AI-supported Quality Function Deployment

YORAM REICH*

Key Words—quality function deployment, artificial intelligence, knowledge acquisition, quality control, design rationale, repertory grids, conceptual structure, enterprise knowledge integration, graphs.

In the *Proceedings of The Fourth International Workshop on Artificial Intelligence in Economics & Management*

Abstract— Manual Quality Function Deployment (QFD) tools are limited in their use and their reuse. Computational tools can alleviate these limitations. In addition, Artificial Intelligence (AI) tools can further enhance the functionality of QFD tools. A graph-based information representation is proposed as the basis for integrating various QFD and AI tools. An architecture of a computational QFD (CQFD) tool based on the graph-based modeling environment *n*-dim is briefly discussed. The ideas are illustrated through the design of a cork remover.

1 INTRODUCTION

The development of any product involves projecting its potential success in achieving its functional and commercial goals. Better quality designs that match customer needs and preferences and integrate manufacturing and other life-cycle issues early on into the design process are more likely to be competitive. Thus, there is significant concern in industry about quality product design that is addressed, among others, by Quality Function Deployment (QFD) tools including the House of Quality (HoQ) (Akao, 1990; Claussing, 1993; King, 1989) and the seven (new) management tools (Mizuno, 1988). The success of QFD tools is detailed in many recent publications (Akao, 1990; Claussing, 1993; King, 1989).

QFD tools are simple to use; their results are displayed in graphical models that are easy to comprehend; they are used as manual tools; and they have proved successful in diverse situations. One can argue that simplicity and manual execution were key factors that enabled the widespread use of these tools in organizations, leading subsequently to better product design. There are also Artificial Intelligence (AI) techniques such as rule-based expert systems or constraints solvers that have been used to implement tools to solve a variety of problems, such as design-for-X or concurrent engineering. These tools seek to address the same issues that QFD tools address, but in contrast to QFD tools, AI tools are complex to use and hard to comprehend. Furthermore, while there are many practically useful operational AI systems, AI techniques cannot claim practical success comparable to that of QFD tools.

Despite their simplicity and practical success, QFD tools have limitations that can be classified into four categories:

1) The *effort* category deals with the ease of handling a QFD tool. It is difficult to handle large amount of data on a paper. Users of existing tools attempt to cope with this limitation by *aggregating issues* so that their number is limited. Also, paper "suffers" any information written. It is the task of its users to make sure that the information is *correct and consistent*. Similarly, the users are in charge of all the simple *calculations* involved in using paper methods.

2) The *expressiveness* category deals with the nature of information that a QFD tool accepts. QFD tools are limited in the types of information that they can code. For example, the HoQ permits only several quantitative (or qualitative) values[1] to be used as evaluations or correlations. However, in some situations, richer and more *precise correlations* can be established based on parametric analysis or physics laws. Also, if we look at QFD graphical models, there is significant amount of information that is not recoverable from them. For example, since the data in many of the tools is a result of group processes, the individual positions are missing and *the reasons* that led to existing choices are missing too. It is desirable to adapt a tool to make use of such information when it become available.

3) The *adaptability* category deals with the ease of modifying a tool to suite diverse needs. This is critical considering that Japanese have used more than 80 types of QFD matrices (King, 1989). These adaptations are difficult to perform with paper methods.

4) The *communication* category deals with the modes in which QFD tools can be used. The use of manual QFD tools requires having good face-to-face (synchronous) communication practices because they are effectively used in group settings. In such settings it may not be easy to *distinguish*

*Department of Solid Mechanics, Materials, and Structures, Tel Aviv University, Ramat Aviv 69978, Israel, email: yoram@eng.tau.ac.il

[1] Some of the procedures discussed below assume that data items in the HoQ represent different types of scales, e.g., ordinal or ratio. A treatment of this is the subject of another paper but beyond the scope of this paper.

between different but seemingly similar positions or to *observe the similarities* between other positions thus some design information may be unnoticed or lost. The ability of QFD tools to capture and communicate design rationale is limited when used in an asynchronous manner, whether for communication among members of the current project or for disseminating information into the organization for future reuse.

