Predicte—An Intelligent System for Indicative Construction Time Estimation

Geoff Stevens, Alan Stretton, Michael S. Register, Steven M. Medoff, Mark W Swartwout, and Magnolia Fung

The project early design-stage indicative construction time estimate (Predicte) system is an expert system that is designed to provide indicative construction time estimates of concrete-framed, multistory building projects at the concept or early design stages when relatively little project information is available. Given a preliminary design concept for a multistory building, Predicte returns a listing of the major construction activities required to build the building, the activity duration times, and their start and end dates. In this chapter, we describe the construction time-estimation process and our automation of this process. We discuss the implementation of Predicte and how it was developed, tested, and deployed. We also discuss the maintenance of the system and its business payoff.

The Predicte system was jointly developed by the Lend Lease Corporation of Sydney, Australia; the AI Applications Group of Digital Equipment Corporation; and the Digital Sydney AI Centre. Initial contact between Lend Lease and Digital began in early 1985, and formal development of Predicte occurred from July 1985 to October 1987.

Today, Predicte is accessible from 12 different Lend Lease locations throughout Australia.

Predicte is one of the first expert systems developed for the construction industry (see Feigenbaum, McCorduck, and Nii [1988] for a list of expert system applications in the construction industry). It is also the first expert system developed for construction time-estimation tasks. Our work in developing Predicte helped us define a new class of expert systems, which we call design-verification and design-evaluation systems. We begin this chapter by describing the time-estimation process and why the application of expert system technology matches well with the problem and with Lend Lease's business needs.

Problem Description

Predicte was developed for Lend Lease's construction subsidiary, Civil and Civic. Customers frequently come to Civil and Civic with an initial proposal for a building project and seek advice on the feasibility of the project and an estimate of how long it will take to complete. This initial estimation process is difficult; in fact, for the Australian construction industry, the initial time estimate is off by 47 percent on the average (estimators tend to be optimistic [Bromilow and Henderson 1977]).

To arrive at an accurate time estimate, the estimator must possess a large amount of knowledge about the construction industry and construction practice. The estimator needs to know how to translate an initial building concept into basic design information and then must be able to apply this knowledge of construction methods, resources, and planning to arrive at an accurate time estimate. Civil and Civic has such a person, an expert with over 30 years of experience in the Australian construction industry.

There are several advantages to applying expert system technology to this problem. Because the expert is near retirement, Civil and Civic can capture and document his knowledge before he leaves. By capturing this knowledge, Civil and Civic can maintain this competitive advantage. Civil and Civic can also distribute this knowledge to its offices throughout Australia. Once the expert retires, other construction estimators must be able to update the knowledge base as new information about construction techniques, building materials, and so on, becomes available. Expert system technology allows for much easier maintenance of a knowledge base than does conventional software technology. In addition, through automation, the estimation process is much faster than the expert can be with a pad and pencil. Also, if changes to the preliminary design are made, a time estimate can be recalculated faster.

With an explanation facility (a feature easily implemented by using an expert system approach), Civil and Civic can show its customers how an estimate was calculated and what assumptions were made in deriving the estimate. Also, non-experts can learn about the estimation process faster by using an expert system that contains an explanation facility.

The estimation process itself is one of design analysis. The first part involves verifying that the proposed building design meets certain structural and functional requirements. Some of these requirements are simple, such as assuring that the total area of the building is not greater than the site area on which it is being built. More complex checking confirms that portions of the design are compatible or are safely combined. If a requirement is not met, then changes must be made to the design. The time estimator must be able to quickly identify which parts of the design need to be changed. The time estimator then presents these data to the customer and provides some information about why the data failed to meet a particular requirement and some assistance in changing the data to meet the building requirements. The client makes a change to the design, and the verification process continues until the design meets all the building requirements.

The second part of the estimation process involves deriving a time estimate. In this part, knowledge of construction techniques and processes is used to estimate the time and resources required. The time estimator must make estimates on information outside the building design, such as weather trends and the availability of human resources, to derive an estimate of the total time needed to construct the building. The time estimator must also devise the most likely sequence of construction events. This sequence of events greatly affects the total time calculation because some construction activities can be carried out in parallel.

