic conditions can be extremely difficult. In order to achieve and sustain the ability to intervene in a workplace and improve design practice in an organization, we need tools, methods for applying them, and a general philosophy that guides the process. Furthermore the philosophy, methods, and tools need to be internally consistent[1]. Our approach consists of a diverse set of tools and methods borrowed from a wide range of disciplines as required by the context being studied and an over-arching philosophy that guides in selecting the right tools and methods for each work context. We have used this approach in several industrial and academic contexts and the results reinforce our claim of this approach's value in supporting engineering design.

* The views and conclusions contained in this document are solely those of the authors and should not be interpreted as representing official policies, either expressed or implied, of the Software Engineering Institute, Carnegie Mellon University, the U.S. Air Force, the Department of Defense, or the U.S. Government.

1. In order to iterate this process in a reasonably efficient manner, we must have a computational infrastructure that supports such iterations by, for example, supporting easy scripting and testing with throw-away code.

Table 1: Our experience in studies of engineering design

Design Project	*Methods Employed*	*Focus*
Process Control system design (Westinghouse)	Direct observation of design meetings; collection of all design documents; recording meetings.	Preliminary design.
Integration of Material Databases (ALCOA)	Tracking information flows with a survey. Creating concept structures using semi-structured interviews.	Information sharing across divisions to reduce duplicated work.
CINERG: Multi-University Collaborative Distributed Design	Direct participation and observation. Analysis of documents and messages exchanged. Post hoc review.	Feasibility of electronic collaboration in asynchronous, distributed design with periodic face-to face meetings and conference calls.
Design of and manufacture of electric power devices (multiple studies)	Questionnaire and direct interviews with participants in all phases of the design manufacture and services. Analysis of critical documents.	Information need and flows in the design and manufacturing process (intra-project and inter-project flows).
Undergraduate project courses in software engineering	Analysis of design information including intermediate and final products and electronic communications among designers.	The effect of communication on outcome.

The outline of the paper is as follows. The first section, "The Nature of Engineering Work," discusses our understanding of engineering design as derived from empirical studies and documented observations. It highlights the complex heterogeneous context of design and the variety of information management activities that comprise engineering work. The next section, "Addressing Information Management," contends that, in order to address the complexity of design contexts, one has to match it with a corresponding variety of building blocks and ways to connect them. "*n*-dim: An Infrastructure for Information Modeling and Applications" discusses our approach to identifying these building blocks and an infrastructure called *n*-dim within which they can be composed (Levy et al., 1993). This section also reviews some basic features of *n*-dim, the continuously evolving infrastructure for developing design support systems. "How *n*-dim Addresses a Variety of Information Activities" illustrates how *n*-dim's features and some applications we have developed address the complexity of engineering design contexts and work.

THE NATURE OF ENGINEERING WORK

In order to understand the nature of engineering work as it is actually carried out in day to day practice, we present some of the more important findings from empirical observations of real design situations. This is followed by a brief discussion of the increasingly distributed and varied contexts within which design takes place. We can draw some conclusions regarding the nature of systems required to adequately support design activities in practical contexts.

Empirical Studies of Design

Empirical Studies in engineering design span a variety of objectives, use a diversity of methods and focus at different levels of granularity (Clark and Fujimoto, 1991; Hales, 1987; Kuffner and Ullman, 1991; Leifer, 1991; Subrahmanian 1992; Tang, 1989; Wilkins et al., 1989). They range from comprehensive product development studies (Clark and Fujimoto, 1991; Hales, 1987) to studies of individual designers (Bucciarelli, 1984). These studies provide a tapestry of design covering the organization of design, the evaluation of normative methods in design, group work around a table, information flow analysis, process-based analysis, and task-related analysis for cooperating groups. In this section, we briefly describe studies of design conducted by us which define and affirm our approach. Table 1 presents summaries of the design process studies we conducted or in which we participated. These studies approached design from different perspectives and employed a variety of methods to gather and analyze data. This diversity enables us to obtain a relatively comprehensive understanding of the design process. Drawing upon these studies and on those of others, we present below some key findings.

- The initial design phase is characterized by the creation of an information base.
- Engineers spend a considerable amount of time in seeking, organizing, modifying, and translating information relevant to their design work (which often transcends the engineer's personal discipline). While specific percentages might vary in different contexts, 75% appears to be a reasonable estimate (Engelmore and Tenenbaum, 1990).

