Carnegie Mellon University
Research Showcase @ CMU

Department of Electrical and Computer Engineering

Carnegie Institute of Technology

1992

The value of design knowledge

Yoram Reich
Carnegie Mellon University

Carnegie Mellon University.Engineering Design Research Center.

Follow this and additional works at: http://repository.cmu.edu/ece

This Technical Report is brought to you for free and open access by the Carnegie Institute of Technology at Research Showcase @ CMU. It has been accepted for inclusion in Department of Electrical and Computer Engineering by an authorized administrator of Research Showcase @ CMU. For more information, please contact research-showcase@andrew.cmu.edu.

NOTICE WARNING CONCERNING COPYRIGHT RESTRICTIONS:
The copyright law of the United States (title 17, U.S. Code) governs the making of photocopies or other reproductions of copyrighted material. Any copying of this document without permission of its author may be prohibited by law.

The Value of Design Knowledge
Yoram Reich
EDRC 05-63-92

In *The Proceedings of the Seventh Banff Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff, Canada, 1992, pp 21-1-21-20*

THE VALUE OF DESIGN KNOWLEDGE

Yoram Reich
Engineering Design Research Center
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213, USA

ABSTRACT

The quality of design knowledge a system has substantially influences its performance; good quality design knowledge is a true asset. Often, the terms knowledge, its quality, and how it is measured, are left vague enough to accommodate several interpretations. This paper articulates two definitions of knowledge and their associated measures. A detailed example of knowledge evaluation using the measures is described. The example demonstrates the value of knowledge quality evaluation. This value is an addition to the methodological function that evaluation provides. Finally, the paper briefly discusses the scope of the measures and their relationships.

1 INTRODUCTION

Design knowledge exists in many ways and qualities: It may be referred to as being accurate, perfect, plausible, approximate, cheap, expensive, implicit, explicit, opaque, etc. Sources of good quality design knowledge are limited; they mostly exist in the minds of expert designers, while a small portion is documented in the literature. In order to identify which knowledge is good, one must have the ability to evaluate knowledge quality. The topic of knowledge evaluation raises several prerequisite issues. First, what do we mean by design knowledge? Second, how is something in general being evaluated or measured? This question may seem unimportant, but in fact, it is central to the discussion since it deals with the concepts of *measure* and *value*.

A measure of design knowledge quality can serve several functions. The first and most general function is practical: The identification of good design knowledge for use by designers or inclusion in computer systems. The second function is methodological: The evaluation of prototype design systems developed in research and determination of their relative merit This evaluation is essential for providing feedback on research progress and for supporting the refinement of ideas (Cohen and Howe, 1989). When evaluating design systems that learn, this function becomes mandatory.[1] A third function is manifest when complex design systems with several modules that solve similar problem compete on resources. In this situation, knowledge quality measures can identify which module to invoke. In this study, we concentrate on the second function: the evaluation of knowledge embedded in design support systems.

The subject of this study belongs to a larger set of methodological and practical issues of evaluation. As we see later, the evaluation of knowledge quality is tightly coupled with the evaluation of intelligent systems. The state of evaluating such systems was documented by Green and Keyes

[1] In fact, research on learning provides significant insight on the measurement of knowledge since the evaluation of learning progress requires the evaluation of knowledge acquired.

(1987). They described the reluctance of system developers to verify and validate their products. The lack of effort spent on evaluating software systems in general, and expert systems specifically is also discussed by Adelman (1991) and Cohen and Howe (1989). The latter also demonstrated the benefits from evaluation to research and the necessary steps needed to allow evaluation to be performed.

If evaluation of systems is addressed in studies, it is often performed *in an ad hoc* manner. To illustrate, Zuse and Bollmann (1987) discussed the chaotic state of measures in software engineering. Most of the software measures appearing in the literature do not conform to the notions of measurement theory (Stevens, 1946). The argument is not that one measure is better than the other, but that a true measure must have certain properties. First, there must be two systems, *observed* and *formal,* where a system is a mathematical entity consisting of a set of objects A, relations on A, and a binary operation on A. For example, an observed system may contain the set of physical objects, the relation "hcavier-than," and the operation of assembling two objects. The formal system can consist of the set of positive real numbers, the relation "larger-than/^{9} and the addition operator. Second, there must be a function from the observed to the formal system that preserves the relations and operation, called *homomorphism.* In the above example, a mapping that assigns each object its weight is a homomorphism. This allows for a meaningful assignment of *values* to the weight of physical objects. Most measures of software systems do not have these properties, although all *should have* according to Zuse and Bollmann. Similarly, one may argue that measures that evaluate knowledge should have these properties.

The preceding discussion assumed that evaluation is quantitative, which usually is perceived as the only "good" evaluation possible. Nevertheless, often such evaluation is impossible to execute and sometimes it is inappropriate. In these cases, combined qualitative and quantitative evaluation methods are appropriate (Kaplan and Duchon, 1988). The quantitative measures must be designed to have the properties of measures; however, the qualitative methods cannot have these properties.

This paper proposes four techniques for evaluating knowledge: structural qualitative and quantitative, and functional qualitative and quantitative. It contrasts their advantages, disadvantages, and appropriateness for different purposes. The paper illustrates them in the evaluation of an experimental design system. It does not deal with the debate whether machines can have knowledge or be intelligent (Searle, 1980; Dreyfus, 1979); and whether knowledge in a system can be "objectively** evaluated in any way.[2]

This paper attempts to raise the issues and focus the attention on the methodological and practical implications of the evaluation of knowledge, thereby motivating further research in this direction. Such research may extend the use of constructs from continental philosophy in the evaluation of knowledge, hence, address some of the deficiencies in knowledge evaluation briefly pointed later.

The remainder of this paper is organized as follows. Section 2 provides two definitions of knowledge and briefly discusses them in the context of design. Section 3 describes four evaluation measures of knowledge. Section 4 describes the design system BRIDGER that is used to demonstrate the evaluation measures. Section 5 provides a detailed example of measuring the bridge design knowledge that BRIDGER has. The extent of the evaluation allows the appreciation of the many concerns affecting the selection and application of different evaluation measures of knowledge. Section 6 concludes the paper.

[2]This comment disagrees with Gaines and Shaw (1989) reference to *"objective knowledge"* as the knowledge arrived by consensus among experts; such knowledge is subjectively created within a constructivist viewpoint. Furthermore, although most empiricists will claim that a measure as described before can be objective, they will thereby ignore the fact that the mapping from the observed to the formal system is theory-laden and subjective.

2 WHAT IS (DESIGN) KNOWLEDGE?

In defining knowledge it must be understood that the definition dictates the type of evaluation measure that can be applied or that may best apply. Many studies on knowledge-based systems avoid the question of what knowledge is by discussing knowledge representation. Implicitly, these studies define knowledge as *whatever is represented,* henceforth termed the *structural definition.* Knowledge is therefore a static entity; it may include facts, axioms, derivations, causal relations, mathematical models, etc.

The structural definition is in contrast to *Hatfunctional definition* which states that knowledge has a purpose and it is defined as *what a system has that allows it to attain goals.* Knowledge cannot be observed directly, but indirectly through observing the "intelligent" behavior of a system. Doyle (1988) termed these definitions as: (1) *explicit knowledge* which is what is represented; and (21 *implicit knowledge* which is what can be deduced from explicit knowledge. Doyle argued that adopting the explicit definition has several limitations: (1) it cannot explain actions that are not logical, default or nonmonotonic reasoning; (2) it cannot explain some psychological phenomena; and (3) it cannot handle inconsistencies that naturally arise, for example, in knowledge generated from several experts.

