1991

User models in DFI :

Stephanie J. Wagaman
Lehigh University

User Models in DFI
A Cooperative Problem-Solving System

by
Stephanie J. Wagaman

A Thesis
Presented to the Graduate Committee
of Lehigh University
in Candidacy for the Degree of
Master of Science
in
Computer Science

Lehigh University
1991

This thesis is accepted and approved in partial fulfillment of the requirements for the degree of Master of Science in Computer Science.

Advisor Donald J. Hillman Date May 1, 1991

Dr. Donald J. Hillman

Chairman Lawrence J. Varnerin Date May 2, 1991

Dr. Lawrence J. Varnerin

Acknowledgments

I'd like to thank my advisor, Dr. Donald J. Hillman, and also Dr. John L. Wilson for their advice and guidance in this endeavor. Thanks also to Marc Barone and Keith Werkman, former DFI'ers, who helped me to understand a system that at first seemed incomprehensible. There are many others who participated at one time or another in our research meetings - Dr. Richard Sause, Nam-Hee Hong, Bill Waid, and especially Stephen Wong.

Special thanks to my husband Bill, whose encouragement, support, and understanding helped me to see this through. Others encouraged me as well, my sisters Jackie and Ceilie, and dear friend Cindy, all of whom are an inspiration to me. Many thanks to my good friends at Lehigh, especially to Mona, for the encouragement and advice.

Finally, this research would not have been possible without the funding provided by the National Science Foundation's Center for Advanced Technology for Large Structural Systems (ATLSS). This funding is gratefully acknowledged.

Table of Contents

List of Figures

Abstract

The focus of my research is on user models, and the different ways they can enhance knowledge-based system performance and credibility. It is an area that integrates both artificial intelligence (AI) and database technology. How to use this information in reasoning to improve the performance of the system falls under AI. How to maintain models efficiently so as not to degrade system performance falls under database technology.

Simply stated, a user model contains information about users. Its use has become important as computer systems have expanded to include a wide variety of users as well as the performance of a larger number of tasks. It now becomes necessary for the knowledge based system to have a model of the user in order to respond correctly. In early computer systems, the burden of getting the machine to operate correctly was placed on the user. Emphasis now is on making computers easier to use - putting more of this burden on the software itself. The success of the software is more likely to be measured by its user interface, not solely by its performance.

User models are widely used in a variety of systems such as information retrieval, database systems, knowledge-based systems, adaptable user interfaces, and within cooperative problem-solving systems. The use of models within a cooperative problem-solving system, Designer Fabricator Interpreter (DFI), is explored. Emphasis will be on how user modeling can assist in cooperative problem-solving.

1. Introduction

User modeling has become an important topic as systems are designed for a wide variety of users. This chapter begins with a historical overview of user modeling. It then discusses the contents of a user model, the contents of a discourse model, and differences between generic and individual user models. Different methods of building user models are discussed including implicit and explicit acquisition and the use of stereotypes. The goals of user models are discussed, followed by a review of models in intelligent tutoring systems, information retrieval systems, expert systems, adaptive interfaces, and cooperative problem-solving systems.

1.1 Historical Overview

Simply stated, a user model contains information about users. It has become important as the range of people using computers becomes more diverse, as well as the range of problems computers are asked to solve. In early computer systems, the burden of getting the machine to operate correctly was placed on the user, and its success was measured by whether or not it solved the problem. Today, emphasis is on making computers easier to use, putting more of this burden on the software itself. The success of the system now is more likely to be measured by its user interface, rather than solely by its performance.

The concept of user modeling began with intelligent tutoring systems (ITS). For an ITS to respond appropriately as a tutor, it must know what the student knows, what he doesn't know, and what he knows incorrectly [Rich 79a]. This concept soon spread to other systems as well. In expert systems, a user model can aid in explanation or even in decision-making. Models can assist interactive retrieval systems and database systems by guiding the retrievals with knowledge it has about the user's goals. In fact, any system that is cooperative can benefit from a user model.