The estimator calculates a time estimate by breaking the project down into a set of major activities (for example, clearing of the site, ground excavation, footings installation). The estimator then determines the likely sequence of activities and uses this sequence to determine how much time each of the individual activities adds to the total project time. Each of the major activities has subtasks whose duration must be evaluated, and these subtasks can, in turn, depend on the evaluation of their own subtasks. Eventually, a level of tasks is reached that depends only on design input information for its evaluation. A task might have several methods of calculation depending on the particular design. For example, the height of the building, the number of basement levels, and the ground material determine the type of footings that are appropriate. Once this determination is made, the method for calculating footing time is selected and used. The final result of the estimation process is a listing of what the major activities are, how long each takes, and when they start and finish.

Because this point is the beginning of the project development stage, it is normal to explore multiple design alternatives before making a final decision on whether to proceed with the project. An expert estimator is able to recognize certain opportunities for time savings and will suggest design alternatives.

Construction time estimation belongs to a class of problems that we call *design-verification and design-evaluation problems.* The design-verification and design-evaluation process begins with a design description as input to the verification module. The *verification module* identifies those portions of the design that need to be altered and interacts with the user to produce a revised design that meets verification requirements. The evaluation module analyzes the revised design. This analysis could include an evaluation of various factors, such as functional performance of the design; cost to build; or, in the case of our application, time to build. The user (in our case, the customer) of such a process then has the option of accepting the design and its evaluation or altering the design to improve the evaluation.

Predicte Architecture and Function

In this section, we discuss the major modules of Predicte and describe most of their features. We start with the *main control module* . This module is responsible for performing design verification and calculating a time estimate. It takes design data from the user and the verification and estimation knowledge base as input. The main control module returns a list of the major activities required for building the given design, the activity start and end dates, and their durations. Figure 1 is a sample activity report.

The verification and estimation knowledge base contains three main types of knowledge. All the necessary design information and other related information are represented as questions that are asked of the user. A large amount of auxiliary information can be associated with a question; for example, graphic sketches can be attached to a question or set of questions to highlight the piece of design information requested. There are 223 input questions in Predicte. Of these, approximately 100 to 140 questions are asked of the user in a normal consultation session.

All the verification requirements in Predicte are represented as constraints between input and derived data. A verification constraint can

Activity	Start	Duration	Complete
Site Establishment	Day 0	12 days	Day 12
Bulk Excavation	Day 12	68 days	Day 80
Footings	Day 80	27 days	Day 107
Basement Outside Tow	Day 107	203 days	Day 310
Podium Structure	Day 310	80 days	Day 390
Podium Facade	Day 425	34 days	Day 459
Pod. Services, Finish	Day 390	240 days	Day 630
Core F.W. Lead Time	Day 0	81 days	Day 81
Tower Basement Struct	Day 107	203 days	Day 310
Lobby Structure	Day 310	67 days	Day 377
Tower Structure	Day 377	287 days	Day 664
Start in lift shaft	Day 446		
LMR Installation	Day 664	121 days	Day 785
Basement Plantrooms	Day 377	92 days	Day 469
Roof Plantroom	Day 664	121 days	Day 785
Facade Lead Time	Day 0	115 days	Day 115
Tower Facade	Day 425	271 days	Day 696
Services, Wet Trades	Day 377	383 days	Day 760
Ceilings, Finishes	Day 521	335 days	Day 856
Furnishing, Equipment	Day 569	287 days	Day 856
Paving, Landscaping	Day 664	72 days	Day 736

Indicative time to opening day = 38.0 months.

Figure 1. Predicte Activity Report.

place a range restriction on an input or derived datum, or it can specify a relationship between data that must always hold. Verification constraints act like demons, ready to fire as soon as they are violated. When a violated constraint fires, the main control module invokes the constraint-resolution mechanism. There are 63 verification constraints in Predicte.

Knowledge of how estimates are calculated from the input data is represented in the form of estimation rules. The rules are grouped into sets; each set is responsible for calculating a value for a particular derived datum. The estimation rules are used by a goal-driven control strategy. The main control module begins with a list of the major construction activities. It then locates the rule sets that are used to calculate these activity times. These rule sets will require more derived data to calculate values, so the main control module finds the rule sets for these derived data. This process continues until the level of input data

```
Buildings with a free-standing height to width
ratio exceeding 6:1 are feasible, but they
incur a high structural cost penalty.

The following data may be causing the problem:
(Q17) Overall width of tower plan = 6 m
(Q18) Overall depth of tower plan = 35 m
(Q52) Lobby structure height = 6 m
(Q78) Typical floor height = 3 m
(Q96) Total number of tower levels above
the lobby structure = 15

Please review this information and enter the
question number you wish to change>_
Available options: Pause, Guide, Help.
```

Figure 2. Constraint-Resolution Behavior.

is reached. There are 1275 estimation rules in Predicte.

