Project-Specific Knowledge Bases in the AEC Industry

by

H. Craig Howard

CIFE WORKING PAPER

Number 6

December, 1989

Stanford University
Project-Specific Knowledge Bases in the AEC Industry

H. Craig Howard, A. M. ASCE

Abstract

Project-specific knowledge is the rationale behind the project data and specifications, including the design decisions that link elements of basic data, design data, project specifications, domain knowledge, and general knowledge to explain the design. This capture and communication of this knowledge can improve the facility engineering process in downstream project activities, in lifecycle operation and facility modifications, and in future projects. The purpose of the paper is to provide a qualitative definition of project-specific knowledge and its potential uses across the project lifecycle. The paper also includes some anecdotal descriptions of useful project-specific knowledge and a discussion of the problems which must be solved in order to automate the capture and communication of this knowledge.

1. Introduction

During the phases of design, planning and constructing a facility, a substantial volume of knowledge and data about the facility and its components is generated, applied, and lost. The most common form of project data communication is the design drawing, which consists mostly of geometric shape and positioning information along with specifications for material types. Hence, in current computer-aided design environments, only the results (the outcome of the decisions) of design, planning, or construction (e.g., as-designed drawings, as-planned schedules, as-built drawings) are stored about the project, while no information is maintained about the objectives, constraints, and assumptions under which these decisions were made. This latter information is project-specific or design-dependent knowledge. This project-specific knowledge must be captured so that it can be used in downstream activities, such as verification of design standard conformance, redesign and facility management, and in future design activities using case-based reasoning.

Communication of all types of data and knowledge is a real problem in the architecture-engineering-construction (AEC) industry. The degree of vertical fragmentation (between project phases, e.g., planning, design, and construction) and horizontal fragmentation (between specialists at a given project phase, e.g., design) in the AEC industry is unparalleled in any other manufacturing sector [Howard 89a]. To further complicate matters, the industry is still exchanging data as it did a century ago with paper drawings and reports. The introduction of the computer to the design process has changed the means of generating the paper, but it has not fundamentally changed the methods of sharing data across organizational boundaries. Techniques for automated

---

1 Submitted to the Journal for Computing in Civil Engineering, November 1989.

2 Assistant Professor, Department of Civil Engineering, Stanford University, Stanford, CA 94305, USA; ARPA: hch@cive.stanford.edu.

3 Data and knowledge are defined in Section 2.1.
data exchange are being developed, e.g., data exchange standards like PDES (Product Data Exchange Specification) [PDES 88] and intelligent database interfaces [Howard 89b]. However, data is only one aspect of the overall problem—the what of the artifact, but not the why. The knowledge about project decisions and their supporting rationale may be even more important.

This paper explores the subject of project-specific knowledge in the AEC industry. The purpose is to provide a qualitative definition of project-specific knowledge and its potential uses across the project lifecycle. The paper also includes some anecdotal descriptions of useful project-specific knowledge and a discussion of the problems which must be solved in order to automate the capture and communication of this knowledge.

One caveat before proceeding: The term design is used frequently throughout this discussion. It is useful to remember that the process of design is used at many stages in the lifecycle ranging from the architect's early conceptual schemes, to engineering system organization, to the component detailing, to construction planning, to operating procedure specifications, etc. Therefore, the use of the word design in this paper is intended to encompass that larger meaning and to be limited to the two standard phases in Figure 1 labelled conceptual design and detailed design.

![Figure 1: Phases in the Architecture-Engineering-Construction-Facility Management Process](image-url)
2. Project-Specific Knowledge: Definitions

This section begins by defining a basis for talking about the differences between data and knowledge within the context of an object-oriented model. Then it examines project-specific knowledge within the larger spectrum of data and knowledge and make explicit the basic assumptions regarding project-specific knowledge that serve as the foundation for this discussion. Finally, it explores a deeper definition of project-specific knowledge through an investigation of the knowledge flows that are represented in design decision.