2.1 Structural Definition

The structural definition has several appealing properties. The main one is knowledge sharing (and trading). If knowledge is what is represented, then knowledge can be abstracted from the system using it and shared with another reasoning mechanism. Two "quantities" of knowledge could be added to yield a larger knowledge base, or knowledge could be transferred between knowledge systems (Neches et al., 1991). A knowledge interface format *(KSF)* is developed to facilitate this transfer (Genesereth and Fikes, 1990). Another appealing property of the structural definition is that it facilitates easy evaluation performed by simply inspecting the declarative structure of knowledge. This is much cheaper than executing behavior assessment experiments.

The structural definition detaches knowledge from its method of acquisition and use, although often, theactof acquisition determines the meaning ofknowledge. This may cause difficulties. For example, in order for systems with different uncertainty management mechanism to cooperate they must establish an interface between them where uncertainty values are translated via a fractionally linear function (Kreinovich and Kumar, 1991). If translation is not performed, the ability to use the knowledge decreases (e.g., the decrease of diagnosis performance in MYCIN when the method of probability calculation is changed (Shordiffe, 1976)). This suggests that one cannot generate and "plug" knowledge that was derived by some mechanism into an expert system and expect it to function properly. Note however, that the structuralists may not worry about this since their definition of knowledge does not include the functionality of knowledge.

2.2 Functional Definition

Newell (1982) opposed the structural view of knowledge. He argued that *"knowledge* is a distinct notion, with its own part to play in the nature of intelligence," independent of representation. In order to define knowledge, Newell defined the "principle of rationality: If an agent has knowledge that one of its actions will lead to one of its goals, then the agent will select that action." This principle governs the use of knowledge for making the appropriate actions. Knowledge is therefore

defined as "whatever can be ascribed to an agent, such that its behavior can be computed according to the principle of rationality. [...] Knowledge is a competence-like notion, being a potential for generating action;"[3] therefore, knowledge should be evaluated functionally. The relation between knowledge and representation is clear. First, "representations exist at the symbol level/' Second, "knowledge serves as the specification of what a symbol structure should be able to do."[4]

23 Design Knowledge

Design knowledge can be defined as knowledge in general, except that it is about design. Since the nature of design processes and knowledge is orthogonal to the question of its evaluation, we will refer to design as a process mapping from requirements or needs to a description of an artifact. In the structural definition, design knowledge may include facts about physical sciences; facts about modeling techniques of, and experiental heuristics about, artifacts; and information about how artifacts may be structured, how do they behave, and how they may be built. At this level, knowledge may be abstracted from its representation; there need not be any commitment to specific representational formalism.

In the functional definition, design knowledge is what can be ascribed to a design agent, such that its design behavior can be computed according to the principle of rationality. This means that knowledge can be described in terms of its operation to satisfy the goals of the design system. If a system cannot use a piece of information in its reasoning, then, the system *does not have* this knowledge.

We argue that the ultimate purpose of design knowledge is to act, reason, and create artifacts; design knowledge as an entity separated from use has no significant meaning and usefulness. We therefore favor the functional definition of design knowledge. Nevertheless, we do not, and probably cannot, resolve which definition of design knowledge is better or correct, we merely provide the definitions and contrast their evaluation.

Researchers on design have used both definitions and in various ways. Balkany, Birmingham, and Tommelein (1991) provided the description of several design systems at the knowledge-level. As such, they favor the functional view of knowledge being separated from implementation issues and geared toward functionality.

Bijl (1987) presented a view against conceptualizing the information within computers as knowledge, a statement indirectly in favor of die functional view (or even in line with continental philosophy). On the other hand, he described a system called MOLE as resting on the foundation of comprehensibility of design knowledge (p. 12). In MOLE, the internal representation is designed to be meaningful and to convey the functionality of the system—this corresponds to the structural view of design knowledge.

Brown and Spillance (1991) discussed knowledge compilation as a process that modifies the structure of design constraints. They view the quality of the modifications in the functional sense and evaluate them by performance tests.

Coyne, Rosenman, Radford, Balachandran, and Gero (1991) said that "in knowledge-based sys-

[3] See Fetzer (1990, p. 127-130) for a critic on this definition. In addition to this criticism, continental philosophy has significantly different ideas about the nature of knowledge, its interpretation, and understanding (Mueller-Vbllmer, 1985). See also (Mallery et al., 1986), for a review of hermeneutics related to computer understanding.

[4] The view of knowledge as a specification for its structural description was used, for example, by Levesque (1984) for building knowledge representation that can support certain functions; and by Kyburg (1988) to draw conclusions on how uncertainties should be represented to support decision making.

terns, the knowledge must be able to be made explicit in such a way that it can be inspected and understood independently of the way it is controlled." Therefore, they view knowledge in the structural sense.

Whereas researchers adopting the symbolic paradigm of design can subscribe to either definitions of knowledge, the studies employing neural networks in design must subscribe to the functional definition since the internal structure of a neural network does not reveal its knowledge content

3 EVALUATION MEASURES

The evaluation of knowledge depends on its definition. Two measures, qualitative and quantitative, are proposed as supplying complementary information for interpretation and validation (Kaplan and Duchon, 1988).

There can be other measures of knowledge, for example, cost or simplicity. The cost of knowledge can measure the time and memory resources needed to manage it and the cost of using it to obtain performance. Simplicity can measure syntactic properties of knowledge structure or the trace of its execution. We view these and potentially other measures as secondary, provided that they are not exponential functions of any of the problem characteristics; otherwise, they would become dominant in the evaluation process.

3.1 Structural Measure

The process of *structural* evaluation is similar to a "brain surgery." Knowledge is measured based on its internal structure. Since a crucial aspect of knowledge is its use in performance tasks, the structural measure will probably involve *projecting* how the knowledge will perform while solving problems. This however will be hard without considering the mechanisms that manipulate that knowledge.

In the case of logic, knowledge can be structurally assessed according to its truth value. In rule-based systems (albeit without uncertainties) knowledge can be verified to be free of redundancy, conflicts, circularity, and incompleteness (Nazareth, 1989). This measure, however, is not useful for evaluating design knowledge which *has* all the above properties, but yet, used by humans effectively to generate designs. The qualitative and the quantitative variants of this measure are discussed next

Qualitative measure. This measure can be traced to the assumption that humans can understand knowledge embedded in systems in a declarative form. It is based on past practice with small systems having simple knowledge representation such as traditional production rules. However, in large and complex systems, it is hardly possible to understand what is the role of a small piece of knowledge and envision its potential run-time interactions with additional knowledge. Therefore, while the ability to inspect declarative knowledge can facilitate debugging, maintenance, and even education, it is difficult to support the application of this measure.

A different approach to support these concerns builds upon advanced mechanism that *manipulate* the knowledge, thereby transforming the evaluation from structural to functional. For example, GUIDON2 uses the knowledge of NEOMYCIN to teach students (dancey, 1988) without letting the student look at the low level knowledge representation.

In machine learning, the comprehensibility of knowledge generated is described as an important measure by some researchers (Michalski, 1986). Note, however, that this measure has evolved from research on concept learning where the structure of knowledge is sufficiently simple and the task is a single-step classification that is easy to comprehend and evaluate. Furthermore, it is easy to predict the knowledge performance from its structure.