- Design is a social and linguistic process requiring the participants to actively negotiate and translate information from one object world into other object worlds each being a composite based on the training, background, experiences (general and specific), etc. of each individual participant (Bucciarelli, 1984). There are difficulties in synthesizing and organizing diverse information into a coherent view.
- Due to the lack of adequate information integration, designers often evaluate only a single alternative.
- The organizational structure of the design team and the institution constrains information integration.
- The media used are inadequate to capture the required level of richness of the information.
- Even in the more analytical side of an engineer's work, the non-formal, non-analytic, tacit information about an analytic step is an important piece of the design information (Subrahmanian et al., 1993b). For our purposes, the significant thing about this is that even in the core of traditional engineering work, the role of translation, annotation, clarification, etc. is of central importance to the substance of an engineers task.
- Design history and rationale are continually being lost. This loss can result in the need to recreate the rationale of a design. This reverse engineering process can lead to repeating the same mistakes and failures encountered during the original design process. The central problem here is that the information required to learn from the past is either not captured or is so poorly organized and documented that its retrieval and value is compromised (Petroski, 1989). It is estimated that less than 20% of the intellectual capital of any firm is re-used.
- Design knowledge evolves since it is composed of a relatively stable core of knowledge surrounded by a much more unstable, rapidly changing periphery (which might later become part of the core).
- The relative size of the stable core with respect to the unstable periphery is a function of the maturity of the constituent disciplines.
- History maintenance for product classes plays an important role in an organization's ability to recoup on its investments in design knowledge.
- When the organization and/or the process is documented by the designers, it is often inaccurate and obsolete.
- The preliminary design phase is chaotic with the identification and definition of the required structures (design processes and organizations) being part of this phase. Engineers spend a significant part of their time coordinating, scheduling, inter-relating, and reconciling their work with others.
- There are multiple perspectives on and terminological differences in design information.
- Computational models and tools are distributed among different groups.
- The tools used impose limitations on effective collaboration.
- Design groups change over project lifetimes in structure and composition.
- There is, often, a mismatch between who has the information and who is assigned the specific design task.
- Communication characteristics (e.g., number of integration channels, communication infrastructure) has an impact on outcome.
- Functions of communication patterns (e.g., terminology used, volume of information exchanged) can be used as indicators of future design outcomes.

In summary, one cannot separate "pure" engineering work (in the sense of creating models, solving equations, etc.) from information management activities (IMA). Given the disproportionate time allocated to IMA in most engineering work, supporting IMA (computational or institutional) takes on considerable urgency. In order to understand what is entailed in providing such support, we can re-phrase the above findings at a higher level of abstraction: Engineers continually and collaboratively carry out their work by manipulating information required to solve the design problem at hand. It is also of considerable importance that engineers be able to build upon and draw from the collective knowledge of the organization thereby enabling its reuse and improving design performance (e.g. lower cost, less time, fewer errors, etc.). In our studies of the current procedures in engineering information management in several industrial organizations, we have discovered the following information integration activities and needs.

Information manipulation is characterized by three sets of activities. The first set is the creation, retrieval, classification, and evaluation of information. Supporting these activities requires functional support for creating, structuring, and finding information, and the use of standards. The second set is the transformation and translation of information across multiple representational structures. Supporting these activities requires functional support for sharing methods and tools, use of standards, integrating legacy methods and tools and external methods and tools, and the ability to evolve the system. The third set is the storage, access, and protection of information. Supporting these activities requires functional support for distributed storage and replication, access control, and security from external damage.

Knowledge building is characterized by two sets of activities. The first set is the capture and re-use of the design process and the design rationale. It requires support for capturing history,

capturing rationale, and structuring information. The second set of activities is the capture, consolidation, and re-use of knowledge (generated from the previous set of activities) by designers with different perspectives. Supporting these activities requires functional support for learning by induction, enabling end user customizing, and sharing information. Collaboration comprises the activities of negotiation and coordination that require support for sharing information, change management, and work flow and process tracking.