When a verification constraint is violated, the constraint-resolution mechanism is invoked. The *constraint-resolution mechanism* takes the data involved in the constraint and traces back to the input data that caused the violation. The constraint-resolution mechanism then presents the contradictory input data to the user and provides some advice on how to resolve the constraint or why the constraint is necessary (this advice is declared with the constraint in the knowledge base). The user is then forced to change the values for the conflicting data. An example of this behavior is shown in figure 2. Once the user has changed the conflicting input values, control is returned to the main control module, and an estimate is calculated.

Once the main control module returns a list of activity times to the user, the user has the option of running the analysis and opportunities module on the estimation results. The analysis and opportunities module is essentially a redesign component that looks at the estimation results, identifies areas in which unusual amounts of time are being spent, and suggests changes to the building design to eliminate the unusual amounts of time. The user has the option of accepting one or any combination of the suggested changes. The analysis and opportunities module then reinvokes the main control module to recalculate an estimate based on the altered design. The analysis and opportunities module uses an analysis and opportunities knowledge base composed of redesign rules. Each rule has a left-hand side that identifies a potential time problem and a right-hand side that suggests a design

```
   Explaining Bulk Excavation Time
When:
   * Excavation perimeters below adjacent foundations
     (Q 191) is > 0
   * Likely ground material is medium strength (Q 206)
   * Average depth of excavation (AVG-DEPTH) is > 0
   * The arrangement is a Tower with basement and
     podium, the tower accessible from the street (Q 14)
Then:
   Bulk Excavation time (ET = 68.3 days) is
   (MAXIMUM E4 (SHORING + (E5 + E7))) days
Where:
   E4            = time needed for underpinning = 54 days
                   (calculated)
   SHORING       = Elapsed time for shoring = 7 days
                   (Predicte estimate)
   E5            = time to excavate volume accessible by ramp
                   = 47 days (calculated)
   E7            = time to remove excavation ramp = 14.3 days
                   (calculated)
   Explain>_
   Available options:  Guide, Activitylist, More,
                       Previous, Set, Quit, Why.
```

Figure 3. Explanation Example.

change to correct the problem.

A user can ask how an activity time was calculated, and Predicte's explanation module will respond with the rule that was used to calculate the time. To make the explanation more readable, some screening of the left- and right-hand sides of the rule is done. Figure 3 shows a sample explanation for the major activity "Bulk Excavation." Left-hand side conditions are listed in the "when" section, the right-hand side formula used to compute a value for the activity is shown in the "then" section, and explanations of what the variables represent are given in the "where" section.

On the left-hand side of the rule being explained, there are several conditions that could have been satisfied in various ways. For example, the general arrangement of the tower could have been one of four different tower arrangements, and the rule would still fire. However, in the explanation, the user is only told about the tower arrangement that was selected and not about the three other tower arrangements that would have yielded the same result. The rationale behind this approach is that the user only wants to see the relationship between the proposed building design and the estimate Predicte calculated. This rationale is even more pronounced when Predicte explains a rule that contains a left-hand–side condition that tests for the absence of a certain design feature. In this case, the entire condition is eliminated from the explanation.

In the example on the right-hand side of the rule being explained, the original formula that was used is

(MAXIMUM E4 (SHORING + (E5 + (E6 + E7)))) .

In this formula, the variable E6 represents "the time to excavate volume inaccessible by ramp." However, in the explanation, E6 is not mentioned in the formula because in this case, E6 is equal to zero. Because E6 does not add anything to the total time, it is excluded from the explanation. The user does not want to see an activity that adds nothing to the total time; it is the same as saying that this activity will not take place in constructing this building.

From the explanation prompt, the user can also see a domain-dependent, first principles justification of why this rule was applicable for the given design. This justification can include graphic sketches to demonstrate the principles behind the rule.