2.1. Data and Knowledge Within an Object-Oriented Model

This section defines data and knowledge in terms of an object-oriented model, in which the objects comprise the physical elements, processes, and concepts that describe the complete project. The objects are grouped into hierarchies* that define the characterization-generalization relationships. The leaf nodes of the hierarchies are object instances that relate to specific physical objects (beam 21, pump 88), processes (activity 27), or concepts (equipment reservation 44: backhoe 23 to be used in trench 12 on day 34) in a specific project.

Given this basis, data can be defined as information that takes the form:

- **form**
  - <attribute> of <object-instance> has <value>
  - <object-instance> has <relationship> to <object>

- **examples**
  - diameter of a #3 rebar = 0.375 inches
  - beam-3 is-supported-by column-1

The key idea in this definition that data are the values for the attributes and relationships of object-instances.

And knowledge can be defined as information that takes the form:

- **form**
  - if <pattern> then <action>
  - <attribute> of <object> has <default value>
  - <object> has <relationship> to <object>

- **examples**
  - if depth > 10 then resize column
  - default concrete strength = 3000 psi
  - beam is a structural element

The first knowledge form corresponds to the classic rule-based, or logic-based, definition of knowledge and can represent design heuristics as well as design constraints. The actions may be procedural (an algorithm) or related to the object structure (change an attribute value), affecting either data or knowledge. The last two forms correspond to the knowledge that is represented in the object hierarchies regarding default values and relationships. The object-oriented programming concept of methods can be represented using a specific version of the first knowledge form: "if <message> sent to <object> then perform <method>.”

These abstract definitions of data and knowledge are far to terse to be completely satisfactory. More of the subtleties in the definitions will become apparent in the following discussion. We will also need a term that encompasses both data and knowledge: information ≡ data + knowledge.

---

4 In the general case, these are intersecting hierarchies of objects.
2.2. Spectrum of Data and Knowledge

The first step in identifying the project-specific knowledge is to classify the elements in that range of data and knowledge as shown in Figure 2. The spectrum ranges from very concrete, tangible details (basic data shown in black) to very abstract, common principles (general knowledge shown in white). Our concern is for the gray area (design decisions and project specifications) between the extremes. We’ll examine each area in depth, working from the outside in (concentrating on how they have been captured and communicated currently).

Basic data is the data known before starting the project. This category may include descriptions of standard components, unit costs, site surveys, experimental results, etc. Basic data does change over time, but within the project view, basic data is fixed. For example, if a project generates a new standard component, that data description belongs in the design data category. Basic data is frequently represented physically in handbooks and catalogues, and electronically in databases.

General knowledge consists of the basic principles of science and problem-solving that are common to many domains, e.g., physics, chemistry, search, synthesis, etc. Computer science has thus far failed to capture any significant portion of general knowledge in a usable form, although this area of general problem solving was a starting point for what is now the field of artificial intelligence.

Design (derived) data is the data generated by the design process, included the final data that characterizes the design and the intermediate results of calculations. Design data includes:
• Attribute values for specific object instances (components, subsystems, and systems) in the design; examples: the depth of beam-3 is 20 inches, the type of frame-5 is moment-resisting.

• Attribute values for prototypical elements of the design; examples: the width of standard-beam-1 is 10 inches.

• Relationships between object instances; examples: column-1 supports beam-3; beam-3 is a standard-beam-1.

Design data is generated through design decisions made by humans or computers. Traditionally, design data was stored on paper in drawings and reports. Much of that data has migrated to digital form in CAD systems and databases, although paper is still the primary medium of data exchange.

**Domain knowledge** is the constraints and heuristics that belong to a profession or an individual professional. The collective knowledge of the profession is manifested as building codes, manuals of design practice, and textbooks. Current research in standards processing has produced prototype systems that can represent and manipulate the knowledge embodied by building codes for checking and design [Garrett 87, Elam 88]. The knowledge of the individual professional is synthesized from the profession's knowledge plus experience in previous projects. The rise of expert systems has provided a much needed tool for formally codifying the domain knowledge of the individual professional, but so far the scope of domain knowledge that can be successfully represented and applied in a single expert system remains very small and limited to nonsynthesis tasks.