The Context of Engineering Work

From these observations and the published literature, we can characterize the context within which engineering work (including, of course, IMA) takes place and some of the issues that need to be addressed by support tools. In what follows, we describe several of these characteristics. An extended list with the consequences of creating design support systems can be found elsewhere (Reich et al., 1996b)

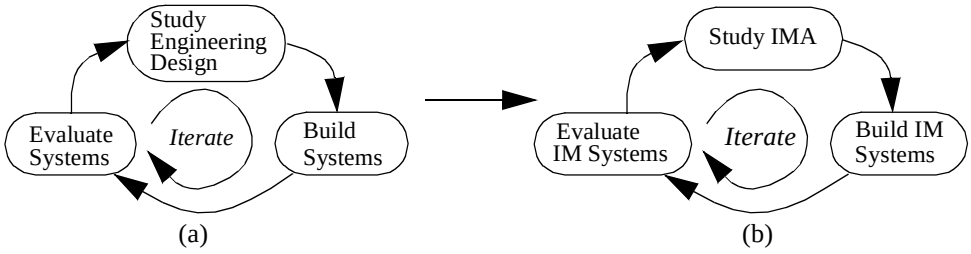
1. *Extended time*. Engineering activities extend over potentially long periods of time. The context of design must be maintained over that period and longer to allow for future reuse and for addressing life cycle issues.
2. *Multiple places*. Engineering activities take place in multiple locations which may change over time.
3. *Multiple cultures, practices, and behaviors*. Engineers participating in design projects come from different cultures. Organizations, through their development, evolve distinct cultures consisting of different practices, policies, and behaviors.
4. *Multiple languages*. People from the same discipline but from different organizational departments or divisions often use different languages or terminologies to describe disciplinary knowledge (Sargent et al., 1992). People themselves also use different languages (informal, e.g., text, images, audio, video; or formal, e.g., equations, 3D models) to refer to different perspectives of the same objects (Subrahmanian et al., 1993b).
5. *Multiple tools*. Some tasks, such as word processing, can be accomplished using different tools or methods. The use of different tools for the same tasks occurs in the same organization and certainly occurs in different organizations that work together. Moreover, existing organizations have significant investments in legacy tools that must be integrated into new computational environments.
6. *Multiple areas of expertise, disciplines, or tasks*. Engineering engages people with multiple areas of expertise in one discipline (vertical integration) as well as experts from multiple disciplines (horizontal integration) (Konda et al., 1992).
7. *Multiple perspectives*. People with the same area of expertise or from the same discipline may have different perspectives about a particular project if they assume different roles in the collaborative effort. One person can sometimes act as a customer and in other cases as a developer. Perspectives evolve or are determined in response to the context of a particular project.
8. *Interchangeable interaction methods*. A tool must support different anytime anyplace interaction methods in the same environment with the ability to switch back and forth between these methods.
9. *Usability and adaptability to workers with different levels of computer-literacy*. Of the tools desinged to support collaboration that are described in the literature, a large number are developed for use by experts who are proficient in the use of computers. More importantly, the people developing these tools may not appreciate the difficulties that regular users may have. In real engineering work, no assumption about the design participant's (customers as well as designers) computer proficiency can be made.

Based on these observations, we are led to the conclusion that much of the difficulty in doing design lies in acquiring, manipulating, transforming, using, and storing information in multiple and varied contexts in a manner suitable for subsequent re-use. These factors results in a situation characterized by a great deal of complexity and variety. As Ashby (1958) points out, a "control system" for such a situation, if it is to be adequate to the task, must exhibit at least as much complexity and variety. In the next section we explain how we approach the problem of providing support in the face of such complexity.

ADDRESSING INFORMATION MANAGEMENT

In order to manage the complexity of engineering design information, organizations have developed, adapted, and adopted a very wide variety of specific methods and tools so as to have the requisite variety necessary for effectively supporting design. By and large these are point tools; i.e., tools which solve well defined and circumscribed problems, often very effectively. Unfortunately such an agglomeration of point tools further compounds the complexity faced by the engineer since each such point tool requires its own sub-language and other arcana. This suggests that we develop an integrated support environment. However, a sufficiently rich integrated environment, unless carefully designed, could end up being as complicated (if not more so) to the engineer than the original problem. In order to deal with this dilemma we chose to build a support system on a foundation of a few well designed features which, when appropriately composed (in light of the existing information management problem in its context) can generate the desired variety in behavior. The strate-

Figure 1: Design support system development cycle



gy, then, is to carefully select features that are both simple to grasp (for the design engineer–the user, and the system designers–the developers) and yet can easily be put together to exhibit a very wide range of behaviors. From a different perspective, and generally because of the attendant complexity, it is almost impossible for any of us as support system builders to know enough of a specific design context to get the larger integrated system right–or even approximately right–the first time.