Because Predicte is dealing with incomplete and preliminary information, Predicte has to make some intermediate estimates, such as the available human resources at a given work location. It would be preferable if the user knew this information, but the user generally does not. In those cases in which the user does know, the user is allowed to override these estimates with a more accurate figure. Only selected derived data can be overridden (otherwise the user could override everything and corrupt the final estimate). These data are flagged in the explanations Predicte generates as a " Predicte estimate." In the explanation module, the user can override a Predicte estimate with a different value, and Predicte will invoke the main control module to recalculate the results based on the altered value.

Because exploring design alternatives is a large part of the initial project evaluation process, Predicte provides a facility for easily modifying the design input data and recalculating an estimate based on these data. Predicte also provides a case-comparison facility that allows the user to compare the estimates of two separate cases. This case-comparison facility allows the user to obtain a quick overview of the major differences between two possible alternatives (in terms of time to build).

Predicte provides a variety of reporting mechanisms. In addition to seeing the time estimates in table format, a user can get a bar-chart time-line of the project displayed graphically on a terminal or

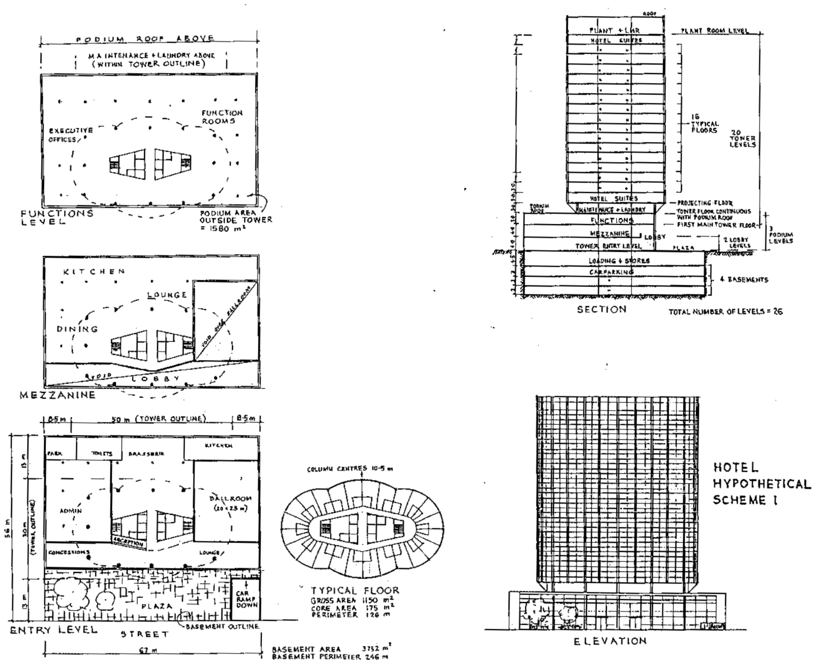

Figure 4. Building Sketches for Initial Hotel Design.

printed on a hard-copy plotter.

Predicte also provides a facility for excluding weather delay estimates in the total estimate. Each major activity has an associated weather delay based on the construction site location and the time of year in which the activity is projected to take place. The user can override specific individual delays or can eliminate all weather delays entirely.

An Example Case

To further illustrate some of Predicte's function, we present a hypothetical case using Predicte. Because of client confidentiality, we cannot present an actual case, but this hypothetical case was designed to be close to a real building project.

The building design that is being evaluated is a high-rise hotel. Initial building sketches for the hotel are presented in figure 4. This information is typically all the input that a time estimator has from which to derive an estimate. The first part of the user's interaction with Predicte is the question-asking phase. In this case, 108 of Predicte's 223

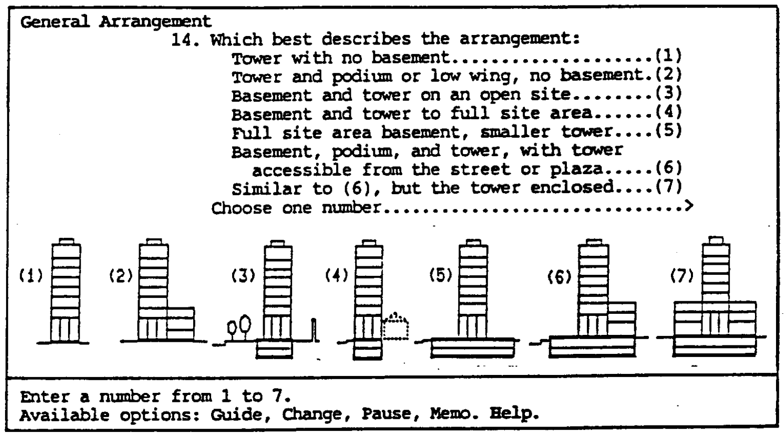

Figure 5. Sample Predicte Question.