Domain knowledge includes elements such as:

• **Constraints** — The spatial, functional, and resource constraints among the design components considered during the design process and their sources (e.g., design standards, construction techniques, other agents, owner, etc.).

• **Definitions** — The physical laws, geometrical relationships, and material properties that provide the equality relationships which supplement the design constraints in the definition of the solution space. Some definitions are domain knowledge (the reinforcement ratio for a concrete member), and some definitions are general knowledge (force equals the product of mass and acceleration).

• **Heuristics** — The experiential knowledge or “rules of thumb” that were used in pruning the design search space. Frequently, these rules serve as approximate evaluation criteria, eliminating alternatives that typically prove infeasible or focusing on alternatives that conform to standard design and construction practices.

As with basic data, domain knowledge changes over time, but is effectively constant for a typical project. Certainly, project-specific knowledge affects domain knowledge—the revision of building codes after dramatic structural collapses gives evidence to that phenomena—but that knowledge communication path is the exception rather than the rule, and therefore domain knowledge remains static during the design and construction phases of most projects.

**Project specifications** define the purpose, function, and performance of the facility being constructed and its major components. The generation of the specifications start in the planning phase of the project with gross requirements on the facility (e.g., square footage, number of floors) and generally continue through the design phases to constrain detailed performance.
characteristics of facility components (e.g., strength of structural connections, capacity of ventilation fans). The project specification is as much an output of design process as the design data. The specification may focus on the required performance of an element (e.g., strength) or may prescribe a specific item from a specific vendor (in which case, this part of the specification is more properly classified as design data).

Project specifications may be explicit (a document that accompanies the design) or implicit (knowledge from intermediate steps that is not explicitly written down). Explicit project specifications are typically recorded and transmitted in documents that accompany the design drawings; in many cases, parts of the specifications are noted on the drawings (usually a textual description linked to a drawing element with an arrow). Implicit project specifications may be part of the engineer's working notes, but they are not transmitted as part of the design.

**Design decisions** are the combinations of information (data and knowledge) that produce design data and project specifications. Starting at the highest level, the facility planner must decide on the gross characteristics of the proposed facility, making decisions such as required square footage based on the owner's needs, economic considerations, relevant building codes, and site conditions. At the most detailed level, the engineer decides to put a specific size of reinforcing bar in a specific column based on computed loadings, code requirements, and cost.

This design decision model in Figure 2 includes two kinds of information flow for design decisions. The input for a design decision may include: basic data (costs for reinforcing bars), design/derived data (computed load on a column), project specifications (required capacity of a pump), domain knowledge (building code limits on allowable stresses), and general knowledge (gravity acts downward). Design data (size of reinforcing bar) and project specifications (required strength of connections) are the typical results of design decisions; however, a design decision can change or supersede a previous design decision and the knowledge about the changed status of that previous decision should be recorded.

If the ingredient and resulting information are represented by in our object model, then the design decisions can be represented as new objects that link existing information objects to new or modified information objects. Figure 3 illustrates a simple example of the project-specific knowledge embodied design decisions. The example concerns the design of a rectangular steel beam that must satisfy a designer-imposed constraint on depth and a code-based constraint on capacity (combined from several equalities and inequalities to simplify this example). The first decision determines the depth of the beam. The designer examines the applicable constraints and the evaluation criteria, and then chooses the maximum allowable depth because the depth has the greatest influence on the code constraint. However, the decision object is more just linkages between data and knowledge objects. The clause "because the depth ..." from the sentence above must be captured in the design decision object to complete its documentation.

In the second decision, the designer considers the strength of the steel and the constructibility constraint along with the other indicated elements to choose the smallest value of width that satisfies the code constraint. The attribute area is assigned as a byproduct.

This example has been greatly simplified to produce an object network that fits on a page in a readable typeface. Documenting a realistic design will require a very large and complex network of the objects. The difficulties in capturing, representing, and applying these project-specific knowledge networks are addressed in Section 4.
**Figure 3: Design Decision Example**
(Circles and rounded rectangles denote object instances; arrows leading into a decision object denote the ingredients of a decision; arrows leading out of a decision object denote the results of a decision.)