It is clear that computer tools can alleviate these four limitations (Reich, 1995). The key issue is how to integrate the tools such that they maintain the usefulness of the original manual QFD tools. To date, there has been little work done on supporting QFD with techniques beyond spreadsheet (Reich, 1995; Pohl and Jacobs, 1994). This paper focuses on the benefits that AI technology can offer to QFD tools in the processes of information acquisition, (re)use, and communication. For the sake of simplicity, we will demonstrate how one version of the HoQ can benefit from AI tools using a design of a cork remover (i.e., wine bottle opener).

2 INFORMATION ACQUISITION

The first stage of creating a HoQ involves acquiring information about the design including customer requirements (room 1 in Figure 1), engineering characteristics (room 3), and existing related products and their rankings along the customer requirements (room 2) and the engineering characteristics (room 5). In parallel, correlations between engineering characteristics and customer requirements (room 4) and the tradeoffs between engineering characteristics (room 5) are elicited. Subsequently, the information in the HoQ is used to calculate new product goals (room 7).

This stage involves information collection from diverse sources such as customer interviews and other marketing surveys. It can be assisted by the use of natural language processing (NLP) tools that can uncover concepts and their relationships by analyzing text. Conceptual differences discovered by analyzing different sources can be inspected to establish a common terminology (Reich et al, 1993).

Gradually, common terms and their relationships emerge. They can be represented in graphical forms as seen in Figure 2. Further analysis of the problem can lead to revealing simple structures underlying the problem. Function and structure diagrams of the problem and existing products can emerge and mapped to one another as seen in Figure 3. These figures are representations of 2 QFD tools: the relation and the systematic diagrams.

AI can further support information acquisition through the use of grid-based knowledge acquisition tools such as KSS0 (Gaines and Shaw, 1993a). Grid-based tools can focus customers and engi-