questions are asked because these answers are the only input needed to derive an estimate for this particular project. The questions are grouped into general categories called *subheadings.* These subheadings are useful to the user when the user wants to explore alternative designs and needs to quickly locate the input information that needs to be altered. Some of the subheadings include general project information (for example, building location, building function), general arrangement of building, tower floor plan, basement plan, lobby structure, tower levels, tower plantrooms, nontypical tower floors, podium, facade, site dimensions, existing services (for example, utilities, rail tunnels), and ground conditions. One of the questions asked in this case is shown in figure 5.

During the question-asking phase, Predicte is continuously verifying that the input information satisfies its verification constraints. As soon as a violated constraint is identified, question asking is interrupted, and the constraint-resolution mechanism takes over; the user is then expected to resolve the violated constraint before question asking can resume. In this case, we assume that no constraint is violated.

Once question asking is finished, Predicte starts to calculate a time estimate using its knowledge base of estimation rules. The ultimate goal is to calculate times for the major construction activities for this project, but along the way, Predicte needs to calculate some intermediate estimates. The two most important intermediate items that Predicte needs to derive are the availability of human resources and the most likely sequence of construction activities. The availability of human resources depends mostly on where (in Australia) the building is being

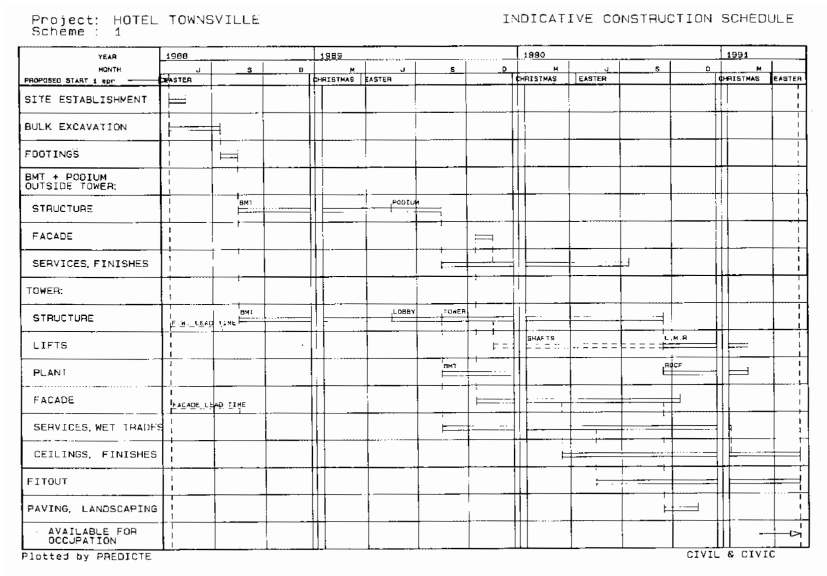

Figure 6. Bar-Chart Display of Activity Times Listed in Figure 1.

built. In this case, the hotel is planned to be built in Townsville, Queensland, one of Australia's smaller cities. Because of Townsville's size, Predicte will estimate a lower figure for a key group of human resources (25 people). This figure can be changed later, as we see.

Predicte has 11 stored construction sequences. Each sequence determines which activities will occur, what the order of activities is, which activities can proceed in parallel, and what the allocation of human resources is for parallel activities. Predicte uses information such as the amount of human resources available and the type of building (that is, does it have a basement or a podium, does the basement or podium extend outside the tower area) to determine the most appropriate sequence of activities for the particular project.

Once a sequence of activities is determined, Predicte calculates the times for the major construction activities. Figure 1 shows the activity times for the hotel design we are evaluating. These times can also be graphically displayed in a bar-chart format (Figure 6). In this form, it is easier to see how long the activities are taking in relation to each other and which activities seem to be taking an unusually long time.