The execution of this measure requires domain knowledge. It is not clear how to quantify it or otherwise how to use it to compare between different knowledge contents.[5]

Quanitative measure. This measure quantifies the structure of knowledge. Viewing knowledge as information allows such quantification (Boulton and Wallace, 1973). A heuristic measure for assessing the quality of a classification, which is relevant to the example described later, is proposed by Gluck and Corter (1985). A generalized form of this measure, called *knowledge utility (KU)* is used in the example (Section 4)

This measure is easy to apply, but requires a knowledge representation formalism that can be quantified. Compared to the qualitative structural measure, it is even less clear what is the relation between this measure and the design performance of a system.

3.2 Functional Measure

This measure evaluates knowledge based on its performance in various tasks. Adelman (1991) discussed three types of performance evaluation: (1) *experiments* which are suited for the early stages of system development and generate fully reproducible results; (2) *quasi experiments* which are for the operational stage of systems and consists of fully controlled artificial studies; and (3) *case studies* which are opportunistic and wholly unconstrained The first two are quantitative measures and the latter is a qualitative measure.

In the context of statistical decision theory, (and with several other assumptions,) one can quantify the value of knowledge. Skyrms (1990) defined knowledge as something that allows making *informed* decisions; he then used this definition to evaluate knowledge content Such an evaluation is possible due to the simple nature of the knowledge involved: a single piece of evidence.

Cohen and Howe (1989) discussed the evaluation of large artificial intelligence systems. They suggested several performance experiments that can be used for this purpose. One of the systems they discussed is DOMINIC, a system for the routine design of mechanical devices (Howe et al., 1986). They shown how performance evaluations guided their research through three generations of the system. The qualitative and the quantitative variants of this measure are discussed next.

Qualitative measure. This measure can be viewed as performing protocol analysis (Ericsson and Simon, 1980) on the system for evaluating its knowledge. The system is used to solve a variety of problems, and its problem-solving behavior is coded and analyzed. The analysis uses language different than that implementing the knowledge internally. Domain knowledge can further enhance the evaluation of the behavior and therefore, the knowledge evaluation.

This is one of the important techniques for evaluating systems (Cohen and Howe, 1989). It is often used in design research. The problem is that often, very few examples with very limited scope are

[5] **In fact, this measure is related to hermeneutics and as such subjected to the debates about, and variations of, it (Malleryetal.,1986).**

provided and usually without substantial analysis.

Quantitative measure. This measure is based on the performance of a system over many problems that span the range of problems the knowledge of the system is expected to solve* The performance of the system is compared with the solutions generated by human experts or normative theories such as decision theory. The solutions can be also evaluated based on shared standards such as design codes. The latter is used in the example discussed later (Section 4).

For example, Gaines (1989) used a functional quantitative measure to evaluate the initial state of knowledge of a learning program. The initial knowledge serves as a base-line for the learner and the amount of data necessary to generate a predetermined fixed performance level measures the quality of the initial knowledge.

In general, the measurement of performance of systems and their comparison is not straightforward. An example from performance evaluation of learning programs is discussed by Kononenko and Bratko (1991). One of the causes for this difficulty, they mention, is that the types of answers from different programs are not exactly the same.

Since this measure summarizes the results in statistics, it tends to hide some details of the system's behavior. Many controlled studies are needed to uncover behavior patterns that can be identified (although only qualitatively) by the qualitative functional measure.

4 BRIDGER

In order to ground the different evaluation measures and their tradeoffs it is necessary to demonstrate them in an evaluation of an existing system. The following section, describes the evaluation of design knowledge accumulated within a design system BRIDGER, a system developed to explore the potential of knowledge acquisition techniques for building design assistant for the preliminary design of cable-stayed bridges.

The section starts by describing the bridge domain and then reviews the system's architecture and operation. Since the purpose of this paper is to focus on knowledge evaluation and not on BRIDGER, only the necessary parts required for the demonstration are described.[6]

In essence, this section provides data point for motivating knowledge evaluation and exploring the properties of different measures. As such, this paper can be perceived as a detailed case study (i.e., Adelman's (1991) third category of experiments discussed before). Therefore, it is a qualitative evaluation of the concept of knowledge measures discussed in this paper and it is a functional evaluation since it discusses the "functionality" that these measures can provide.

4.1 Domain of Cable-Stayed Bridges

Figure 1 describes the main components and dimensions of a cable-stayed bridge. A cable-stayed bridge is composed of a superstructure and a substructure. The superstructure is composed of a deck, towers, and stays that are attached to the towers and support the deck. The figure shows some of the properties which are used to describe a cable-stayed bridge; additional properties include: SPAN-N, the number of spans of the bridge; DECK-A, DECK-MI, TOWER-A and TOWER-MI,

[6] Sec (Reich, 1991a; Reich, 1991b; Reich, 1992) for further details.

the cross-sectional areas and moment of inertia of the deck and the tower, respectively; DECK-M, the material of the deck; and STAY-A, the cross-sectional area of the stays.

In a design scenario the requirements are expressed as a set of specification property-value pairs (e.g., the required length of the bridge). Design is then executed by making design choices as assignments of design-description property-value pairs (e.g., SPAN-M=500ft., STAY-N=20).

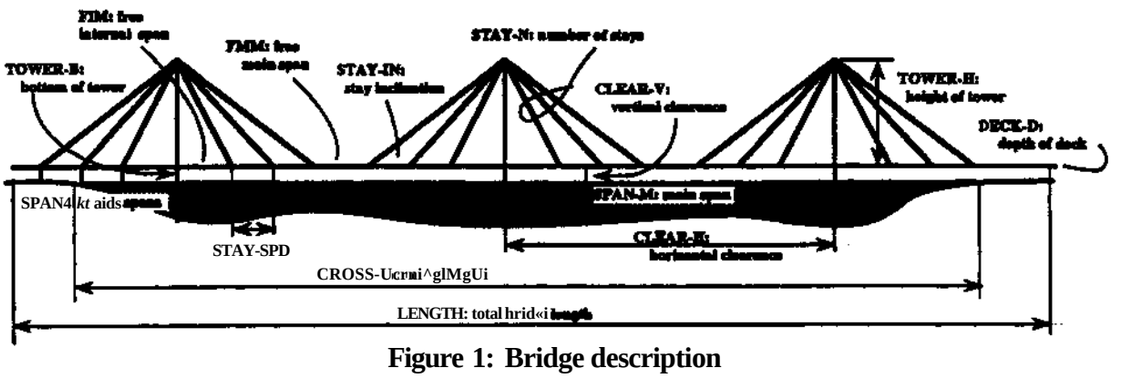


Figure 1: Bridge description

A rough illustration of the complexity of the domain can be conveyed by the number of properties used to describe various aspects of the problem: 9 properties describe a specification; 30 properties describe a design, 15 properties describe the analysis results, and 4 properties describe the evaluation of a bridge.

4.2 BRIDGER'S Architecture

BRIDGER'S architecture, shown in Figure 2, consists of two main systems: synthesis and redesign. The synthesis system is responsible for synthesizing several candidates from a given specification. Synthesis knowledge is generated by learning from existing designs and from successful design examples that are selected by the user. Since the knowledge created is heuristic by nature, candidate designs are usually inadequate in some aspects. The redesign system resolves this problem.

Candidate designs are transferred to a module that analyzes them and submits them to a redesign module, if necessary. The redesign module is responsible for modifying designs after their analysis. On receiving the analysis results, this module retrieves the best design modification for the bridge, The user can override the redesign modifications and supply explanations that enhance redesign knowledge. The results of the redesign system are acceptable designs. The designer evaluates the results and can submit a subset of them to the synthesis system for further training.