We are then faced with a fundamental dilemma: either develop good solutions to limited problems (in the sense of limited applicability, domain, or value) or develop comprehensive solutions that tend to be either unusable or just simply wrong. An alternative strategy would be to begin small and gradually build up the integrated system in a series of iterations. Additionally, while integrated environments cannot and will not evolve from point tools, they must be able to incorporate them. Based on our experience and understanding of engineering design, the role of the integrative tool is to provide bridges between the specific to the general, among disciplines, and functions, and to address the collection of information based activities as a whole.

Our approach is created to deal with these observations. We begin by assuming that we will fail in the first few rounds of development. Instead of trying to avoid such failures, we anticipate them, and indeed factor them into the development process in such a way as to rapidly converge to the larger, more reliable, and useful system. This convergence is achieved by the careful construction of basic building blocks which lead to a set of tools, methods, and code modules that exhibit the desired behavior: they are simple to put together, to comprehend, to use, and if necessary to throw away. For example, we have identified a canonical representation for information and knowledge which appears to be extremely general. Thus far, we have been able to represent all types of information and knowledge using this canonical representation.

Hence, while on the surface our iterative approach is not fundamentally different from other approaches in software engineering (Boehm, 1988), the guiding principles, the architecture, the tools and methods, are all internally consistent and designed to support the rapid development of a series of increasingly rich support systems which can then be followed by a hardening phase for final deployment. The basic features of our approach are:

- information flow studies (Finger et al., 1993; Subrahmanian et al., 1993a) which identify the specifics of the situation;
- user participation (Reich et al., 1996a) in as integrated a fashion as possible to engender the maximum possible communication bandwidth as well as legitimacy and buy-in;
- rapid prototyping (Dutoit et al., 1996; Reich et al., 1996b) using specially developed infrastructures and languages designed for the prototype as opposed to class-based development;
- field testing; and
- a distinct code hardening and maintenance step (which might be undertaken by another development group) (Dutoit et al., 1996).

The process we evolved is shown in Figure 1. In light of our experienced observation of design work, the general cycle shown in (a) is reinterpreted as shown in (b). We hasten to add that, in keeping with our general approach of tentativeness, this process is also being continuously refined to suit specific projects and we believe that such refinement will always take place. In order to execute these steps, we have identified five broad methods: (1) information flow-study, (2) user participation, (3) prototyping, (4) testing by users (uncontrolled study) industry/classroom, (5) code maintenance and hardening. The relations between the process steps and the methods is given in Table 2. Each method has to be realized by some infrastructure component or specific tools as shown in Table 3. In this paper, we focus on the development of the infrastructure (columns 2 and 3 of Table 3). The other aspects are discussed elsewhere (e.g., Subrahmanian, 1992; Dutoit, 1996).

N-DIM: AN INFRASTRUCTURE FOR INFORMATION MODELING AND APPLICATIONS

The basic premise of the *n*-dim system is that every member in the product design team operates in an information space, called a *workspace*, that is characterized by the domain of experience and skill of the participant (Levy et al., 1993). The information space of the product is characterized by the union of the information spaces of the individual participants. (This allows us to address the issues associated with multiple locations, languages, areas of expertise, and perspectives of the design participants.)

Table 2: Objectives/services and methods used to attain them

	1	*2*	*3*	*4*	*5*
Methods *Process Steps*	*Information flow-study*	*User participation*	*Prototyping*	*Testing by users (uncontrolled study) Industry/Classroom*	*Code maintenance and "hardening"*
Understanding of the current state of information management	An information map of the Business division studied	Identification of a specific target area for support			
Development of support systems		Use and system specification document	A series of working prototypes	Areas of improvement of use and performance before testing	Improving scope. quality, performance, and usability
Assessment of support system effectiveness	An information map after system installation	Continuous feedback.			
Evolution of system		Continuous identification of new needs	Continuous evolution	Identification of needs (research and improvement) to reduce effort and time	