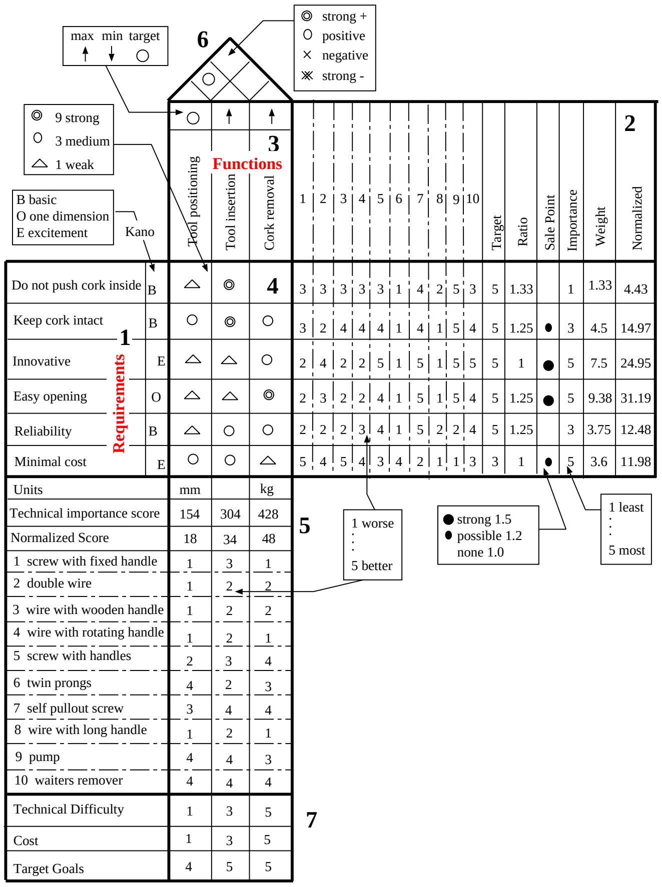

Figure 1: The House of Quality

neers on products similar to the present design and on their different properties. Tools such as KSS0 provide on-line assistance to their users by continuously analyzing the data accumulated. Thus, if two cork removers seem similar from their rankings on all product properties KSS0 will request a property that differentiates between the two. For example, at some intermediate stage products 3 and 4 rank the same on the first 4 customer requirements leading to eliciting the reliability property. Also, if all products are ranked similarly on two properties, KSS0 will request to recall another cork remover that ranks differently on both. This process can elicit many products with different properties and different operating principles such as those based on a screw, pressure, or twin prong mechanisms; thus, grid-based methods can assist in completing rooms 1, 2, 3, 5 of the HoQ. At a more detailed level, grid-based methods can be used to acquire information about classes of removers, such as screw-based removers. In such a process, different properties including the pitch of a screw or the precision of its manufacture can be elicited by inspecting similar screw-based cork removers. These properties can augment the relation or systematic diagrams.

Grid-based methods are often used by individuals (or by groups, in the same way as QFD

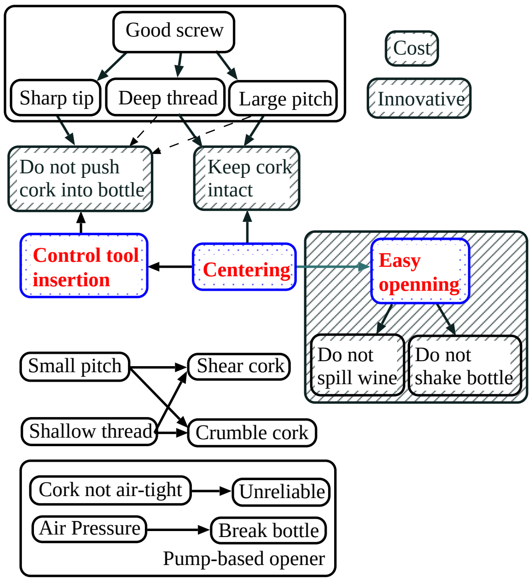

Figure 2: Relation diagram

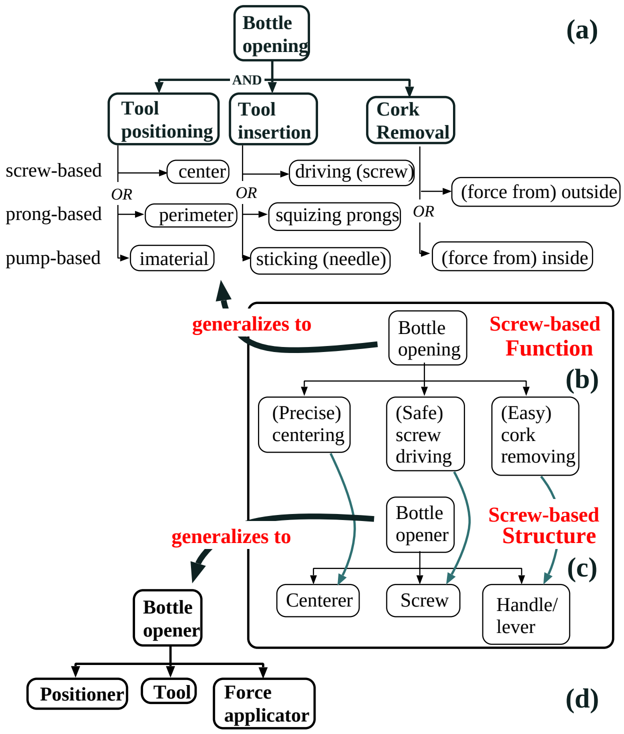

Figure 3: Systematic diagram

tools are). Nevertheless, the information in several grids acquired from several individuals can be analyzed and compared to result in the consensus, conflict, correspondence, or contrast between individuals (Gaines and Shaw, 1989). This process has an effect similar to brainstorming with analytical means for providing guidance and feedback. Subsequent to the comparison of multiple grids, customers and designers can revise and enhance their grid information. The analysis and revision processes can iterate, leading to an information structure that is comprehensive and its evolution documented. The HoQ in Figure 1 can be perceived as the final product of several such iterations, including information from the diagrams (Figures 2 and 3). In particular, the engineering characteristics converged to the three sub-functions that appear in the systematic diagram and the customer requirements reflect concepts from the relation diagram.