1.2 Goals of User Modeling

The goals of user models within knowledge based systems can be categorized in three broad areas: 1) improve system effectiveness, 2) improve system efficiency and 3) enhance system acceptability [Sparck Jones 86].

Effectiveness refers to the correctness or reliability of the decisions being reached by the system. Systems that involve user models directly in decision-making will enhance system effectiveness. Systems with user models that only enhance explanation text are not concerned with the contributions of user models to effectiveness.

Efficiency refers to the economy in which decisions are made, i.e. economy in searching and in checking consistency. In this case, user models do not influence decision-making directly, but rather are used as heuristics to guide or narrow the search.

Acceptability refers to the way the system is viewed by others and is an area where user models contribute the most. The inferred plans and goals of users influence the content of system dialogue and explanation. Many systems tailor explanations to the user's level of expertise, or phrase text in a way that is consistent with inferred goals.

1.3 Contents of User Models

The contents of user models will naturally vary with the application domain. But generally speaking, there are four categories to classify the knowledge contained in a user model: goals and plans, capabilities, attitudes, and knowledge or belief [Finin 88]. These categories are briefly described here.

A user's goal is the state of affairs he wishes to achieve, while his plan is the sequence of actions that will result in achieving his goal. Goals can be either short-term and related to the topic of discourse, or they can be long-term overall goals. A cooperative system can help the user achieve his goals by first inferring what they are.

A user's capabilities usually refers to his mental capabilities or his level of expertise. This area is extremely important in intelligent tutoring systems as it helps the system to infer the student's particular strategy in problem-solving. This strategy is then used when attempting to explain the student's errors. The level of expertise is also an important area in generating expert system explanation, which can be tailored to the user.

The third category of knowledge is that of the user's biases, preferences, and attitudes. These are often short-term characteristics that represent the user's attitude towards the topic of discourse. Also included in this category is the user's perspective or point of view on a specific object.

The last category represents knowledge and belief. This covers the user's knowledge about the domain, about the general world, and in a cooperative system, knowledge about the other agents. Domain and even real-world knowledge can usually be stated explicitly and programmed into the system. Knowledge about other agents, however, is implicit and is built as dialog among agents progresses.

1.4 Discourse Models

The discourse model can be viewed as either part of the user model, or separate but related to the user model [Morik 88/Cohen 88a]. In any respect, the discourse model contains the information about the conversation or the dialog-dependent knowledge. This can include an indication of the structure of the discourse, the attentional state, the intentional structure, and an organization of the objects mentioned in the discourse [Cohen 88a]. The attentional state represents the objects being discussed. The intentional structure refers to the immediate goals of the user regarding the object of discussion. Here is where the short-term goals are represented and updated as a dialog continues.

The discourse model is dynamic and short-term, i.e. it terminates at the end of the session. It is also used to lend relevance to dialog with the user. In a cooperative problem-solving system, it can enhance dialog among agents and aid in the problem-solving activity. Agents can also learn about other agent's goals through the discourse model.

1.5 Generic vs. Individual Users

A system that contains a single model of a generic user can have that model permanently imbedded within it [Rich 79a]. A generic user model assumes a homogeneous set of users, i.e. all users are similar enough with respect to the application that they can be treated as one type. A system that employs a single generic user is ZOG, a frame-based system that facilitates user-computer communication [Robertson et al 81]. Natural language systems also maintain a single generic model. One benefit in choosing this type of model is its quick response time.

There is, however, a limit to the usefulness of a generic model when a system has a wide variety of users. An improvement over the generic user would be a set of fixed generic models which represent different classes of users. Many tutoring systems employ this method. For example, the UNIX Consultant (UC) system classifies its users into four classes - novice, beginner, intermediate, or expert [Chin 88a].

Alternatively, individual user models contain specific information about a single user. A system designed around individual user models will have a separate model for each user of the system. These individual user models must be built implicitly, by extracting information from the user's actions. These systems must also deal with the issues of conflict resolution, the corrective actions necessary when two pieces of conflicting data reside in the model.