ECOBWEB is the learning system that implements the synthesis system. It acquires synthesis knowledge and uses it to synthesize new bridges. ECOBWEB represents knowledge in a classification hierarchy. It has several operators that build the classification from examples. Learning and synthesis progress by using one-step look-ahead search in the space of classification hierarchies directed by an evaluation function to select the best operator.

The evaluation function, called *category utility (CU)*$_9$ evaluates a classification of a set of designs

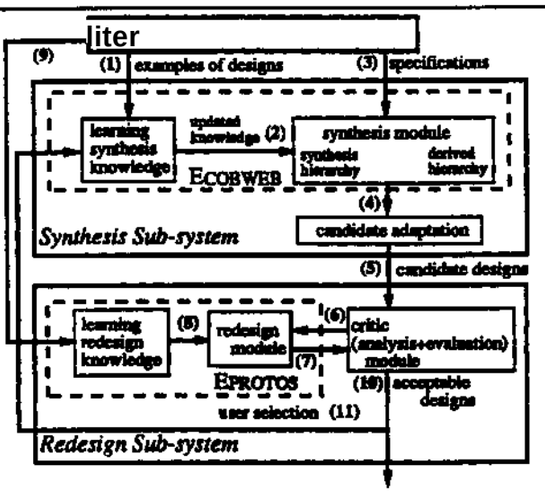


Figure 2: BRIDGER's architecture

into mutually-exclusive classes $C\backslash$, C2,•.., C, by:

$$CU = \frac{\sum_{k=1}^{n} P(C_k) \sum_i \sum_j P(A_i = V_{ij}|C_k)^2 - \sum_i \sum_j P(A_i = V_{ij})^2}{n} \quad (1)$$

where C^ is a class, A; = Vy is a property-value pair, $P(x)$ is the probability of JC, and n is the number of classes. The first term in die numerator measures the expected number of property-value pairs that can be guessed correctly by using the classification. The second term measures the same quantity *without* using the classes. Thus, the category utility measures the expected *increase* of property-value pairs that can be guessed *above* the guess based on frequency alone. The measurement is normalized with respect to the number of classes. The higher is the value of CU_y the better the quality of the classification is.

BRIDGER has a variety of synthesis strategies, ranging from case-based to prototype-based design and from extensional to intentional strategies. To simplify the discussion we use the simplest strategy: an extensional case-based strategy. In this approach, BRIDGER *retrieves* a pre-determined number of candidate designs from the classification hierarchy. The candidates are complete descriptions of previously designed bridges. BRIDGER then *adapts* these candidates to fit the new specification by performing various scaling operations.

5 EXAMPLE

This section describes a detailed evaluation of BRIDGER's design knowledge as it develops through learning. Four hierarchies, $K\backslash_y$ *Ki* Ki,* and £4, were generated. Hierarchy $K\backslash$ was generated from a set of original 96 bridge examples. Hierarchy *K2* was generated from the 96 examples after their analysis and redesign. Therefore it contains higher quality examples. Hierarchies *K$* and *K** were generated from 144 and 192 good quality examples, respectively.

Our "unit" of knowledge will be a training example for ECOBWEB.[7] An additional example

[7] Here, we can see the theory-laden nature of evaluation: considering an example as a unit of knowledge is natural

modifies knowledge and the subsequent performance. In the terminology of measurement theory, the relation on knowledge will be "better-than," and the binary operation is appending-hierarchies.[8] Since hierarchy $K\backslash$ is equal to K_2 in terms of the number of examples, we cannot treat them as distinct according to the selected unit of knowledge. On the other hand, $K\backslash$ is built of lower quality examples and cannot be compared meaningfully with £3 or AT4; Nevertheless, for reference purposes we included its evaluation as well.

5.1 Structural Measure

The structural content of knowledge was measured in two ways: quantitative and qualitative. The qualitative measure attempts to "understand" the knowledge generated from a domain perspective; and the quantitative measure is based on the evaluation function employed by ECOBWEB for assessing classifications.

Qualitative measure. Figure 3 shows the *K%* synthesis hierarchy generated from 96 examples of bridges. The classes are described with some of their properties. Some properties are shown in bold font; these are the characteristic properties. Intuitively, characteristic property values of a class are those property values that are very common in the class and rarely appear in the other classes of the same level. The figure also shows the name of each class and in parenthesis the number of bridges used to generate it

The hierarchy is subdivided into two large subclasses: class C which contains long bridges (i.e., long LENGTH and SPAN-M properties) with many stays, and class B which contains short bridges with fewer stays. Further subdivisions mainly reflect differences in the LENGTH, CROSS-L, SPAN-M, SPAN-N, and DECK-M properties. Several patterns emerge in the hierarchy. They can be interpreted using domain knowledge, and may point to some design heuristics. The number of examples, however, is not sufficient to allow learning to discover strong patterns; any explanation should cautiously be accepted.

For example, class B contains only bridges with steel decks and class C contains 44% concrete-deck bridges; in addition, the average main span of class B is shorter than that of class C. These trends point to a preference for using concrete for longer bridges and steel for shorter bridges. The first is correct, but the second is n o t/ A close look at the subclasses of C shows that H and I, which contain only steel bridges, have longer average main span than the two other classes, containing mainly concrete-deck bridges. Therefore, the preferences stated before no longer apply. The conclusion is that the average main span value of C is only a common value of the class but does not necessarily provide a good characterization of the class.

This example demonstrates the subjective and imprecise nature of the qualitative structural measure. It is not clear which aspects in the class description are important and how they should be interpreted. It is not clear how the *value* of the measure is to be defined. Is it by a carefully controlled statistical study in which different experts rate the knowledge? Is it by counting "interesting" statements

---

to ECOBWEB, but may not make sense for other systems.

[8] The operator appending-hierarchies can be defined as taking one hierarchy and training it by the examples used to generate the second hierarchy. This is an order dependent procedure that causes a violation of a commutative property of measures.

[9] See Tfcble 3.3 in (Podolny and Scalzi, 1986) showing lower bids for concrete bridges as opposed to steel bridges for several recent long bridges. Also see Figure 3.8 of that reference showing that concrete is preferred to steel for short spans.

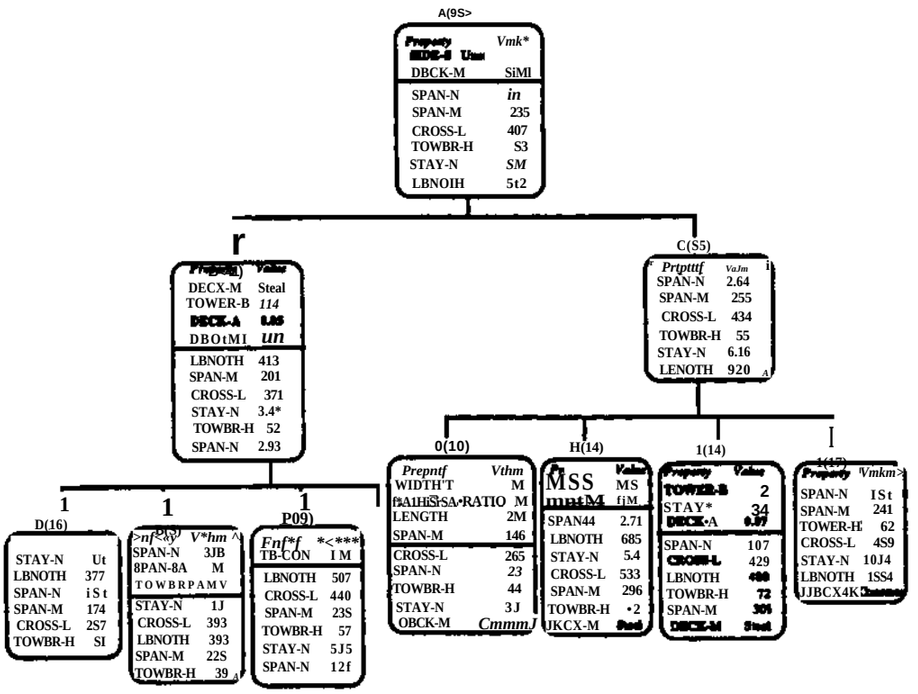


Figure 3: K_2 synthesis knowledge base

made about the hierarchy?[10]