Grid-based methods such as KSS0 include various grid-data analyses such as clustering or entailment (Gaines and Shaw, 1993a). Clustering can reveal that some properties can be grouped to more abstract properties thus supporting the aggregation of properties. In addition, if a customer requirement is clustered with an engineering characteristic a correlation between them can be established for use in room 4, or if two engineering characteristics are correlated a tradeoffs between them can be revealed.

Finally, an entailment between engineering characteristics or between an engineering characteristics and a customer requirement suggests the existence of a tradeoff (in room 6) or a correlation (in room 4), respectively. For example, from the rankings in room 5, tool positioning entails tool insertion thus leading to establishing a positive tradeoff between them. The establishment of these relationships can benefit from background knowledge previously elicited for different purposes. Given that such knowledge is effectively coded and can be easily inspected, it can assist the present design; we return to this point later.

The ranking by customers and engineers is error prone. Inconsistent rankings can emerge easily since related rankings are done in isolation. For example, while easy opening and cork removal are correlated strongly, the ranking of products 6 and 9 on these properties differ substantially (3 vs 1 and 3 vs 5). A heuristic checking mechanism can be coded to check such differences, but in addition, correlations in room 4 that differ significantly from entailments calculated from other rankings can be marked for inspection.

3 INFORMATION USE

After acquiring and organizing the information in a HoQ, it can be used to set goals for the new design. The goals can be derived from an existing design (e.g., product 5) by identifying the improvement over it or by directly establishing the requirements for the new design. In the former case, we have a new and a base design. In the latter case we can identify one or two old designs that are close to the new one. This identification can be assisted by a case-based reasoning (CBR) tool. In our design, given that we have a limited set of previous designs we only need the CBR tool to focus us on those products closely related to the present design based on the requirements and its anticipated structure or style—a minimal task for CBR. In addition, CBR can be employed in later design stages or in more complex designs. For example, CBR can identify parts of HoQs from previous designs that can be used to address parts of a new product design. This can be highly useful when dealing with highly customized products or when developing similar products for different markets that employ different design standards.

After product goals are determined, product design progresses through the selection of a design concept, its potential decomposition, its embodiment by selecting from libraries means for implementing functions into structures including exploiting opportunities for function sharing. Some of these processes can be assisted by QFD tools and others may benefit from various AI techniques. Again, the critical issue is integrating the tools such that they assist in the usual engineering work.

4 INFORMATION COMMUNICATION

In the information acquisition and analysis as well as in the usage stages many decisions are made ranging for ranking a product, to establishing a common term, to using a particular analytical tool for establishing tradeoffs between engineering characteristics. These decisions processes are presently lost. Computational tools provide opportunities to structure and maintain these processes and their outcomes. Each decision can be augmented by an IBIS-like (Rittel and Webber, 1973) information structure that maintains the various choices, the arguments for and against each choice, and the resolution (Reich, 1995). Figure 4 displays the definition of such IBIS models. Such an information structure allows to exercise asynchronous communication modes for making decisions (Subrahmanian et al, 1993a) and provides better grounds for understanding decisions at later design stages or in future projects.

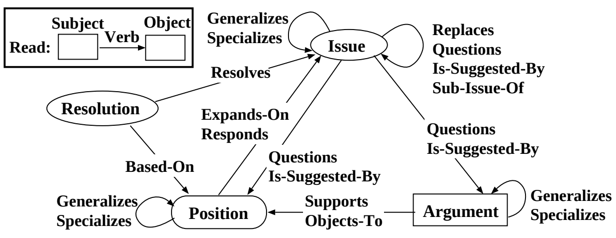

Figure 4: Definition of IBIS models

Figure 5 depicts partial IBIS models explaining the assignments of a tradeoff and a correlation. In each case, the corresponding issue is the HoQ value and the possible positions are the possible values for this HoQ entry. Arguments can support or object to particular positions. The IBIS model of the correlation (5c) depicts an elaborate argument that includes parts of the relation diagram (Figure 2) and supporting evidence from a complex finite-element analysis procedure of a screw insertion into a cork. This argument is only valid for a screw-based remover. If a new remover is developed using another operating principle, this argument is inapplicable and the support for the strong correlation is retracted. Other correlations can become valid or the user of the HoQ is required to fill in the missing data.