This union of information, the product (or organization) information space, is not a straightforward union as there are terminological inconsistencies across the information spaces and well understood and not so well understood relations between the elements of the information space. Further, in each information space of the participants and in the product information space, the organization of information itself evolves as process and product understanding increase to form a shared memory (Konda et al., 1992). The objective is to support the individual evolution of knowledge and the collective evolution of knowledge in the form of information structures that are constructed by the participants in the course of the product development process. The history of both process and product is critical to ensuring that evolution takes place in an effective manner. This is important both to the short term evolution of a project and to a long term evolution of policies of operation. To address this, we have taken as our hypothesis that a generalized graph modeling environment that operates over the elements (other information structures– graphs and atomic information elements) in the information spaces is necessary to capture the structure and evolution of information and knowledge, both formal and informal and individual and group. We hypothesize that this generalized graph is a canonical representation from which all others can be derived.

Concepts in *n*-dim

Information Objects: Information objects are of two types: atomic objects and structured objects. Atomic objects are strings, numbers, images, audio fragments, etc. They are not decomposable. Structured objects are graphs whose nodes are atomic objects or other structured objects. The graph includes named links that can exist between any two nodes.

Models: For convenience we use the term model to denote both atomic and structured objects. Objects are referenced in a model rather than being embedded in a model. Models imply object association by having their pointers collected together. Named links are used to describe the relationships between the object pointers.

Flat space: Flat space is a term we have given to the conceptualization of an information space where any model is directly referable. This allows for the creation of a user defined set of relationships across information objects of any granularity. Users have the ability to create any arbitrary model over a subset of the entire collection of information objects in the information space.

Modeling languages

A model can be abstracted to create a set of building blocks that correspond to the type of information objects in the graph and the types of named links in the graph. These abstractions can be made to create a vocabulary which can, in turn, be used to create other model instances. For example, one can create an object and abstract the features of that object in creating another object of different dimensions, scale, etc. Here, one has developed a language for describing that particular artifact. Languages restrict the type of objects and named links users may use to construct further instances of the model type. Modeling languages are

Table 3: Methods, Tools, and Outcomes

	1	*2*	*3*	*4*
Tools *Methods*	*Questionnaire and interviews*	*Infrastructure for evolving information systems*	*Layered modular architecture*	*Social Science methods (regression/multiple regression/ natural language analysis)*
Information flow-study	Identifying communication gaps			
User participation				Source of action research methodology
Prototyping		Support for quick prototyping, customization, legacy tool integration and evolving the infrastructure	Potential re-use of existing legacy layers (e.g., DB)	
Testing by users (not controlled study) Industry/ Classroom		High usability to support early testing		Identification of needs (research and improvement.) to reduce effort and time
Code maintenance and "hardening"		Support for improving performance of validated code	Supports improving layers with new technologies	
Basic research (e.g., study the role of Communication in design projects)				Identification of needs (research and improvement.) to reduce effort and time

models; therefore, any model can be used to define the grammar of other models.

Such a grammar defines what is a correct instance of a model (its semantics) in a modeling language. Additionally, we can increase the power of this approach by attaching behavior to a model using what we call operations. In essence, operations are pieces of code which, when executed with the relevant parameters, allow a model to automatically perform actions on behalf of the user (or the modeling language designer). For example, an operation on a model might be used to inform the user when someone adds a part to that model. Symmetric to the semantics behavior outlined above, operations are inherited by model instances created by using the model to which those operations are attached as the modeling language. Thus, the system allows for standardization of modeling languages and their use and for the evolution of new graph types from the model instances. As a result, the system supports both deductive and inductive approaches to the modeling process.

As more modeling languages and operations are developed, they start to form repositories whose items can be reused for creating new languages or applications or adapting old ones. We have built the infrastructure so that it will support the flexible creation of such repositories and their effective reuse.

Evolution: Private, Public, and Published

History is critical to effective evolution and ordered evolution is essential to recording history. We have developed an ordered evolution of the system with the following three facilities. These facilities deal with different levels of granularity: private, public, and published.

Private: Private, as the name denotes, is the private information space of the individual. There are no restrictions on how a private space is managed. The users can add, delete, and restructure their information objects.