Observing the content as the hierarchy grows can potentially explain some of the design behavior revealed by the other measures. Figure 4 illustrates the growth pattern of the synthesis hierarchy. It reflects the organization of knowledge rather than its content Initially, the hierarchy is "flat," consisting of the root node and its leaves. When additional information is accumulated, a second level starts to grow. Approximately twice the number of examples is required to form that second level. This pattern of growth continues later.

The design performance (i.e., quality and time) is not expected to improve continuously, but rather in stages. To illustrate (see Figure 4), assume that a design is initiated with hierarchy (a) and that the best candidate is class C_2. If BRIDGER is asked to synthesize *n* candidates, it will consider all the sons of Q and output the *n* best matches to the new specification. After additional training, hierarchy (b) is generated and used for the same design; synthesis progresses from C_3 to C_4, and finally, to *C$*. Now, synthesis chooses the best *n* sons of class CU as candidate designs. The sons of CU form a more homogeneous class than the sons of Q, but this has required doubling the number of training examples. Additional training leads to the generation of of hierarchy (c). If synthesis progresses through the path Q, C7, C10, and Cu, then candidates are generated from the sons of C10 (ft out of the 6, assuming than $n \leq 6$). If the path ends at class C9, the candidates are generated from the sons of C7 (ft out of the 20, assuming than $n > 2$). The first case will demonstrate an overall performance improvement, but the second will show similar performance

[10]Hermeneutics will reject the idea of quantifying this "measure" arguing that the qualitative knowledge evaluation should maintain its present nature and be elaborated further as the text proceeds.

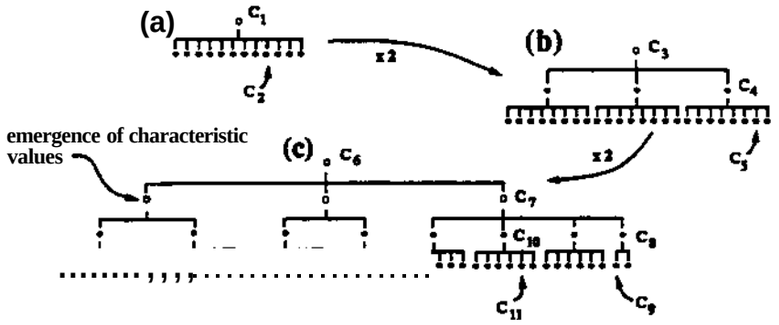


Figure 4: Qualitative description of hierarchy growth

quality and degraded time performance to that demonstrated by hierarchy (b). These differences suggest that learning is not continuous, although it may seem so when performance is averaged over many cases.

Summary of qualitative structural measure.
This measure is subjective and incomplete (i.e., no value is assigned to the measure). The evaluation can provide insight about the behavior of knowledge in design; mainly, since the internal mechanisms of the systems are known. However, in general, the internal mechanisms of a system are unknown or too complex and it may be hard to extrapolate the functionality of knowledge from this measure.

Quantitative measure. The quantity of knowledge can be defined by a measure, called *knowledge utility (KU),* that calculates the *increase* in the number of properties that can be predicted for a given specification when using the *hierarchy,* relative to the number of properties that can be predicted by using values' frequency. The category utility function *(CU)* calculates the same item for a *classification,* not for a *hierarchy.* Applying *CU* recursively, starting from the root of the hierarchy, yields the desired utility measure:

***knowledge - utility (class)*:** **(2)**

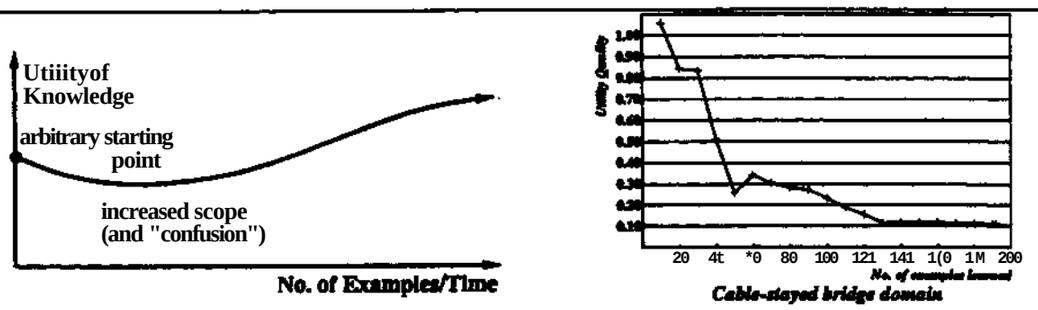
***if class is a leaf class, return* 0.0;**

else, return CUxn+ J2 W*) x knowledge-utility (son).
kesomsof class

After the calculation, the value is normalized by the number of properties describing artifacts.

When experience grows, the design knowledge emerging in the hierarchy is expected to converge to better knowledge as shown in Figure 5(a). Starting from an arbitrary value influenced by the set of initial designs, the scope of knowledge increases as additional designs are learned. In principle, the larger scope allows for the generation of more designs; however, since new properties or property: value pairs have little association with previous property-value pairs, the quality of the generation will be poor. This initial growth of scope may reduce the quality of the existing knowledge. As more designs are learned, knowledge is accumulated on many possible combinations of property-value pairs and the utility increases.

Figure 5(b) shows the knowledge utility as a function of the examples learned by BRIDGER for the domain of cable-stayed bridges. The value of about 0.1 reached after learning 192 examples suggests that approximately 8 out of the 58 properties describing designs can be predicted accurately.



(a) (b)

Figure 5: Improvement of design knowledge

This may seem to be a rather disappointing result since it is difficult to envision that a knowledge with such a low utility can be helpful in synthesis. It should be remembered, though, that knowledge utility calculates the utility *above frequency prediction,* and it averages this calculation over the complete classification, including its insignificant parts.

Summary of quantitative structural measure.
This measure is abstracted from the mechanisms that manipulate knowledge although it still relies on *CU* which certainly governs the system's behavior. The bad value assigned to this measure in the evaluation is in contrast to the good performance reported later, this discrepancy demonstrates the difficulty in formulating good quantitative structural measures that can be used to predict performance.

52 Functional Measure

This measure evaluates knowledge in synthesis activities performed on 48 test specifications (see (Reich, 1991a) for details). The four knowledge hierarchies, *K\9 K2,* £3, and AT4, were used to synthesize 4 new candidates for each specification. The 192 (48 x 4) synthesized bridges were used in the qualitative and quantitative functional evaluations.