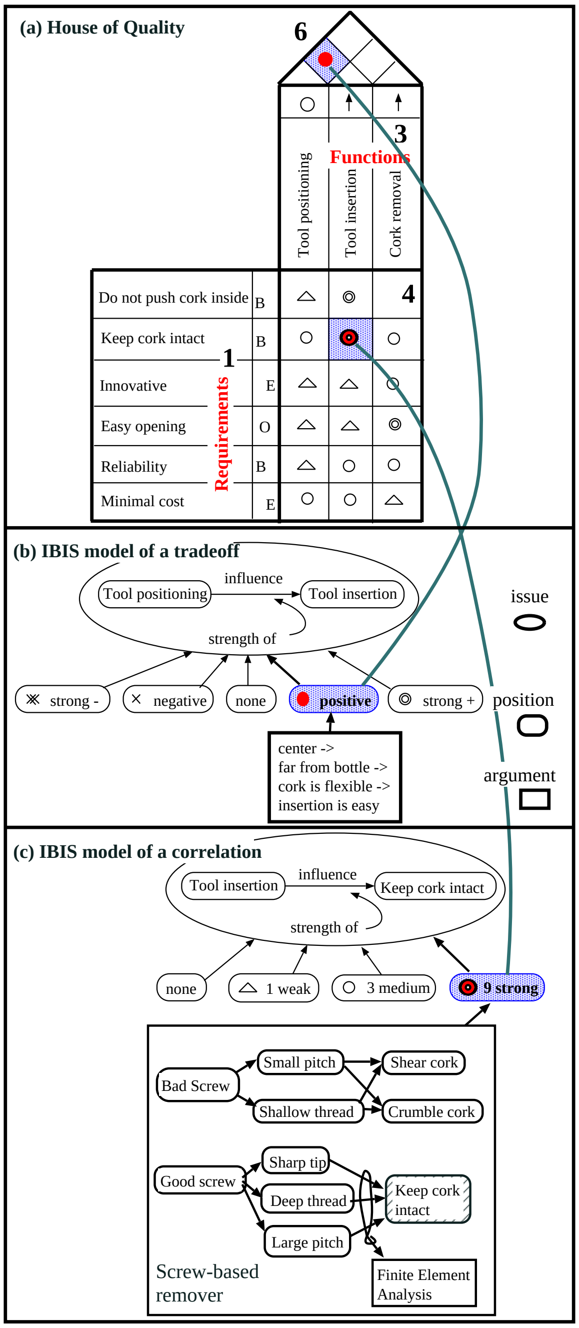

Figure 5: HoQ and IBIS models

5 THE ROLE OF GRAPH MODELS

Graph-based models are powerful enough to model diverse types of information and in many engineering applications are the choice of modeling language. For example, *conceptual structures* (Sowa, 1984) are a graph-based system for representing concepts and relations that is general as predicate calculus. There have been studies on the use of conceptual structures for various CAD-related topics such as: mapping the enterprise information language EXPRESS (Wermelinger and Bejan, 1993) and modeling the knowledge interchange format KIF (Sowa, 1993). Since conceptual structures cannot represent uncertainties well, others graph-based models called *influence diagrams* (Howard and Matheson, 1983) can be used to represent probabilistic knowledge for probabilistic inference. Influence diagrams are extensions of Bayesian networks and have been used in artificial intelligence, decision analysis and statistics. Figure 6 depicts an approximate hierarchy of some graph-based models.