In general a design decision is the result of a composite information flow that includes ingredients from several of the six categories. Listed below are several examples of composite information flows that can be treated as aggregate quantities.

- **Intentions** — the intended functionality of each system component (such as a rigid-frame or a laterally supported beam) and the intended functional relationships between components (such as a ventilation duct being supported by the structural frame). Intentions involve an intersection of the domain knowledge about basic functions, the project specifications (explicit and implicit), and the design data to produce new project specifications (usually implicit). Recent research on design representations suggests that an explicit object-oriented formulation of the design description in terms of form, function, and behavior is important to effective computerization of the design task [Luth 89].

*Draft: January 1989*
• **Assumptions** — the designer’s hypotheses about system behavior, construction methods, future uses, etc. (e.g., “the most likely cause of failure for column 1 is buckling”). Assumptions are implicit project specifications that require the application of domain knowledge (including experience) along with design data.

• **Evaluation criteria** — the basis for a choice among satisfactory alternatives. The choice of a specific design alternative involves the minimization of the cost/value ratio [Luth 90]. Cost and value may be determined with simple equations (domain knowledge) and basic data, or they may be approximated by qualitative assessments from domain knowledge (e.g., “value” of redundancy in a structural system). The quantitative and qualitative criteria are important ingredients in the explanation of design choices. Here is an example of useful evaluation criteria knowledge in a design decision: A designer chooses a larger steel beam than is structural required because that larger beam is cheaper due to local availability; if that beam deteriorates and must be replaced, then a future engineer may safely choose a smaller steel beam if cost factors have changed.

• **Alternatives** — other feasible or infeasible designs that were considered and rejected during the design process including the reasons why they were rejected (e.g., unsatisfied constraints, higher cost, lower value). An alternative is represented by a combination of design data and project specifications linked to a design decision that explains why that alternative was not chosen.

• **Design Plan** — the sequence of steps in the design process including the various levels of abstraction through which the design progressed and the alternatives considered at each level of abstraction. The plan includes all design steps, even those that which represent decisions that eventually retracted as alternatives are proven infeasible or inefficient. The plan is a design decision that results in other design decisions. To explicit make the design plan for the example in Figure 3, there would be another decision object which represents the intersection of information to represent the design plan: “first select depth, then select width.”

2.3. **Assumptions about Project Specific Knowledge**

Before further defining project-specific knowledge, it is important to make explicit several assumptions that underlie this exploration:

• Project-specific knowledge is generated during the planning, design, construction, and operations phases.

• Very little of that project-specific knowledge is captured in current reports, specifications, drawings, CAD systems, and electronic databases.

• Much of the currently uncaptured project-specific knowledge would be useful at some later point in the execution of the current project and would also be useful in future projects.

• It is possible to capture and store that project-specific knowledge through an intelligent user interface in an electronic form that is both machine-readable and machine-usable. The distinction between machine-readable and machine-usable is significant. Textual descriptions are machine-readable, but until computers can understand natural language, the only computer use for that text is to regurgitate it in response to a keyword search or other indexed access. The connections between input (knowledge and data) and output (again, knowledge and data) must be explicit and justified.
• It is possible to provide useful access to that stored project-specific knowledge at later stages of the project through an intelligent user interface.

The first three assumptions originated from an intuitive analysis of the AEC process and have been confirmed in our observations to date. The last two assumptions have yet to be proven—given evolving computer technology, the objectives should be obtainable, but we don’t yet know exactly how to do it. Section 4 presents a few ideas about how to transform these assumptions into reality.

3. Uses of Project-Specific Knowledge

Assuming that the project-specific knowledge contained in the implicit project specifications and design decisions can be captured and transmitted, the next question is how might it be applied. The potential uses can be grouped into three broad categories:

1. In downstream tasks in the current project — Project-specific knowledge can improve sequential design processes (e.g., where the output of one designer is the input for another) and communication across engineering project phases (e.g., from design to construction).

2. Across the life cycle of the constructed facility — The acquisition of project-specific knowledge should not stop with the completion of the project; it should be an ongoing activity throughout the life of the facility. The experience gathered in the management and operation of the facility is vital to future maintainers and to future designers of similar facilities.