Qualitative measure. Instead of analyzing the complete trace of synthesis, we focus the evaluation on one important synthesis step: the retrieval of candidate designs. Table 1 shows the number of different existing designs retrieved, and the names of the designs most commonly used. The number of times each existing design was used in the generation of candidates, out of the 192 new candidates, is given below each name. The small number of designs retrieved reflects an internal tendency that characterizes the knowledge. In light of the discussion on Figure 4, it is not surprising that the structure of the hierarchy can lead to such behavior at the early stages of learning.

The difference in the number of designs retrieved suggests a synthesis pattern similar to that presented in Figure 4. In particular, the synthesis pattern emerging from the *K* and the *K%* hierarchies is probably similar to the path Q, C7, Cg, and C9. Such a path forces the retrieval of designs from classes of designs higher in the hierarchy. Since a large class is used as a source of existing designs, the selection would usually favor a small number of 'strong' matches. If the path is from Q> to Cu, the selection would be from smaller and different groups of classes, leading to the retrieval of a larger number of distinct designs. This is the case when *K3* and £4 are used.

Table 1: Summary of retrieved designs

Knowledge	# of different designs	designs retrieved # of times retrieved (out of the 192)							
K_1	12	E10 45	E12 45	E49 45	E60 45				
K_2	8	E46 43	E55 43	E78 43	E91 43				
K_3	19	E2 16	E10 16	E& 16	E88 16	E19 12	E22 12		
		624 12	E26 12	£2£ 12	E49 12	E115 12	E111 9		
K_4	19	E80 25	E144 25	E192 25	E3 15	E159 13	E162 13	E168 13	E135 13

Figure 6 shows the four designs most often used when synthesizing with the K_2 hierarchy. These designs are listed in Table 1. The two designs on the right are scaled down by a factor of two. All four designs are two-span bridges with average main span (224 m). The range of spans is large, allowing the retrieval of designs that are relevant to a new specification therefore do not require significant scaling.

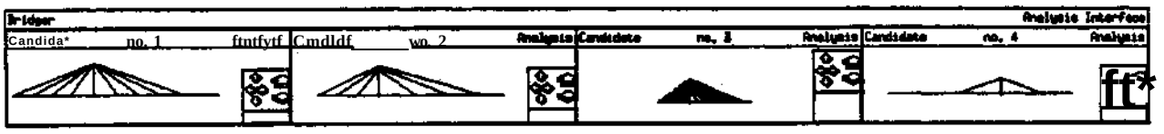


Figure 6: Functional assessment of *Kz*

Figure 7 shows the 12 designs most often used when synthesizing with the *K$* hierarchy. Most of them are three-span bridges. The average length of the main span is 179 m. A surprising observation is that most of the bridges have a small number of stays. This fact and the observation that almost no three-span bridges were used by the *K%* hierarchy point to the existence of a *shadowing* phenomenon. Certain bridges are not retrieved since they reside on hierarchy branches that are rarely visited. But once these branches become accessible, their leaves start being used as candidates.

Figure 8 shows the 8 designs most often used when synthesizing with the *K** hierarchy. There is a better balance between two- and three-span bridges, and more variation in the number of stays. The length of the main span of these designs is longer than before (345 m) and its variability is slightly less then that observed for the *Kt* hierarchy. The increasing average length helps design large bridges without compromising the design of bridges with small spans.

Summary of qualitative functional measure.
This measure can be used to predict the quantitative measure by generalizing over behavior patterns. It can also be used to confirm the qualitative structural measure. A more detailed qualitative measure can point to important issues that need to be addressed in improving the system, for example, should the *shadowing* effect discussed before and confirmed here be handled? how should the measure be quantified, by the number of designs retrieved, their quality, their variety?, etc..

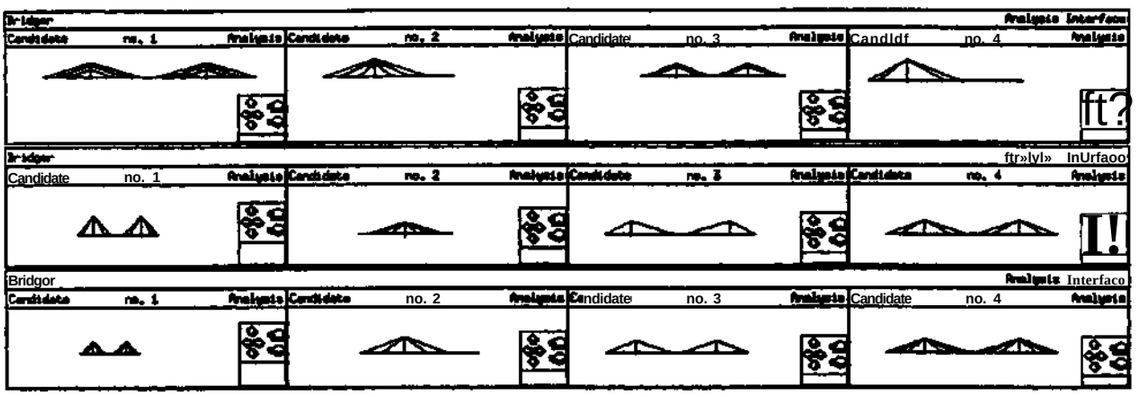


Figure 7: Functional assessment of £3

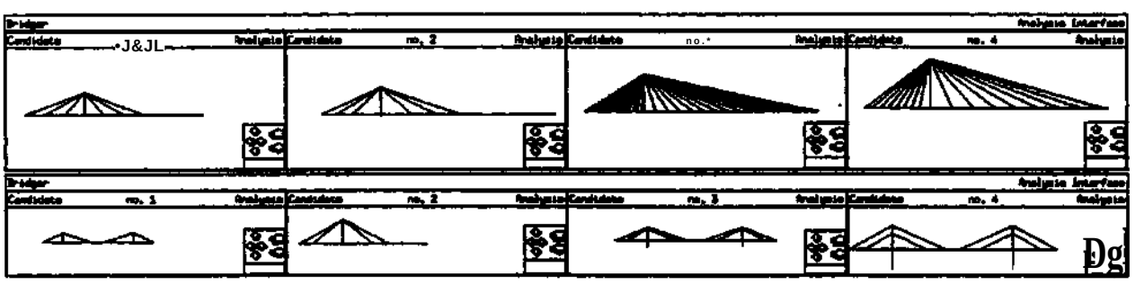


Figure 8: Functional assessment of A4

Quantitative measure. This measure evaluates the design performance of BRIDOER while designing candidates for the 48 test specifications. Since the power of BRIDGER's synthesis process comes from two sources: (1) the retrieval of designs closely related to the new specification; and (2) the adaptation of candidates with scaling values; testing must address both.

The retrieval process is evaluated by the amount of *scaling* of the main span of the retrieved design needed to satisfy the new specification; it measures how close is the retrieved design to fulfilling the dimensional specification. The candidate adaptation process cannot be tested independently. The combination of the two processes is tested by measuring the *quality* of candidate designs. The quality of a design is a weighted summation of the constraints that a design violates (Reich, 1991a). In the terminology of measurement theory, the hierarchies are the observed system and the scaling or quality values (i.e., the real numbers) are the formal system. The homomorphism maps hierarchies to values.

The specifications used in the testing have considerable influence on the results. Specifications can reflect the average specification of the bridges designed thus far or be far from these average. Traditional expert systems will fail to perform on specifications that do not reflect their experience or range of operation, a phenomena called *brittleness.* The experiment also evaluates this aspect

Table 2 provides the statistics of the *scaling* needed to adapt the candidates to the specifications of the 48 test problems and the *quality* of the designs synthesized The columns denoted by *total* provide the average of these measures. The columns denoted by *lower, average,* and *upper,*

provide the results for three groups of specifications corresponding to far-lower-than, similar-to, or higher-than, the average specification of existing bridges. These groups roughly divide the set into three equal parts.