One way of combining knowledge about classes of users with knowledge of individuals is through the use of stereotypes. A stereotype is a collection of frequently-occurring characteristics or attributes that describe groups of users [Rich 79a]. A model is initialized with data belonging to a particular stereotype, and is updated as new knowledge is inferred. Rich's GRUNDY system and Morik's Real-Estate Advisor both use stereotypes to enhance their individual models [Rich 79a/Morik & Rollinger 85]. The use of stereotypes is discussed in more detail in the next section.

1.6 Stereotypes

When implicit modeling is not practical, explicit modeling using a priori knowledge is employed. One method of building such an a priori model is with the use of stereotypes. As mentioned earlier, stereotypes are a collection of characteristics that describe groups of system users. It is a mechanism by which individual user models can be built on the basis of a small amount of information, but can be used to predict a large number of facts.

A system using stereotypes must also have a set of triggers, or events which activate the appropriate stereotype. Once activated, the predictions a stereotype contains are incorporated into the user model.

GRUNDY is a system which recommends novels that may be of interest to the user based on a model it has built of the user [Rich 79a]. This model is built using the self-descriptive words that the user has supplied, combined with a priori stereotypes. Within

GRUNDY, a stereotype contains a set of attributes, values and ratings. The attribute (also called a facet) is a collection of characteristics. The value is the description of the trait, and the rating is a certainty factor or confidence level that a facet should be associated with a particular stereotype. When triggered, the attribute-value-rating triple becomes part of the user model, along with the justification. The justification, which serves as a pointer to the trigger which activated this piece of stereotypical information, is necessary in conflict resolution. All triggers in GRUNDY contain a rating representing the probability that the stereotype is actually appropriate for that particular situation.

Since user modeling with stereotypes is often times based on guesses made by inductive reasoning or probabilistic inferences, incorrect inferences can occur. This means that a new piece of information which is to be added to the model conflicts with data already in the model. These conflicts must be resolved to preserve the integrity of the model.

There are two different conflicting situations possible in GRUNDY. One is concerned with conflicts between whole stereotypes, and another is concerned with conflicts over the value of an individual facet. An example of a conflict between stereotypes is when both male and female stereotypes are triggered. In resolving the conflict, the system must decide which one to keep active by examining which trigger had a higher confidence value. The second example involves conflicts in individual facets. The most common situation is when a general stereotype conflicts with a more specific one. In this case, the more specific stereotype overrides the general information. When conflicts occur between two specific facets, and neither of them are facts (which override inferences), then the certainty values are enlisted again to resolve the conflict.

1.7 Implicit vs. Explicit Acquisition

Acquisition refers to the technique by which facts are learned about the user. These facts can be acquired explicitly or implicitly. Explicit knowledge comes directly from the system designer in the form of stereotypes, from the application system, or by direct query of the user [Finin 88]. These explicit methods leave much of the responsibility in the hands of the user, especially those that query the user directly and base their model solely on the responses given. Two systems that query the user are the Real Estate Advisor and UMFE

[Morik & Rollinger 85/Sleeman 85a).

Acquiring knowledge about the user implicitly is much more difficult. It involves observing the user's behavior and inferring facts about his goal and plans, implying that most of the information contained in the model will be guesses. Thus, the system must have some way of representing its certainty that a fact is true and subsequently have methods of performing maintenance on the model [Rich 79a].

Maintenance involves incorporating new learned knowledge into an existing model. If it is consistent with the knowledge in the current model, it is simply added to it. If it is inconsistent with knowledge in the current model, the inconsistency must be resolved. Two approaches to conflict resolution are evidential reasoning and default reasoning [Finin 88]. Evidential reasoning uses certainty factors that measure to what degree a fact is believed to be true. Default reasoning is where certain facts are held as true in the absence of evidence to the contrary.