Public: This mode of operation is a public forum area. Here the primary objective is to provide the ability to all participants to share and add to the model, both synchronously and asynchronously. As with any forum, the language of the forum is restricted to the purpose and domain of discourse as determined by the participants or the existing body of knowledge. History can be recovered by viewing a model's state in time.

Published: The published mode of operation is an archival facility. Any information object that is entered into the published information space cannot be withdrawn (i.e., it is persistent). Changes are published by copying, modifying and then re-publishing a model. The system automatically records the act of

copying and re-publishing, thereby keeping a branched (time and owner) history of the model. The model that allows for the tracing of the origin of the document is itself a graph within the system.

In addition to the need to record history, the need to search for information and effectively visualize information in different ways is equally important. As more information is created in *n*-dim, knowledge could be organized in repositories that ease the location and reuse of relevant knowledge.

The above characterization of the system is necessarily abstract, as the details of the system cannot be described in this limited space.

Strength and weaknesses of *n*-dim

The primary strength of the system is its approach to dealing with software development and knowledge development in an evolutionary manner. The system combines evolution, history, and modeling within the same framework–the framework of graph based modeling. The other main strength of the system is its flexibility in allowing the easy integration of legacy tools, they can be invoked from within the system in their native form or can be integrated fully into the system. Further, the system also allows for the creation of new tools by the user as needed (Dutoit et al., 1996). For example, we are integrating a Natural Language Processing (NLP) tool to allow us to handle terminological differences in design contexts. We are also expanding our research efforts in creating a graphically based end-user scripting language capability to make the above tasks easier.

Another strength of our system is the infrastructure upon which it is built. The flexibility of the object tool kit allows for extensions to the system incrementally without damaging the underlying system (Dutoit et al., 1996). This problem is acute in many commercial systems, where moving from one version to another version often requires a transition time which may last from hours to weeks.

The *n*-dim system itself is an infrastructure that is customized to particular applications and within which new applications can be built. For example, we have developed several types of issue-based discussion applications and tested them (e.g., IWEB, Coyne et al., 1994). *n*-dim is not a system that can just be bought and installed. This can be viewed as a weakness from a commercial point of view and we are keeping that much in mind as we plan for commercialization. But a flexible infrastructure with the strong capabilities of *n*-dim including its quick prototyping and code hardening capabilities is potentially a great strength for any organization that chooses to make the investment.

HOW *N*-DIM ADDRESSES A VARIETY OF INFORMATION ACTIVITIES

We have developed the *n*-dim infrastructure based on a small set of features we have identified in addition to the graph-based canonical representation of information described in the previous section. We have also developed some applications using the infrastructure. In order to ensure that the goal of the information infrastructure conforms to the needs of the design context, we have developed a table of influences (Table 4) to provide an understanding of how features and applications in the *n*-dim system are developed with reference to their impact on the dimensions of complexity of design contexts. As contexts are studied and applications are developed, a cycle of hypothesizing and evaluating the impact of the applications on the dimensions of the design context occurs. This cycle enables us to perform a continual refinement of the core set of features that constitute the integrative environment.

We have created Table 5 for information management activities and their support with respect to *n*-dim features and applications. The purpose of the table is to provide a check list to ensure that the scope of the evaluation of the impact of features and applications covers individual information management activities. As mentioned earlier, the development of an information system requires the search for a minimal set of features and applications that will allow for the matching of the needs and requisite variety demanded by the context. Thus, it is important that we use a check list of factors such as the dimensions of the design and the dimensions of the information management activities in understanding the implications of any feature and application added to the system.

Tables 4 and 5 illustrate the endeavor of designing information management systems as a design problem where the impact of several interacting factors are unknown in specifying the correct design. They serve as drivers for creating and testing hypotheses about the utility of particular features and applications in an integrative environment. By using this iterative and evolutionary approach we believe an integrated information management for design can be created to match the complexity and variety exhibited by a design context.

To illustrate this process, consider the example of NLP tools in *n*-dim. We made the hypothesis that variations in the terminology used by designers could be exploited to understand the design process better. For instance, designers using a large number of terms at the onset of integration could indicate that numerous concepts are being discovered and reconciled. This high rate of discovery so late in the process could be caused by the failure of designers to communicate effectively before the integration phase.