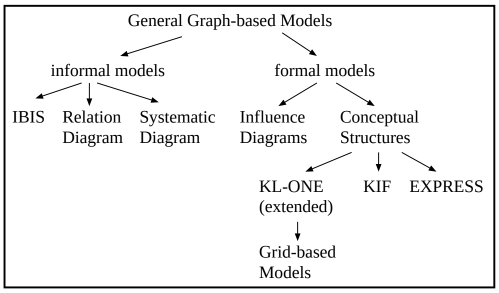

Figure 6: Relations between various graph models

By building a computational support tool for QFD on the basis of a general graph-based modeling system such as *n*-dim, the system discussed in Section 6, several benefits become evident. First, it is easy to implement support for all the graph-based QFD tools we have encountered in Figures 2, 3, and 5. Furthermore, the HoQ (or grid data) can also be represented easily as a graph. Second, such a tool will be able to handle the variety of other graph-based modeling languages and integrate them with QFD models.

The process of using a graph-based modeling tool can progress as follows. During the information acquisition, informal models such as the relation or the systematic diagrams are created. At the same time, grid-based methods (which can be derived from languages similar to KL-ONE (Gaines and Shaw, 1993b)) are used. These models can share nodes and links. For example, the systematic diagram and the HoQ share the three sub-functions depicted in Figure 3(a), the relation diagram and the IBIS model in Figure 5(c) share a set of models and their relationships, and the HoQ and the relation diagram share some of the customer requirement concepts. This "sharing" means that the models point to the *same* object in the collection of all objects. Also, a model is merely a subset of the objects with some of their relationships, and multiple models can be created with the same set of objects. We refer to the set of objects as the general graph or the *flat space of objects*.

As the information acquisition process progresses, additional models are created, and occasionally, more formal graph-based models such as conceptual structures are employed to represent the information. In some cases, in order to create these formal models, a mapping from the informal to the formal models is created.

After enough information from many projects is accumulated in models, the sequence of building and using models using QFD and AI tools changes due to interactions between the tools or models. For example, information coded in the general graph can be used to rank products while using grid-based tools and the graph can be further augmented with new informal parts while using a grid-based tool. Opportunities for models reuse demonstrate the significant benefits of this approach for capturing enterprise information. These opportunities can benefit from CBR tools that detect parts of previously developed models that are applicable to the present product design.

6 IMPLEMENTATION

n-dim (Levy et al, 1993) is a system developed to support collaborative design and is based on empirical studies of designers revealing that design is a continual negotiation of constraints, terminology and trade-offs for the creation of a shared understanding of the design process and product (Konda et al, 1992; Subrahmanian et al, 1993a). Designers exchange information expressed in different representational forms (e.g., pictures, text), in different media (e.g., paper, electronic), and in different modes (e.g., formal, informal). In manipulating design information designers use a variety of models (Subrahmanian et al, 1993b). *n*-dim is implemented under the assumption that a general graph-based modeling environment can capture significant portion of the models used by designers, where models are collections of objects and relations. Thus, the set of models is a set of multiple potentially overlapping classifications over the universe of objects.

Flat Space of objects and models. An *n*-dim object can be anything that can be stored electronically. The world according to *n*-dim is conceptually flat: objects do not contain other objects. There are two categories of objects in *n*-dim: atomic and structured where the latter are built out of other atomic or structured objects. An atomic object can be anything electronically storable such as text, image, or video. A model is the primary type of structured object: a collection of objects and their *links* or relationships, which are themselves objects (see Figures 2, 3, and 5(b,c)).

Once constructed, models in n-dim become regular objects; thus, the same object or model may participate in many other models. The simplicity and generality of model structures enable the capture of rich context of a given object.

Roles of a model. Models play two primary roles in n-dim: instance/prototype and language. In its role as a prototype, a model may be copied to serve as initial conceptualization for new models. For example, Figure 3(b) is a prototype model constructed for representing the function of screw-based removers that evolved into a more general model for representing the function of any remover in 3(a). In addition, every model can be viewed as representing a *class* of models in a generative sense; that is, the set of links and objects used in the model become the vocabulary, and the (embedded) rules of composition become the syntax and scope of semantics for building other models. In this sense, a model serves as a language. All objects refer to another model as their *modeling language*, and are said to be *in* that language. The only kinds of objects and links that can be put in a model are those mentioned in its language, and only legal compositions of these objects and links can be created. A model viewed as a modeling language can be thought of as a grammar. For example, Figure 4 depicts the definition of the IBIS modeling language and Figure 7 depicts the modeling language of systematic diagrams of three types: functional, structural and behavioral. In the latter, models constructed with this language can be entities where an entity is either function, structure, or behavior, and links can be sub links. The first type of concept used in a model will determine whether it is a function, structure, or behavior systematic diagram.