3. In future projects — Project-specific knowledge about multiple projects forms the basis for that quality called “experience” that is so frequently relied upon by senior designers and construction planners. Knowledge about previous solutions, plans, methods, successes, failures, and operating problems can be reused throughout future projects to improve the speed of the design process and the quality of the new facility.

3.1. Downstream Project Activities

The term “downstream” refers to any facility project activity where previous decisions are important to the current tasks. Throughout the architecture, engineering, and construction, automatic transfer of project-specific knowledge can make significant contributions in several areas such as design verification, redesign, error reduction, optimization, and construction operations:

• Design verification — Verification is becoming increasingly more important in today’s practice in design and construction. The intense level of design checking and validation in the nuclear industry is a likely harbinger of things to come for the entire industry. Project-specific knowledge can make a critical difference in the verification of the design; i.e., when checking to see that all the various code constraints are satisfied. Knowledge of the intended functionality is required to verify the conformance of a design to standards, which often refer to the function when specifying required conditions. Looking at only the final design results, intended functionality is frequently hard to determine.

This need for representation of design intent in standards processing is illustrated in work done by Garrett on SPEX [Garrett 87] and Elam on SICAD [Elam 88]. Both of these component design/checking systems need to know whether a particular structural component was designed as a beam or a column in order to know what part of the
voluminous design standard to consider. In addition, many design standards permit several different methods of analysis and design to be performed (usually a conservative, simpler method and a less conservative, more complicated method). One would need to know which method was chosen so that conformance with a standard is properly evaluated. When one moves away from component checking and into the real problem of checking an entire structure for conformance, it is vital to have more information than just the final results. For example, knowing how components were grouped in the design process would be extremely useful in evaluating conformance.

- **Redesign** — All too frequently, design conditions change, with the result that components, assemblies, and entire systems must be partially or completely redesigned. Even if the same designer is responsible for making the changes, a complete record of the initial design process may speed the alterations and greatly reduce the likelihood of error. Project-specific knowledge elements for redesign can include previously rejected alternatives that may now be more attractive, interaction constraints with other systems that might easily be missed during hasty changes, and design plans that may be restarted from an intermediate point without having to retrack the entire process.

- **Error reduction in subsequent design tasks** — Serious errors can be introduced in the design by those who do not understand the original intentions and assumptions of a component or system, either through unseen interactions or through inappropriate changes to the original design. Even when the original designer is called upon to check subsequent steps or modifications, dangerous problems can go undetected (several examples are given in Section 5) Project-specific knowledge bases can communicate all of the design considerations to subsequent designers. This improved communication has the potential to prevent downstream errors and to facilitate the review of changes with respect to the original design considerations.

- **Optimization in subsequent design tasks** — The design process is both parallel and sequential. The output of one designer is the input of another. Sometimes the sequential flow results in suboptimal decisions being made in downstream activities. For instance, if a structural beam were just slightly shallower, a standard duct size could be used at a substantial savings. The mechanical engineer who follows the structural engineer usually doesn't have the expertise to know what can be changed from the previous design step, nor does he/she know the relative cost tradeoffs in the previous problem versus the current problem. These tradeoffs are normally resolved by direct communications between the two designers, if at all. Project-specific knowledge can be important aid to the second designer in determining when to pursue possible changes in a previous design step; it would not replace the direct interaction, but would focus that interaction on those instances where a real savings is possible.

- **Construction** — Designs contain spatial and functional errors that are revealed during construction: ducts and beams that shouldn't intersect, but do; pipes that should intersect at a joint, but don't; connections that don't fit; members that don't support construction loads; etc. A good first step in correcting an error is to understand why it occurred in the first place—accessing the original rationale can lead to possible corrective alternatives and can prevent ill-advised fixes. Sometimes the choice of a construction method is implicit in the system or component selected, and that knowledge needs to be carried along with the design as more than a footnote on a drawing. A proper understanding of the designer's functional characterization of the system can prevent mistakes based on wrong assumptions during assembly.