Table 4: *n*-dim features addressing design context dimensions

Features and Applications of n-dim *Dimensions of design context*	*General Graph*	*Flat Space*	*Modeling Languages*	*Publication*	*Tool Encapsulation*	*Workspaces*	*Scripting Language*	*Search*	*Repositories*	*Issue-Based Discussion*	*NLP Tools*
Time				+				+		+	
Place				+				+		+	
Culture	+		+			+					+
Languages	+	+	+			+	+				+
Tools			+		+		+		+		
Expertise	+		+		+					+	+
Perspectives	+	+	+			+				+	+
Interaction					+		+		+	+	
Usability	+		+		+		+		+		

Table 5: *n*-dim features addressing IMA

Information Manipulation	*Knowledge Building*	*Collaboration*	*Features and Applications of n-dim* *Information Management Activities*	*General Graph*	*Flat Space*	*Modeling Languages*	*Publication*	*Tool Encapsulation*	*Workspaces*	*Scripting Language*	*Search*	*Repositories*	*Issue Based Discussion*	*NLP Tools*
+			Creating Information	+	+	+			+			+		+
+	+		Structuring Information	+	+	+			+			+		+
+			Finding Information								+	+		+
	+	+	Sharing information				+				+	+	+	
+			Information Visualization			+		+			+			
+		+	Using Standards	+				+		+				
+		+	Sharing Tools					+		+		+		
+			Integrating Legacy Tools					+		+				
	+	+	Capturing History			+	+					+		
	+	+	Capturing Rationale			+	+					+		
	+		Learning by Induction	+	+	+		+			+	+		

To test this hypothesis, we studied a number of software projects that relied on electronic means of communication (e.g., electronic mail, newsgroups) (Bruegge and Dutoit 1997; Dutoit, 1996). We used NLP tools to extract noun phrases from the electronic messages and developed a statistical model to analyze the factors that influenced their variations[2]. It was found, for example, that delayed negotiation of terms between design teams was indicative of future problems at the integration phase. More generally, we found that communication metrics can be used as indicators of problem areas and potential downstream risks to the design project. Based on this study, we are currently deriving a basic set of analysis and diagnostic tools that can become part of the support environment and, if desired, used by designers to forewarn them. It is from this experience that the "+" sign of the NLP negotiation cell in Table 5 was obtained.

As we learn more from the empirical study of design, the contents of these tables will evolve. Entire rows (or columns) may be consolidated, deleted, or created as technologies, work processes, knowledge, and organizational culture change. On a smaller scale, as our knowledge grows, the entries in each cell could change (from a "+" to a blank or vice versa). Perhaps of greater value, the tables can be used as guides in selecting specific studies or implementations as indicated by blank cells, rows, or columns.

SUMMARY

In this paper, we have outlined an approach to creating design support systems that is based on observations of design practice. The approach is an iterative process composed of data-driven hypothesizing and creating, testing, and evaluating support systems in the design context to understand the impacts they have on information management activities. In developing our methods, we work with an organization as partners to build and maintain support systems for knowledge capture, dissemination, and maintenance within the firm. In these partnerships the client provides the context, methods, and tools for doing design, we provide our tools and methods for developing support systems, and as a joint team we develop the system. This team develops a prototype support system with the user and tests the system for effectiveness. If during development we find there are needs that cannot be fulfilled by current technologies or we need methods to understand information flow dynamics in a group, then we look for them in other disciplines or develop them as part of our basic research. The desired outcome is that we walk away with a deeper understanding of group design and management of knowledge in organizations and that our partner has a system for knowledge capture, dissemination, and maintenance that improves their design performance.

2. This is an example of the use of social science approaches shown in Table 3.

ACKNOWLEDGMENTS

This work has been supported by the Engineering Design Research Center, an NSF Engineering Research Center.Support has also been provided in part by Asea Brown Boveri Ltd. and Union Switch and Signal, Inc.

BIBLIOGRAPHY

Ashby, R. (1958). Requisite variety and its implications for the control of complex systems. *Cybernetica*, 1(2):1-17.

Boehm, B. W. (1988). A spiral model of software development and enhancement. *IEEE Computer*, May:61-72.

Bruegge, B. and Dutoit, A.H. (1997). Communication Metrics for Software Development. In *Proceedings of the 19th International Conference on Software Engineering*, pages 271-81, Boston, Massachusetts.

Bucciarelli, L. L. (1984). Reflective practice in engineering design. *Design Studies*, 5(3):185-190.