At this point, the user has a variety of options for exploring design alternatives in the hopes of decreasing the time estimate. The user can get explanations of how the times were calculated specifically for those activities that seem to be taking too long. These explanations will help

1. Structural progress may be limited by the local availability of resources. It may be possible to import a larger crew from interstate and reduce the construction time, although the subcontract labor cost may then be increased. Would you like to rerun using an imported structural crew?

2. Additional time will be needed for transfer beams to bridge over a column free function room within the tower area. Would you like to rerun without a wide column free room within the tower lobby area to avoid this problem?

3. The addition of projecting floors has added to the time needed for the tower structure. Would you like to rerun using a different arrangement for the projecting floors?

Figure 7. Predicte's Suggested Changes to Hotel Project.

the user understand how the design can be modified to decrease the time estimate. Also, the explanations will identify any intermediate Predicte estimates that can be overridden if the user wishes. This option is described in some detail in the previous section.

Another simpler option is to use Predicte's analysis and opportunities module. This module automatically identifies problems with the design and suggests changes that will decrease the time estimate. Figure 7 shows three changes that this module has suggested for this particular case. The first change is not a design change, but it does suggest an option to the user for decreasing the time estimate. By bringing in a second crew to the work site from another area of the country (remember, because of the building location we could only obtain a limited number of workers locally), the total time for building the hotel decreases from 38 to 30 months. However, as Predicte indicates, this reduction can result in a higher overall labor cost, so the client needs to decide whether keeping the labor cost down or keeping the time to build down is more important.

The second and third changes do propose a design revision. The user will typically go back to the original building sketches and see if a change can be made that will accommodate the change Predicte is proposing. In this case, by turning the tower 90 deg, we can eliminate the need for transfer beams over the hotel ballroom and the need for projecting tower floors at the same time. The revised building sketch is shown in figure 8. By making this design change and importing the second work crew, we decrease the time estimate to 26 months. The final activity report for the revised design is shown in figure 9. Notice that this activity report also reports that the resource estimate Predicte

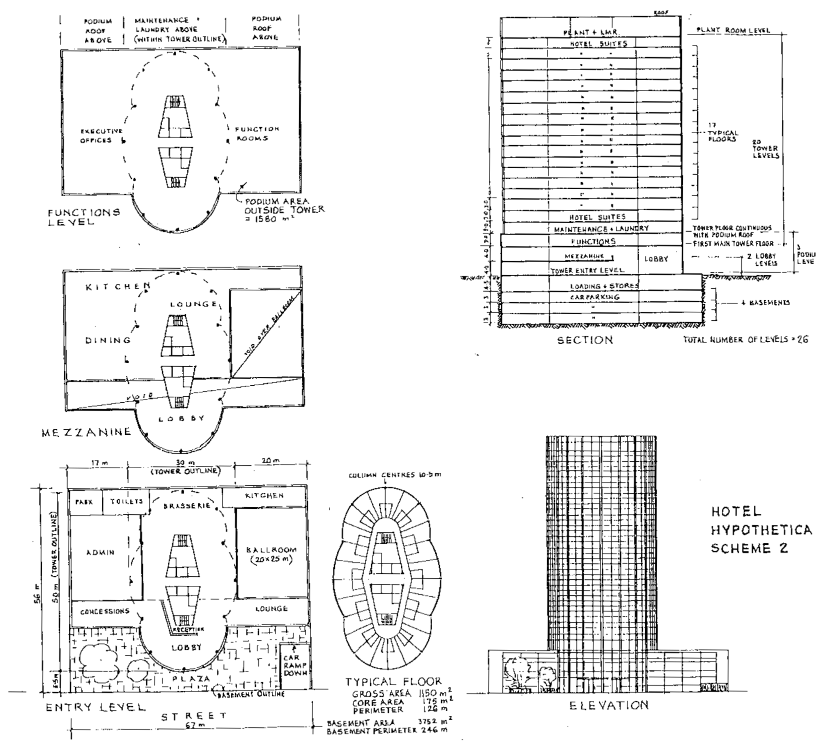

Figure 8. Building Sketches for Revised Hotel Design.

made was overridden. Thus, in this case, the analysis and opportunities module has identified a simple design change and a simple project resource change that saves a considerable amount of time over the estimate for the original building design.