Hence, the usefulness in capturing and storing project-specific knowledge is that it permits the re-examination of upstream decisions by the same, or other, agents in the design process. This is a
critical capability when the decisions made upstream cause unanticipated downstream problems in design, construction, and operations.

3.2. Facility Life Cycle

Forward thinking owners often insist on as-built drawings to get a paper record of the completed facility. Some of those owners are looking ahead to the realization of the information age and requesting digital versions of this data (Stanford's own Facility Projects Management Office is beginning to do just that [Tyson 89]). A few facility managers are even working to capture textual explanations of the design via computers in large projects such as NASA's new spacestation and the Boston Harbor cleanup.

Knowledge from the design phase can support facility management by communicating the designer's intentions and assumptions, which may or may not match with the operator's expectations. For example, if the designer intended a certain room to be part of a residential unit and designed the floor to support code-mandated distributed loads, then the future use of that room as a classroom might result in dangerous overloading. The design knowledge base can help the owner/operator to determine how the evolving use of the facility is served by the original design. When the facility needs repair due to damage or wear, the knowledge of the original specifications provides a basis for designing and safely executing repairs as well as in the recognition of hazardous conditions. Finally, when a facility must be expanded or retrofitted to accommodate growth or new technologies, the project-specific knowledge can be reused for redesign and construction operations as described above for downstream activities.

Within the operation phase, the capture of project-specific knowledge in the form of operating procedures and troubleshooting heuristics is an important addition to a "facility" memory that will outlive staffing and even ownership transfers. Right now, many owners are hurrying to capture the expertise of retiring engineers who have been responsible for keeping major facilities running smoothly. For instance, a California utility company is working to capture the knowledge of a senior dam engineer before he retires and leaves their hydroelectric network to his relatively inexperienced replacements. Furthermore, project-specific knowledge captured during facility operation can provide significant feedback for new, improved designs resulting in facilities that are cheaper to operate and maintain.

3.3. Future Projects

Recent knowledge-based expert systems for structural engineering design have focused on design-independent knowledge (abstract reasoning rules for designing), and while great strides have been made in that area, there is still a significant need to develop systems to take advantage of the wealth of knowledge contained in every substantial facility project. On the other hand, previous database-oriented design efforts have focused primarily on knowledge-poor databases of solutions, in which the traditional engineering handbook of solutions has simply been replaced by digital data.

Experienced designers do not design strictly by abstract reasoning processes, nor do they exhaustively search a space of previous design solutions, testing whether each exactly matches the current design criteria. Because they have typically performed many similar design tasks to reach a level that we would term "experienced", these expert designers have been exposed to a wide variety of design problems and the reasoning processes associated with those design problems. Therefore, the experienced design professional has a memory of good (and bad) designs and knows the rationale behind each of them. Potentially, the designer may have generalized some of this experience into abstract reasoning rules or associations (frames), but most of the experience is
still in the form of project-specific knowledge—knowledge about specific previous designs and their supporting reasoning.

The best approach for applying project-specific knowledge to new design problems combines design-independent reasoning with project-specific (case-based) reasoning into an integrated knowledge-based system (see Figure 4) [Howard 89b, Wang 88]. The integrated design system should function as a design aid, not as a black box designer. Therefore, the integrated system would interact with the designer during the design task, functioning both as a design assistant and as a knowledge acquisition system by recording the designer's steps and rationale. In this way the design system becomes a true apprentice to the experienced designer, progressively learning to solve more and more difficult design problems. It can also be used to tutor novice engineers by replaying cases from its memory, providing experience that might otherwise take much longer to acquire.

A few small prototypes have been already implemented for the architecture-engineering-construction domain. Among them are:

- STRUPLE [Zhao 88] is a prototype system that uses previous design solutions to identify the relevant design elements for a structural design synthesizer.
- RESTCOMM (REdesign of STructural COMponent) [Rafiq 88] is a Prolog-based research prototype for experimenting with similarity and matching in a small case base of reinforced concrete beams.
- FIRST [Daube 88] redesigns structural beams by accessing a case memory of solution plans.