Clark, K. and Fujimoto, T. (1991). Product Development Performance. *Harvard Business Press*, Cambridge, MA.

Coyne, R., Dutoit, A., Uzmack, J., and O'Toole, K. (1994). *IWEB (Information WEB): Information management for software*. Technical Report EDRC-05-87-94, Engineering Design Research Center, Carnegie Mellon University, Pittsburgh, PA.

Dutoit, A. (1996). *The Rôle of Communication in Team-Based Software Engineering Projects*. Ph.D. thesis, Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA.

Dutoit, A., Levy, S. N., Cunningham, D., and Patrick, R. (1996). The Basic Object System: Supporting a spectrum from prototypes to hardened code. In *Proceedings of OOPSLA '96*, pages 104-121, New York, NY. ACM.

Engelmore, R. S. and Tenenbaum, J. M. (1990). *The engineer's associate: ISAT summer study report*. Unpublished.

Finger, S., Subrahmanian, E., and Gardner, E. (1993). A case study in concurrent engineering for transformer design. In Rosenburg, N. F. M., editor, *Proceedings of ICED-93 (The Hague)*, pages 1433-1440, Zurich. Heurista.

Hales, S. (1987). *Analysis of The Engineering Design Process in an Industrial Context*. PhD thesis, Department of Engineering, University of Cambridge, Cambridge, UK.

Konda, S., Monarch, I., Sargent, P., and Subrahmanian, E. (1992). Shared memory in design: A unifying theme for research and practice. *Research in Engineering Design*, 4(1):23-42.

Kuffner, T. A. and Ullman, D. G. (1991). The information requests of mechanical design engineers. *Design Studies*, 12(1):42-50.

Leifer, L. J. (1991). Instrumenting the design process. In *Proceedings of ICED-91* (Zurich).

Levy, S., Subrahmanian, E., Konda, S. L., Coyne, R. F., Westerberg, A. W., and Reich, Y. (1993). *An overview of the n-dim environment*. Technical Report EDRC-05-65-93, Engineering Design Research Center, Carnegie Mellon University, Pittsburgh, PA.

Petroski, H. (1989). Failure as a unifying theme in design. *Design Studies*, 10(4):214-218.

Reich, Y., Konda, S. L., Levy, S. N., Monarch, I. A., and Subrahmanian, E. (1996a). Varieties and issues of participation and design. *Design Studies*, 17(2):165-180.

Reich, Y., Konda, S., Subrahmanian, E., Cunningham, D., Dutoit, A., Patrick, R., Westerberg, A., and the *n*-dim group (1996b). Being agile when building information infrastructures for agile enterprises. Submitted for publication.

Sargent, P., Subrahmanian, E., Downs, M., Greene, R., and Rishel, D. (1992). Materials' information and conceptual data modeling. In Barry, T. I. and Reynard, K. W., editors, *Computerization and Networking of Materials Databases: Third Volume*, ASTM STP 1140. American Society For Testing and Materials.

Subrahmanian, E. (1992). Notes on empirical studies of engineering tasks and environments, invited position paper. In *NSF Workshop on Information Capture and Access in Engineering Design Environments (Ithaca, NY)*, pages 567-578.

Subrahmanian, E., Daleng, E., and Maeland, A. (1993a). *Information flow analysis of hydro power design*. ABB CR Report, ABB NOCRC, Billigstad, NO.

Subrahmanian, E., Konda, S. L., Levy, S. N., Reich, Y., Westerberg, A. W., and Monarch, I. A. (1993b). Equations aren't enough: Informal modeling in design. *Artificial Intelligence in Engineering Design, Analysis, and Manufacturing*, 7(4):257-274.

Tang, J. (1989). *Listing, drawing, and gesturing in design*. Technical Report SSL-89-3, Xerox Palo Alto Research Center, Palo Alto, CA.

Wilkins, D. J., Henshaw, J. M., Munson-Mcgee, S. H., Solberg, J. J., Heim, J. A., Moore, J., Westerberg, A., Subrahmanian, E., Gursoz, L., Miller, R. A., and Glozer, G. (1989). CINERG: A design discovery experiment. In *NSF Engineering Research Conference*, pages 161-182, Amherst, MA. College of Engineering, University of Massachusetts.