Implementation

Predicte is a stand-alone system that runs on any VAX/VMS system. For implementing Predicte and the general class of design-verification and design-evaluation expert systems, a language called Candle was developed. Candle provides a knowledge engineer with an easy-to-use syntax for specifying input questions, verification constraints, evaluation rules, and redesign rules. Candle is built on a constraint-propagation network language called CNL. CNL is a modified version of the constraint-propagation language described by Steele (1980). Both Candle and CNL are written in Vax Lisp, Digital Equipment Corporation's im-

Activity	Start	Duration	Complete
Site Establishment	Day 0	12 days	Day 12
Bulk Excavation	Day 12	54 days	Day 66
Footings	Day 66	27 days	Day 93
Basement Outside Tow	Day 93	123 days	Day 216
Podium Structure	Day 216	80 days	Day 296
Podium Facade	Day 295	50 days	Day 345
Pod. Services, Finish	Day 295	190 days	Day 485
Core F.W. Lead Time	Day 0	81 days	Day 81
Tower Basement Struct	Day 93	123 days	Day 216
Lobby Structure	Day 216	33 days	Day 249
Tower Structure	Day 249	190 days	Day 439
Start in lift shaft	Day 295		
LMR Installation	Day 439	121 days	Day 560
Basement Plantrooms	Day 249	92 days	Day 341
Roof Plantroom	Day 439	121 days	Day 560
Facade Lead Time	Day 0	115 days	Day 115
Tower Facade	Day 295	166 days	Day 461
Services, Wet Trades	Day 249	262 days	Day 511
Ceilings, Finishes	Day 357	226 days	Day 583
Furnishing, Equipment	Day 393	190 days	Day 583
Paving, Landscaping	Day 439	72 days	Day 511

Indicative time to opening day = 26.0 months.

The following Predicte estimates have been overridden:

Maximum formwork resources likely to be available locally:
User overridden = 50 people
Predicte estimate = 25 people

Figure 9. Predicte Activity Report for Revised Design.

plementation of Common Lisp (Steele 1984).

Constraint-propagation networks were used as the foundation for the representation scheme for two main reasons. First, constraint networks provided dependency structures that allowed for dependency-directed backtracking algorithms to be used for the efficient retraction of input data. Because of the frequency of design changes, the representation system needed to be able to retract data and recalculate results in an efficient manner.

Having dependency structures also allowed for easy tracing back to the input data when a verification constraint was violated. Many verification constraints involved derived data; when they were violated, the dependency structures were used to trace back to the input that caused the violation. For more information on Candle and CNL, the reader is referred to Register (1986) and Medoff, Register, and Swartwout (1989).

Development of Predicte

Knowledge acquisition for Predicte was easier than with the typical expert system. The expert was able to write down on paper the entire estimation process. The knowledge engineers' task was one of knowledge debugging rather than knowledge acquisition.

Because of the vast geographic distance separating the expert from the developers, some communication processes needed to be put in place. There were two main means of communication. More formal communications occurred at the project management meetings that were held quarterly. For day-to-day informal communications, an electronic mail link was established between Australia and the United States. This approach worked fine for a while, but toward the end of the project, the cultural and professional differences between the building industry in Australia and the computer industry in the United States proved too great. More face-to-face meetings were required for accurate understanding.

The development of Predicte followed an iterative cycle similar to the cycle discussed in Buchanan et al. (1983). The first six months of the project were spent investigating potential knowledge representation schemes (by the knowledge engineers) and developing the knowledge base (by the expert). After this period, the system was developed and tested in stages, called *base levels*. There were four base levels in all, each with expanded knowledge and increased function over the preceding level. Testing occurred throughout the process; the expert had a library of prior cases that were used to periodically validate the knowledge base. The total cost of developing Predicte (which includes developers' and expert's time) was 12 person-years.

Testing, Deployment, and Maintenance

Once development ended in October 1987, five months of testing the complete system began. Two main types of testing were performed. The first type involved testing the behavior of the system. The expert generated approximately 2500 hypothetical building project cases to ensure that the knowledge base was behaving as it was intended.

The second type of testing involved comparing Predicte-generated results to the actual results for 10 recently constructed projects. The testing included a detailed analysis and comparison of the actual and predicted activity times, and the reasoning behind the results was compared to make sure that Predicte's reasoning and justifications were sound. Daily construction site diaries were used to make these comparisons.