1.8 Review of Related Work

1.8.1 Intelligent Tutoring Systems

An intelligent tutoring system, or ITS, is a computer program that instructs the student in an intelligent manner [Van Lehn 88]. Many ITS's achieve this with an inferred model of the student's understanding of the subject matter. Inferring the model is often referred to as diagnosis, or the process that manipulates the model. The input for diagnosis is taken from interaction with the student such as answers to questions, commands issued, or keys depressed. The student model can be used in a variety of ways

If the ITS uses a structured curriculum, a student model is consulted before advancing to the next topic. In this application, the student model represents the student's level of mastery. The model is also used to determine when to offer advice, and what form the advice takes. In explaining a concept to the student, the system must know what the student knows. Lastly, some ITS's generate problems dynamically as the tutoring session progresses. The student model is consulted in order to supply a reasonable problem which is beyond the student's current capabilities.

Most ITS's work by employing an expert model as well as a student model, the student model being the expert model plus a collection of differences. There can be two kinds of differences between them, missing conceptions and misconceptions [Van Lehn 88]. A missing conception (used the most) is an item of knowledge that the expert has and the student does not. A misconception is an incorrect item that the student has and the expert does not.

The student model is then compared to the expert model. There are three techniques which can be used to relate one to the other: overlay modeling, differential modeling, and perturbation modeling [Van Lehn 88]. Overlay modeling is the simplest technique and assumes that all differences between the student's and expert's behavior can be explained by lack of knowledge on the part of the student. This overlay technique was used in WUSOR II, an advisor for the game "Hunt the Wumpus." [Goldstein 82] This particular type of modeling works well for WUSOR II since the type of knowledge it contains is at the surface. Overlay modeling does not work well when the expert model contains deep knowledge. Another drawback of overlay modeling is that there is no provision for knowledge of a student that differs from the expert's.

Differential modeling is a modification to the overlay model. Instead of comparing the knowledge of the student with that of the expert, the performance of the student is compared with the expert's performance in the same situation. Thus the skills necessary in order to perform a task are compared. WEST, an electronic boardgame, uses a differential model to tell when a player has made an optimal move [Burton & Brown 82b]. It has the same drawbacks as overlay modeling, assuming student knowledge to be only a subset of the expert.

Perturbation models are more advanced than either overlay or differential models in that they can represent student beliefs beyond the range of the expert model. A common method of doing this is by augmenting the expert knowledge with likely misconceptions or "bugs" which the students might have. The DEBUGGY system uses a procedural network which represents correct methods of performing subtraction [Brown 82a]. Attached to this network is a set of procedures representing common mistakes which students make. Other systems accomplish this task via a bug library. It introduces the possibility that an error may be from an incorrect skill, rather than from a missing skill.

1.8.2 Information Retrieval Systems

Recently user modeling has been applied to the field of document retrieval. An interface, which identifies relevant aspects of the user's background and goals, serves to mediate between the user and the database. Daniels has identified five functions of a user model which aid in the document retrieval process [Daniels 86].

The first function of the user model is to determine the status of the user. Within a university, this status often is defined by either academic or non-academic. These categories can be further subdivided, i.e. academic can be either staff or student.

A second function is to determine the user's goals which are specific to his search. Daniels has identified the following search goals:

- Current search goals which indicate what type of out the user requires.
- Goals leading to search which could include reviewing the literature or filling gaps in knowledge.
- Specific goal such as thesis preparation.
- General goal such as personal or professional advancement

A third function of the user model is to determine the user's state of knowledge in the field, such as novice or intermediate. Another aspect to the state of knowledge is the decomposition of a topic into component parts. Once decomposed, it is determined which subdivision will be evaluated.

A fourth user model function is to determine the user's familiarity with information retrieval systems. This will be either none or some. If the user has no previous experience, the system may want to offer some instructions. If the user indicates that he has some experience, further details are elicited regarding the search procedures.