![Diagram of Integrated Knowledge-Based Design System]

**Figure 4:** Overview of Integrated Knowledge-Based Design System [Wang 88]
4. Challenges in Building and Applying Project-Specific Knowledge Bases

Let us return the last two assumptions of Section 2.3. How is it possible to capture and store project-specific knowledge? And how is it possible to provide useful access to that stored knowledge? The challenges in applying project-specific knowledge are in the areas of acquisition and use (see Figure 5). Supporting project-specific knowledge will require a new types of design tools and user interfaces. The following sections explore the requirements on that software.

4.1. Acquiring Project-Specific Knowledge

The initial research in this area has frequently elicited responses of the form: "The designer doesn't want to stop what he/she is doing to turn to the computer screen and document or justify the actions." However, as noted in section 3.2, the external demand for this knowledge is such that some designers are being required to do just that. The problem is that textual representation of design knowledge are machine-readable, but not machine usable. The project-specific knowledge base should consist of explicit connections between explicit knowledge objects.

The key underlying concept is that the designer should not have to turn to separate knowledge capture interface to document the design—the expression of project-specific knowledge should be a natural outgrowth of the design process. The design user interface must make the explanation (the why) a natural outgrowth of the decision action (the what). All of the design tasks should be conducted using this single interface. Work on user interfaces for design databases suggests that the interface should be an integrated, multi-modal system in which the designer has several different communications mechanisms for describing design actions [Howard 88]. The communications modes should include different types of graphics and textual commands to provide the designer with maximum expressive power.

Certainly such a design user interface poses many challenges in human-computer interaction. A further question is how smart must the interface be to fully capture the knowledge behind the
In its basic form, the user interface would serve as a knowledge network editor, permitting the designer to specify the linkages between knowledge and data that constitute design decisions. On the other hand, it is possible that the user interface must be smart enough to serve as an apprentice to the designer—anticipating design actions, trying to understand choices, and making the designer providing sufficient justification that the interface “understands” the design decisions.

The problem of acquiring knowledge can be partitioned with an eye to the intended use of the knowledge. There are three sources for the project-specific knowledge:

- The **designer**, whose specific knowledge should be carried through the rest of the project and into the operation of the facility;

- The **expert designer**, whose knowledge can be captured regarding significant case studies for use in case-based reasoning (a small number of important cases can be more useful and certainly more manageable than a large, unbalanced case base); and

- The **computer**, which will assume more and more of the design responsibility as computer-aided design systems become more and more intelligent.

### 4.2. Using Project-Specific Knowledge

In exploring the uses of project-specific knowledge, we need to complete the matrix in Table 1, showing the degree to which the different types of project-specific knowledge can positively influence the potential application areas. A qualitative assessment of the intersections of the knowledge and its uses will be helpful in identifying the most promising starting points for project-specific knowledge research in the architecture, engineering, construction, and operation of facilities.

An important part of making the project-specific knowledge useful is providing the right kind of user interface to locate and display that knowledge. A *knowledge browser* would need the same multi-modal communications capabilities required in the design user interface to provide flexible, powerful methods for describing the kind of information that the user is seeking.

<table>
<thead>
<tr>
<th>Uses of Project-Specific Knowledge</th>
<th>Project-Specific Knowledge Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Intentions</td>
</tr>
<tr>
<td>Downstream</td>
<td></td>
</tr>
<tr>
<td>Verification</td>
<td></td>
</tr>
<tr>
<td>Redesign</td>
<td></td>
</tr>
<tr>
<td>Error reduction</td>
<td></td>
</tr>
<tr>
<td>Optimization</td>
<td></td>
</tr>
<tr>
<td>Construction</td>
<td></td>
</tr>
<tr>
<td>Lifecycle</td>
<td></td>
</tr>
<tr>
<td>Operation</td>
<td></td>
</tr>
<tr>
<td>Rehab and retrofit</td>
<td></td>
</tr>
<tr>
<td>Future Designs</td>
<td></td>
</tr>
<tr>
<td>Case-based design</td>
<td></td>
</tr>
<tr>
<td>Lifecycle feedback</td>
<td></td>
</tr>
</tbody>
</table>

**Table 1: Potential Applications of Project-Specific Knowledge**

Draft: January 1989
5. Anecdotal Examples of Useful Project-Specific Knowledge

Listed below are four examples drawn from actual projects where project-specific knowledge needed to be communicated. Only in the final example was the project-specific knowledge communicated successfully, and that case involves direct human intervention rather than traditional project data flow. The cases refer back to Table 1 to identify potential impacts of project-specific knowledge.