A successful deployment effort began in April 1988. Three main goals were used to judge whether the deployment of Predicte was successful. The first goal was that Predicte be used on all potential concrete-framed multistory building projects. This goal was achieved in April 1988. The second deployment goal was that Predicte be made available to all Civil and Civic's offices throughout Australia. This goal was accomplished in April 1988 by installing a tested and debugged version of Predicte on a single machine at a central location and allowing other offices remote access to the system. The last goal was the identification and training of an initial set of users. A group of users from several different Civil and Civic offices throughout Australia was identified and trained on Predicte. Several training sessions were given, the last one in May 1989.

Since October 1987, maintenance of the knowledge base has been exclusively performed by the expert with no assistance (save some initial training). We believe this is a positive reflection on the design of Candle. The types of modifications the expert has made to the knowledge base are bug fixes, the tuning of building constants used in rules to refine and improve estimation results, and the incorporation of new knowledge about new construction methods. Over the course of the last two years, 98 rules were added, deleted, or modified (approximately 7.7 percent of the knowledge base).

Most of the maintenance to Candle has involved fixing bugs and some cosmetic changes to the user interface. Maintenance of Candle has been performed at the Digital Sydney AI Centre; the engineers there have written a maintenance guide for assisting an engineer who knows little about Candle.

Business Payoff and Conclusion

It is difficult to precisely quantify the business payback of Predicte to Civil and Civic. Although Predicte can help modify building projects to save time and money, there are too many people and criteria involved in the decision-making process to claim that Predicte is directly responsible for such savings. Predicte also significantly cuts down on the time to prepare an estimate; however, the amount of time saved is small compared to the magnitude of the projects Predicte is used to evaluate. Predicte's major benefit to Civil and Civic is reliability and uniformity of time estimation throughout its organization. This benefit leads directly to improved customer satisfaction (that is, no surprises when work starts), and it is the customer satisfaction that is extremely difficult to quantify and is important in the construction business.

Predicte is used to evaluate potential projects of a large magnitude. In a nine-month period, Predicte evaluated building proposals ranging in value from 10 to 220 million Australian dollars. From the size of these proposals, it is easy to see that the task Predicte performs is critical to Civil and Civic's business.

Predicte also provides some advantages over the way Civil and Civic was previously performing time estimates. With Predicte, all Civil and Civic's time estimators distributed throughout Australia can perform at or near the expert level. With Predicte, Civil and Civic captured and preserved the knowledge of a key expert before he left the company. With Predicte, alternative building schemes can be explored much faster and with greater depth than they could have been by manually recalculating an estimate.

One of the biggest benefits of having Predicte is that all the reasoning and justifications for calculating an estimate are easily available to the user. This ability actually goes beyond even most experts in this field. Typically, software systems are seen as black boxes that cannot justify their recommendations. However, in this case, it is the human expert who is the black box, and the software system is seen as transparent, with the ability to justify its conclusions.

Predicte represents a major breakthrough in the construction industry because of its use of expert system technology to perform indicative time estimation. The development of Predicte has also led to the definition of a class of expert systems, called design-verification and design-evaluation systems, and the invention of a knowledge representation system designed to address this class of systems.

References

Bromilow, F., and Henderson J. 1977. *Procedures for Reckoning and Valuing the Performance of Building Contracts* . Melbourne, Australia: CSIRO Division of Building Research.

Buchanan, B.; Barstow, D.; Bechtal, R.; Bennett, J.; Clancey, W.; Kulikowski, C.; Mitchell, T.; and Waterman, D. 1983. Constructing an Expert System. In *Building Expert Systems,* 127–167. Reading, Mass.: Addison-Wesley.

Feigenbaum, E.; McCorduck, P.; and Nii, P. 1988. *The Rise of the Expert Company* . New York: Times.

Medoff, S.; Register, M.; and Swartwout, M. 1989. A Framework for Design Verification and Evaluation Systems. *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing* 3(2): 71–84.

Register, M. 1986. Candle: A Representation Language Combining Rules, Constraints, and Dependencies. Presented at the First Australian Artificial Intelligence Congress, Melbourne, Australia.

Steele, G. 1984. *Common Lisp: The Language*. Burlington, Mass.: Digital.

Steele, G. 1980. The Definition and Implementation of a Computer Programming Language Based on Constraints, Technical Report, AI-TR-595, Artificial Intelligence Lab., Massachusetts Institute of Technology.