The fifth function of the user model within document retrieval is to determine the background of the user. The background can be broken into four relevant subdivisions: employment, residence, academic, background, and specific details like the budget available for the search. This data about the user may contribute to the information interaction.

1.8.3 Expert Systems

Much work has been done in the area of expert system explanation and user modeling. Producing explanations for an expert system is a complex task and recently has been considered separate from the problem-solving activity. Previously, explanations have been derived from the execution trace, or the path of inference steps from the data to the conclusion. [Wick & Thompson 89] cite several reasons why the explanation and reasoning should be decoupled.

It has been noted that human experts, when accounting for complex reasoning, rarely do so exclusively in terms of the actual process used to solve the problem. Often times things that were considered along the way are omitted in the explanation. Sometimes evidence not used in the problem-solving activity is introduced as additional support. The execution trace is often an indirect path. Many things that are evaluated by the system will have no bearing on the explanation of the decision reached.

Another reason to decouple problem-solving and explanation is to reduce the trade-off between problem-solving effectiveness and explanation effectiveness. Many times a method that is more efficient may not lend itself to producing understandable explanations In fact, in order to produce good explanations, additional knowledge beyond that used in problem-solving is required, such as linguistic knowledge and additional domain knowledge or rationale.

User models can be used in solving many of the problems associated with explanation generation. Models can contain the user's plans, goals and even his level of expertise which can influence explanation text. An example is found in ADVISOR, an expert system that provides information about courses [McKeown & Weida 88]. The explanation context and even the advice given are influenced by the user's goals, which are inferred during the session. This system highlights goal-related advice in an explanation by reorganizing and pruning a tree-structured inference trace produced by the system.

Other systems exploit the user model to aid in decision-making, rather than in explanation generation. An example of this is the GRUNDY system [Rich 79a]. As discussed previously, this system builds a model based on self-descriptive words which the user supplies. These words add facts, inferences, or whole stereotypes to the user model. The books in GRUNDY are rated with the same attributes contained in the model. This way similar values between user model facets and book facets help select a novel of interest.

Another system which uses the user model data directly in decision-making is the Real Estate Agent [Morik & Rollinger 85]. This system queries the user for information related to the domain of apartment seeking, i.e. price range, number of persons to occupy the apartment, and children, etc. It uses stereotypes in the form of inference rules which generate a list of criteria that are rated as very important, important, or unimportant. For each criterion, there are rules which consist of conditions which fulfill that criterion. These rule conditions are checked against the apartment database for suitability.

1.8.4 Adaptive User Interface

For any computer system to be successful, albeit an expert system or database retrieval system, it must have a well-designed man-machine interface. Much emphasis has been placed on the design of this interface, including that of an adaptive interface. The idea of an adaptive interface is straightforward. Simply, it means that the interface should adapt to the user, rather than the user adapting to the system. There are two ways a system can be adaptive [Norcio & Stanley 89]. The first method entails leaving the interface in a form that enables modification by the user if the behavior of the system is judged unsatisfactory. The second form of adaptation is the dynamic adaptation by the system itself. This dynamic adaptive interface requires information that is not required by a static interface. Much of this knowledge is about the user and is built in the form of a user model.

In an adaptive interface, the user model varies from user to user, and needs to be modified by the system as the user gains more experience. The purpose of the model is to deduce the user's level of expertise and experience. This can be done by collecting input data such as command types, error rates, and speed, much like in intelligent tutoring systems [Norcio & Stanley 89].

But there are cognitive issues that can be important in modeling the user as well. In order for a system to adapt to an individual user, it must encompass information about the users' cognitive differences. Two such categories in which users might differ are verbal and spatial abilities [Norcio & Stanley 89]. [Yallow 80] investigated this concept by presenting material in both graphic and verbal formats. His results showed that it is better to present material to a user in a format which he has demonstrated greater abilities. For example, the system may use a graphical display for the user who has demonstrated a high degree of spatial ability, or a textual display for the user who has demonstrated good verbal abilities.