Table 2: Scaling and Quality statistics of candidates

Knowledge	Scaling				Quality			
	lower	average	upper	total	lower	average	upper	total
	1.25	338	555	33X74	118831	35.63	62.22	278.36
K_2	0.97	250	4.08	2.154	0.34	4.61	325.81	50.19
K_3	0.97	253	357	2.092	057	255	5.67	2.89
K_4	0.88	132	2.85	1325	0.41	0.73	3.06	1.20

A MANOVA (Hays, 1988) analysis was performed to assess the statistical significance of the differences in the performance levels observed. The total scaling values satisfy: /^ *K3* >o.oi *K4;*[11] where the >o.oi indicates that *K%* and *K3* are greater than *K4* with statistical significance at the $p < 0.01$ level and that the difference between K_2 and £3 was not statistically significant Therefore, *the more knowledge* BRIDGER *has, the more relevant are the retrieved candidates.* The improvement is not a smooth function, but occurs in steps as predicted by the structural qualitative measure. The total quality values satisfy: K_2 >o.oi *K^K*;* therefore *the more knowledge* BRIDGER *has, the better the quality of candidates it generates.* Similar results were observed for the lower, average, and upper ranges. In addition, the group of specification influences the results. The scaling values satisfy: *lower* <o.oi *average* <o.oi *upper,* whereas the quality values satisfy: *lower, average* <o.oi *upper.* This confirms a known engineering heuristic stating that it is relatively easy to design artifacts that are similar to past experience or slightly scaled down and harder if designs are to be scaled up. Since designs are generated for every specification, BRIDGER *does not exhibit a brittle behavior}*[12]

In terms of measurement theory, the fact that some measures were not different in a statistically significant manner suggests that the mapping between the observed and the formal system is inadequate since the differences in the hierarchies (e.g., different number of examples) were not mapped into different measures of performance. One reason for this inadequacy results from assuming that performance (and the value of knowledge) is proportional to the size of die hierarchies or the number of examples used to generate the hierarchies. The known power low of practice governing learning (Newell and Rosenbloom, 1981) suggests that performance (and the value of knowledge) varies as a power function of the number of examples.

Another MANOVA analysis was run with this model. The results of the *scaling* remained as before, but, the results of the *quality* were more conclusive: *K2* >o.oi *K$* >o.oi *K4* and *lower* <o.oi *average* <o.oi *upper.* This exercise further demonstrates: (1) the care in hypothesizing a performance measure and testing it; and (2) the difficulty in creating accurate quantitative measures.

[11] As mentioned before, the results of *K* are not analyzed, If, however, we introduced*! into the analysis, the results of the scaling would be the same, but those of the quality would change to no statistically significant differences. This is due to the large variability in the quality of bridges designed with *K*. This example demonstrates the importance of executing a careful statistical evaluation.

[12] Of course, if we start introducing new properties into bridge descriptions, BRIDGER would have to adjust to them. However, this is not expected to be difficult

S ^ T s t r ^ ^ o T S X measures, but it depends on the th^ry or on the specification of the mapping between the observed and the formal systems used to create it. S o r e s t rnusYbe acknowledged that statistical results are also subjective. By summarizing a sySperformance in statistics, this measure tends to hide detailed information tiiat can be usefu[1] in understanding the system behavior; nevertheless, additional detailed tests can be performed that reveal more performance characteristics.

6 DISCUSSION AND SUMMARY

Two definitions of knowledge have been offered, each leading to two types of qu^ty measures. The four measures were demonstrated in the evaluation of the design system BRIDOER. The demonstration has several limitations. For example, the 48 test specifications where not generated rSdomly and some of the measures were only partially executed. In essence, this demonstration is a single case study advocating for the benefits from an ability to assess the value of knowledge. Nevertheless, the demonstration conveys several important aspects of knowledge evaluation. First, none of the measures alone provides a complete insight of knowledge quality; rather, the measures complement each other. Second, the type of measure defined or model posited (e.g., knowledge utility or power law of practice) has a significant impact on the success of the evaluation. Often, fiSg gS m o d e l s is a significant research problem. Third, the design of data collection experiments, whether qualitative or quantitative, can lead to interesting observations (e.g., the division of the specification set into three groups).

Table3summarizes*ediscussion.The*«fl/i^^ ***structural*** **measure is subjective and incomplete.** Its use requires having substantial domain expertise. It can potentially predict the design behavior ofadesign system. It will be useful in systems with manageable size knowledge, otherwise it will be hard to execute. The *quantitative structural* measure is precise but detached from the mechanisms that manipulate knowledge. It may predict the system design performance if it is based on some of the system's mechanisms. Its execution requires the development of a formalism that will allow quantifying knowledge, this formalism will usually be system dependent The *qualitative functional* measure can predict the quantitative measure and confirm the structural measure. It is important for revealing subtle issues in the behavior of a design system that need to be addressed m future development The *quantitative functional* measure is precise but also subjective. Inaddition, it loses details due to its tendency to summarize performance by statistics.

Although we favor the functional definition of knowledge, this paper does not attempt to defend this position. However, the paper argues firmly that the definition adopted and the measure used should be consistent that is, the structural (functional) evaluation should be used for assessing the quality of knowledge if the structural (functional) definition of knowledge is adopted. In fact beside this consistency, it is optional and useful to exercise all the four measures. Each of them has some value and feedback to provide and they all complement one another.

Table 4 summarizes the different measures according to the terminology of measurement theory. A question mark denotes an unspecified entry. The two qualitative measures are incomplete since the mapping between knowledge to textual information is not quantifiable. The quantitative measures are complete but are not fully articulated. For example, the functional quantitative measure employs addition as the binary operation, but does not specify what exactly is measured by the real numbers. For instance, to satisfy the addition operation in the example, the knowledge unit should reflect the logarithm of the number of examples and the measure should evaluate the logarithm of the performance. Therefore, we argue that *not one of the measures is, or could be, "objective.*

Table 3: Measures of design knowledge

		Structural		Functional	
		Qualitative	**Quantitative**	**Qualitative**	**Quantitative**
1	Type	subjective imprecise coarse	subjective precise coarse	subjective imprecise detailed	subjective precise coarse
2	Require	domain expertise	**precise** **formalism**	expert decisions, test cases	performance metrics
3	Maybe used to	partially predict performance	partially predict performance	predict performance	quantify performance
4	**Apply on**	simple representation, manageable size knowledge	everything subject to row 2	everything including "black boxes-	everything subject to row 2
5	Summarized by	textual information	quantitative data	textual + quantitative information	concise quantitative data

Table 4: Observed and formal systems

	Observed	Formal			
		Structural		Functional	
		Qualitative	**Quantitative**	**Qualitative**	**Quantitative**
Set	**knowledge** hierarchies	text	positive real numbers	text+ numbers	**positive real** numbers
Relation	**better-than**	**?**	>	**?**	>
Operation	**appending**	**?**	+	[7]	+

Future work includes completing several important tasks* The first task is the collection of data on evaluations of additional (design) systems. These cases might be generalizable, thus, used to develop a methodology for the evaluation of (design) knowledge. Another task deals with the formation of guidelines for system development that will facilitate appropriate evaluation. Finally, research should address the evaluation of knowledge embedded in a design setting (i.e., cooperation between a human designer and its design assistant), thereby leading to models that reflect actual design settings.