1. The 1983 collapse of a California ice arena is a classic example of a failure to maintain a building based on the designer’s assumptions. The roof of the ice arena collapsed under the weight of accumulated snow. The elegant design for a snow melting system had worked well for 18 years until a fiber-filled coating was applied to the roof to correct leaks. This well-meaning maintenance effort dramatically increased the coefficient of friction between snow and the corrugated steel deck, rendering the melting mechanism ineffective.

**Missing project specific knowledge:** the design decisions linking the snow melting function in the project specification to the roof design data.

<table>
<thead>
<tr>
<th>Intentions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rehab and retrofit</td>
</tr>
</tbody>
</table>

2. A 12-year-old multi-story concrete and timber office building is located in a seismic hazard zone. After recent earthquakes, the owner ordered a seismic evaluation of all buildings on the site. A key element to that evaluation is the understanding of the original designer's structural systems for resisting lateral loads due to earthquakes. However, the design plans fail to indicate such significant details as whether critical connections were intended to be moment-resisting. The building must be reverse-engineered to determine how the code-mandated requirements for strength were satisfied.

**Missing project specific knowledge:** the design decisions linking the connection function in the project specification to the connection design data.

<table>
<thead>
<tr>
<th>Intentions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verification</td>
</tr>
</tbody>
</table>

3. The 1981 collapse of the skywalks at the Kansas City Hyatt Regency is an example of a failure that resulted from the fabricator's lack of understanding regarding the original design intent of the hangers. The fabricator's connection changes resulted in the serious weakening of an already understrength connection. Even the designer forgot the original intent of the connection detail in approving the changes, with catastrophic results.

**Missing project specific knowledge:** the design decisions linking the connection function to the hangar design data.

<table>
<thead>
<tr>
<th>Intentions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error Reduction</td>
</tr>
</tbody>
</table>

---

5 Cases 1 and 3 were identified as part of [Rafiq 90].
4. Construction was underway for a multi-level reinforced concrete structure in an seismic hazard zone. When mixing the concrete, the contractor had the option of using higher strength, higher density aggregate for the same cost as the aggregate specified in the design. Reasoning that stronger concrete is always better concrete, the contractor was prepared to use heavier aggregate when the project engineer intervened. The engineer explained that weight of the aggregate was a critical factor in the dynamic response of this structure. The aggregate specified in the design had been chosen based on a detailed dynamic analysis, and a heavier aggregate would have caused unacceptable stresses during an earthquake.

**Missing project specific knowledge:** the design decisions linking dynamic analysis derived data and the applicable stress constraints to the aggregate density design data.

| Error Reduction | \( \sqrt{\text{Intentions}} \) |

6. Conclusion

The life cycle for civil engineering facilities—such as nuclear power plants, buildings, dams, and bridges—are measured in decades (and sometimes centuries) during which time staff changes are guaranteed to occur. With the increasing complexity of today's buildings and industrial plants, the time from initial planning through final construction can also be lengthy (especially for mega-projects like the English Channel tunnel). If question arise about design decisions, it is likely that the original engineer making those decisions will either be unavailable or will have forgotten all but the highest level details and their justifications. Hence, the representation of project-specific knowledge is not a luxury, but a necessity for civil engineering projects, in order to provide a "project memory." From the combinations of many project memories will emerge generalizations and associations that we commonly refer to as *design experience*.

**Acknowledgements**

My thanks to Jenmu Wang, Taufiq Rafiq, Ana Cristina Bicharra Garcia, Douglas Phan, and James Garrett for their input at the discussions in which the ideas described in this paper were crystallized. This work has been partially supported by grants from the Center for Integrated Facility Engineering at Stanford University and the National Science Foundation.

**References**


*Draft: January 1989*