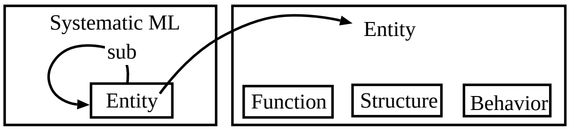

Figure 7: A modeling language for systematic diagrams

Clearly modeling languages can be created for generating any syntactically correct graph-based model including the models of QFD tools. The conceptual separation between models' structure and presentation allows for viewing the models in their natural way (e.g., viewing the HoQ as a house and not as its underlying graph). Similarly, modeling languages can be constructed for generating conceptual graphs or influence diagrams.

Communication. n-dim provides a variety of communication mechanisms: synchronous and asynchronous. The synchronous is not conceptually difficult compared to the asynchronous. Asynchronous communication is supported by maintaining the context of discussions in addition to supporting history or rationale maintenance modeling languages like IBIS (Figures 4 and 5). For example, when a group negotiates over an issue, a member of the group can alter a model and make it persistent and exchangeable. n-dim maintains pedigree of information in revision models through which the evolution of the model can be easily traced.

External tool encapsulation. n-dim allows to encapsulate external tools written in diverse programming languages and use them in models. One such tool is a natural language processing (NLP) tool that can build terminological structures from a corpora of text (Reich et al, 1993). Another tool allows users to conduct a brainstorming session for eliciting concepts across the network. The development of this tool with n-dim facilities was fairly easy. Other candidate tools for encapsulation include inference engines for conceptual structures or influence diagrams and simulators of qualitative physics graphs.

An architecture of a CQFD tool. Figure 8 shows the architecture layers of an n-dim-based CQFD tool. All but the top layer are n-dim layers (Levy et al, 1993) demonstrating the great advantage of using n-dim as the implementation vehicle. With little attention to lower layers and only concentrating on the first three layers, we can obtain a flexible tool that runs on a heterogeneous environment and provides additional functionalities such as communication or distribution for free. This architecture allows to experiment with different implementations of different layers and optimize the layers in isolation.

Architecture Layers	*Current Implementation*
QFD Tools	**prototype CQFD**
Graph-Based Modeling Kernel	**n-dim**
Implementation Language	**stitch**
Object System	**BOS**
Distributed System	**ISIS**
RDBMS	**POSTGRESS, Informix**
Operating System	**Mach, Ultrix, Sun OS, HP-UX, AIX, OSF**
Hardware	**MIPS, SPARC, HP, RS6000, Alpha**

n-dim layered architecture

Figure 8: An architecture of CQFD

7 SUMMARY

This paper argues that AI tools can be used to improve the effectiveness of manual QFD tools and enhance their functionality. We have shown that AI tools can provide support for the information acquisition, utilization, and communication stages related to QFD tools.

Through the use of an underlying graph representation, QFD graphical models can better share information between themselves and with other computational services developed on graph structures. In this way, CQFD can easily incorporate IBIS-like design rationale models and interface with computational services developed for grid-based knowledge acquisition tools, conceptual structures, influence diagrams, or qualitative physics models.

The paper proposes an architecture based on *n*-dim that provides the infrastructure for embedding QFD tools and their AI support tools. *n*-dim adds many additional attractive properties such as a distributed environment, information management in a database, and communication facilities.

Presently, we are in the process of implementing a simple CQFD tool with an IBIS-like rationale capture method. This tool will be used in a design project course and will provide insight about the usability of CQFD tools before embarking on a full integration with *n*-dim. From experience with other QFD tools, we already know that even such tool leaves much information unrecorded. The use of *n*-dim will allow us to incrementally improve our CQFD design and gradually incorporate into it other AI support tools.