ACKNOWLEDGMENTS

This work has been supported in part by the Engineering Design Research Center, a National Science Foundation Engineering Research Center.

REFERENCES

Adelman, L. (1991). Experiments, quasi-experiments, and case studies: a review of empirical methods for evaluating decision support systems. *IEEE Transactions on Systems, Man, and Cybernetics,* 21(2):293-301.

Balkany, A., Birmingham, W. B., and Tommelein, I. D. (1991). A knowledge-level analysis of several design tools. In Gero, I, editor, *Artificial Intelligence in Design[9]91, Proceedings of The First International Conference on Artificial Intelligence in Design, Edinburgh, UK,* pages 921-940, Oxford, UK. Butterworths.

Bijl, A. (1987). Strategies for CAD. In *Intelligent CAD Systems* /, pages 2-19. Springer-Verlag, Berlin.

Boulton, D. M. and Wallace, C. S. (1973). An information measure for hierarchical classification. *The Computer Journal,* 16(3):254-261.

Brown, D. C. and Spillance, M. B. (1991). An experimental evaluation of some design knowledge compilation mechanisms. In Gero, J., editor, *Artificial Intelligence in Design'91, Proceedings of The First International Conference on Artificial Intelligence in Design, Edinburgh, UK,* pages 323-336, Oxford, UK. Butterworths.

Clancey, W. J. (1988). Acquiring, representing, and evaluating a competence model of diagnostic strategy. In Chi, M. T. H., Glaser, R., and Farr, M. 1, editors, *The Nature of Expertise,* pages 345-420. Lawrence Erlgaum Associates, Hillsdale, N.J.

Cohen, P. R. and Howe, A. E. (1989). Toward AI research methodology: Three case studies in evaluation. *IEEE Transactions on Systems, Man, and Cybernetics,* SMC-19(3):634-646.

Coyne, R. D., Rosenman, M. A., Radford, A. D., Balachandran, M., and Gero, J. S. (1990). *Knowledge-Based Design Systems.* Addison-Wesley, Reading, MA.

Doyle, J. (1988). Implicit knowledge and rational representation. Technical Report CMU-CS-88-134, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA.

Dreyfus, H. L. (1979). *What Computers Can't Do—The Limits of Artificial Intelligence.* Harper Colophon Books, New York.

Ericsson, K. A. and Simon, H. A. (1980). Verbal reports as data. *Psychological Review,* 87(3):215-251.

Gaines, B. R. (1989). The quantification of knowledge-formal foundation for acquisition methodologies. In Ras, Z. W., editor, *Methodologies for Intelligent Systems, 4,* pages 137-149, New York. North-Holland.

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.

Genesereth, M. R. and Fikes, R. E. (1990). Knowledge interchange format, version 3.0 reference manual. Technical Report Logic-92-1, Department of Computer Science, Stanford University, Stanford, CA.

Gluck, M. and Corter, J. (1985). Information, uncertainty, and the utility of categories. In *Proceedings of the Seventh Annual Conference of the Cognitive Science Society, Irvine, CA,* pages 283-287, San Mateo, CA. Academic Press.

Green, C. J. and Keyes, M. M. (1987). Verification snd validation of expert systems. In *Proceedings of the Western Coherence on Expert Systems (WESTEX-87),* pages 38-43, Los Alamitos, CA. IEEE Computer Society Press.

Hays, W. L. (1988). *Statistics. Fourth edition.* Holt, Rinehart & Winston, New York.

Howe, A., Dixon, J., P., C, and M., S. (1986). Dominic: A domain-independent program for mechanical engineering design. *International Journal for Artificial Intelligence in Engineering,* l(l):23-29.

Kaplan, B. and Duchon, D. (1988). Combining qualitative and quantitative methods in information systems research: a case study. *MIS Quarterly,* 12(4):571-586.

Kononenko, I. and Bratko, I. (1991). Information-based evaluation criterion for classifier's performance. *Machine Learning,* 6(l):67-80.

Kreinovich, V. and Kumar, S. (1991). How to help intelligent systems with different uncertainty representations cooperate with each other. *Cybernetics and Systems,* 22(2):217-222.

Kyburg, H. E. J. (1988). Knowledge. In Lemmer, J. F. and Kanal, L. N., editors, *Uncertainty in Artificial Intelligence 2,* pages 263-272. North-Holland, Amsterdam.

Levesque, H. J. (1984). Foundations of a fucntional approach to knowledge representation. *Artificial Intelligence,* 23(2):155-212.

Mallery, J. C, Hurwitz, R., and Duffy, G. (1986). Hermeneutics: From textual explication to computer understanding. Technical Report A.I. Memo No. 871, Massachusetts Institute of Technology, Artificial Intelligence Laboratory, Cambridge, MA. Also apppeared in *The Encyclopedia of Artificial Intelligence,* John Wiley & Sonse, New York, 1987.

Michalski, R. S. (1986). Understanding the nature of learning: issues and research directions. In Michalski, R. S., Carbonell, J. G., and Mitchell, T. M., editors, *Machine Learning: An Artificial Intelligence Approach, Vol 2,* pages 3-41. Tioga Press, Palo Alto, CA.

Mueller-VoUmer, K., editor (1985). *The Hermeneutics Reader.* Continuum, New York, N.Y.

Nazareth, D.L. (1989). Issues in the verification of knowledge in rule-based systems. *International Journal of Man-Machine Studies,* 30(3):255-271.

Neches, R., Fikes, R., Finin, T, Gruber, T, Patil, R., Senator, T., and Swartout, W. R. (1991). Enabling technology for knowledge sharing. *AIMagazing,* pages 37-56.

NeweU, A. (1982). The knowledge level. *Artificial Intelligence,* 18(1):87-127.

Newell, A. and Rosenbloom, P. S. (1981). Mechanisms of skill acquisition and the power law of practice. In Anderson, J. R., editor, *Cognitive Skills and Their Acquisition.* Erlbaum Associates, Hillsdale, N.J.

Podolny, W. and ScaM, J. B. (1986). *Construction and Design of Cable-Stayed Bridges. Second edition.* John Wiley and Sons, New York.

Reich, Y. (1991a). *Building and Improving Design Systems: A Machine Learning Approach.* PhD thesis, Department of Civil Engineering, Carnegie Mellon University, Pittsburgh, PA. (Available as Technical Report EDRC 02-16-91).

Reich, Y. (1991b). Design knowledge acquisition: Task analysis and a partial implementation. *Knowledge Acquisition,* 3(3):237-254.

Reich, Y. (1992). The acquisition and utilization of basic aesthetic criteria in design. *Artificial Intelligence in Engineering,* (accepted for publication).

Searle, J. R. (1980). Minds, brains, and programs. *The Behavioral and Brain Sciences,* 3:417-424.

Shortliffe, E. H. (1976). *Computer-based Medical Cunsultation: MYCIN.* Elsevier, New York, NY.

Skyrms, B. (1990). The value of knowledge. In Savage, C. W., editor, *Minnesota Studies in the Philosophy of Science: Volume XIV: Scientific Theories,* pages 245-266. University of Minnesota Press, Minneapolis, MN.

Stevens, J. (1946). On the theory of scales and measurement *Science,* 103:677-680.

Zuse, H. and Bollmann, P. (1987). Software metrics: Using measurement theory to describe the properties and scales of static software complexity metrics. Technical Report RC 13504, IBM Research Division, Yorktown Heights, N.Y.