8 ACKNOWLEDGMENTS

The *n*-dim project and its members, Robert Coyne, Douglas Cunningham, Allen Dutoit, Eric Gardner, Suresh Konda, Sean Levy, Ira Monarch, Robert Patrick, Mike Terk, Eswaran Subrahmanian, Anish Srivastava, Mark Thomas, and Art Westerberg, have been the obvious source of the *n*-dim environment but also a source of insight for the ideas presented in this paper.

9 BIBLIOGRAPHY

Akao, Y., editor (1990). *Quality Function Deployment*, Productivity Press, Cambridge, MA. Original Japanese version from 1988.

Claussing, D. (1993). *Total Quality Development*, ASME Press, New York, NY.

Gaines, B. R. and Shaw, M. L. G. (1989). "Comparing the conceptual systems of experts." In *Proceedings of The Eleventh International Joint Conference on Artificial Intelligence*, pages 633–638, Detroit, MI, Morgan Kaufmann.

Gaines, B. R. and Shaw, M. L. G. (1993). "Eliciting knowledge and transferring it effectively to a knowledge-based system." *IEEE Transactions on Knowledge and Data Engineering*, 5(1):4–14.

Gaines, B. R. and Shaw, M. L. G. (1993). "Knowledge acquisition tools based on personal construct psychology." *The Knowledge Engineering Review*, 8(1):49–85.

Howard, R. A. and Matheson, J. E., editors (1983). *Readings on The Principles and Applications of Decision Analysis*, Strategic Decisions Group, Menlo Park, CA.

King, B. (1989). *Better Designs in Half the Time: Implementing QFD Quality Function Deployment in America*, GOAL/QPC, Methuen, MA, 3rd edition.

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.

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.

Mizuno, S., editor (1988). *Management for Quality Improvement: The Seven New QC Tools*, Produtivity Press, Cambridge, MA. Original Japanese edition by JUSE from 1979.

Pohl, K. and Jacobs, S. (1994). "Concurrent engineering: Enabling traceability and mutual understanding." *Concurrent Engineering: Research and Applications*, 2(4).

Reich, Y., Konda, S., Levy, S. N., Monarch, I., and Subrahmanian, E. (1993). "New roles for machine learning in design." *Artificial Intelligence in Engineering*, 8(3):165–181.

Reich, Y. (1995). "Computational quality function deployment *is* knowledge intensive engineering." In Tomiyama, T. and Mantyla, M., editors, *Proceedings of KIC-1: International Workshop on Knowledge Intensive CAD*.

Rittel, H. W. J. and Webber, M. M. (1973). "Dilemmas in a general theory of planning." *Policy Sciences*, 4:155–169.

Sowa, J. F. (1984). *Conceptual Structures: Information Processing in Mind and Machine*, Addison-Wesley, Reading, MA.

Sowa, J. F. (1993). "Relating diagrams to logic." In Mineau, G. W., Moulin, B., and Sowa, J. F., editors, *Conceptual Graphs for Knowledge Representation, Proceedings of the First International Conference on Conceptual Structures (ICCS'93), (Quebec City, Canada)*, pages 1–35, Berlin, Springer-Verlag.

Subrahmanian, E., Coyne, R., Konda, S. L., Levy, S. N., Martin, R., Monarch, I. A., Reich, Y., and Westerberg, A. W. (1993). "Support system for different-time different-place collaboration for concurrent engineering." In *Proceedings of the 2nd IEEE Workshop on Enabling Technologies Infrastructure for Collaborative Enterprises (WET ICE)*, pages 187–191, Los Alamitos, CA, IEEE Computer Society Press.

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

Wermelinger, M. and Bejan, A. (1993). "Conceptual structures for modeling in CIM." In Mineau, G. W., Moulin, B., and Sowa, J. F., editors, *Conceptual Graphs for Knowledge Representation, Proceedings of the First International Conference on Conceptual Structures (ICCS'93), (Quebec City, Canada)*, pages 345–360, Berlin, Springer-Verlag.