One way which an adaptive user interface can dynamically model its user is via a neural network. User modeling can be viewed as a classification task, where users are placed into predetermined categories such as novice or expert, depending on their behavior. The user characteristics modeled become the basis for classification. These characteristics form an attribute set which contains the features that determine the classification. ADAM (Advanced Distributive Associative Memory) classifies its users this way by associating an input pattern with a class pattern [Finlay & Beale 90].

1.8.5 Cooperative Problem-Solving Systems

Cooperative problem-solving involves a collection of agents (viewed as logically-distinct processing elements) which are attempting to solve a problem. The agents can compute solutions to subproblems which are interdependent, and then communicate its results. A distributed problem-solving network can be viewed in two ways. One way is as a single entity which decomposes a problem and assigns subproblems to its various subprocesses. A second view can be a collection of independent problem-solvers that can communicate [Durfee et al 87]. Each problem-solver has the knowledge to make its own decisions about subproblems to solve and subproblem solutions to communicate. Since these problem-solvers make local decisions about what actions to take, each is self-interested because it attempts to maximize its own rewards of achieving its local goals.

There are many theoretical issues that arise in such systems. One is the control method that efficiently guides the problem-solving process. A second issue is that of communications that will yield cooperation. The third issue deals with organization and knowledge that leads to improved cooperation, communication, and decision-making.

The use of models within cooperative problem-solving systems is one way to organize knowledge that can substantially improve cooperation and communication among agents. Agent models can contain the same knowledge as user models, their goals, plans, and beliefs, but may contain knowledge about other agents as well. Some systems incorporate a multi-learning scheme that enables agents to learn about one another. This learning can be in the form of data exchange or knowledge transfer among the agents [Shaw 90].

The Designer Fabricator Interpreter, DFI, is an example of such a cooperative problem-solving system. It is a knowledge-based system that evaluates beam-to-column connections from the viewpoint of three construction agents - the designer, fabricator, and erector. Its intent is to address the lack of interaction among these agents. The system gives structural designers, the intended users, the ability to check their preliminary connection designs against general fabrication and erection knowledge to determine how their initial design decisions may affect fabrication and erection processes. The use of models to aid in the cooperative problem-solving will be investigated.

1.9 Outline of Thesis

Chapter 2 begins with a summary of the current DFI system followed by a list of areas where improvements were deemed necessary. Chapter 3 is a description of the proposed new DFI, which includes agent models to aid in the problem-solving process. Chapter 4 is a closer look at these models, the knowledge they would contain, and an example of how they would work. Chapter 5 contains a summary and future extensions of this work.

2. DFI - A Cooperative Problem-Solving System

The DFI system was developed to improve cooperation among the three construction agents. It is briefly described in this chapter beginning with an overview of the system. The information flow of the system is then discussed, along with the agent communication scheme. The connection evaluation process and the use of the connection information form is also summarized, as well as the development of the explanation facility. Lastly, system deficiencies are discussed, leading up to the rationale for the redesign of DFI.

2.1 Overview

The specific objective of DFI is to provide a tool which can incorporate construction knowledge into the preliminary design stage of beam-to-column connections. It achieves this by developing a framework for distributed cooperative problem-solving among construction agents. The system is distributed in that each agent has his own issues, and attempts to maximize his own position by achieving local goals. The agents review connections that are proposed with respect to these issues, and respond accordingly. In this manner, issues relating to downstream construction processes such as fabrication and erection are brought to light early in the design phase.

2.2 Information Flow

An information flow diagram is shown in Figure 2-1. The boxes in the figure indicate major computational processes. The dashed box at the bottom of the figure represents the connection evaluation process by which three agents individually evaluate proposed connections. The ovals represent input from and output to the user.

The user inputs the initial building design including beam and column schedules and framing plan. The building description is maintained in a frame representation as a hierarchical decomposition of objects such as floors, beams, column, connections and their component and fastener pieces. After a floor is chosen, the user selects a particular column and beam. He is then guided interactively to select a standard connection configuration. This part of

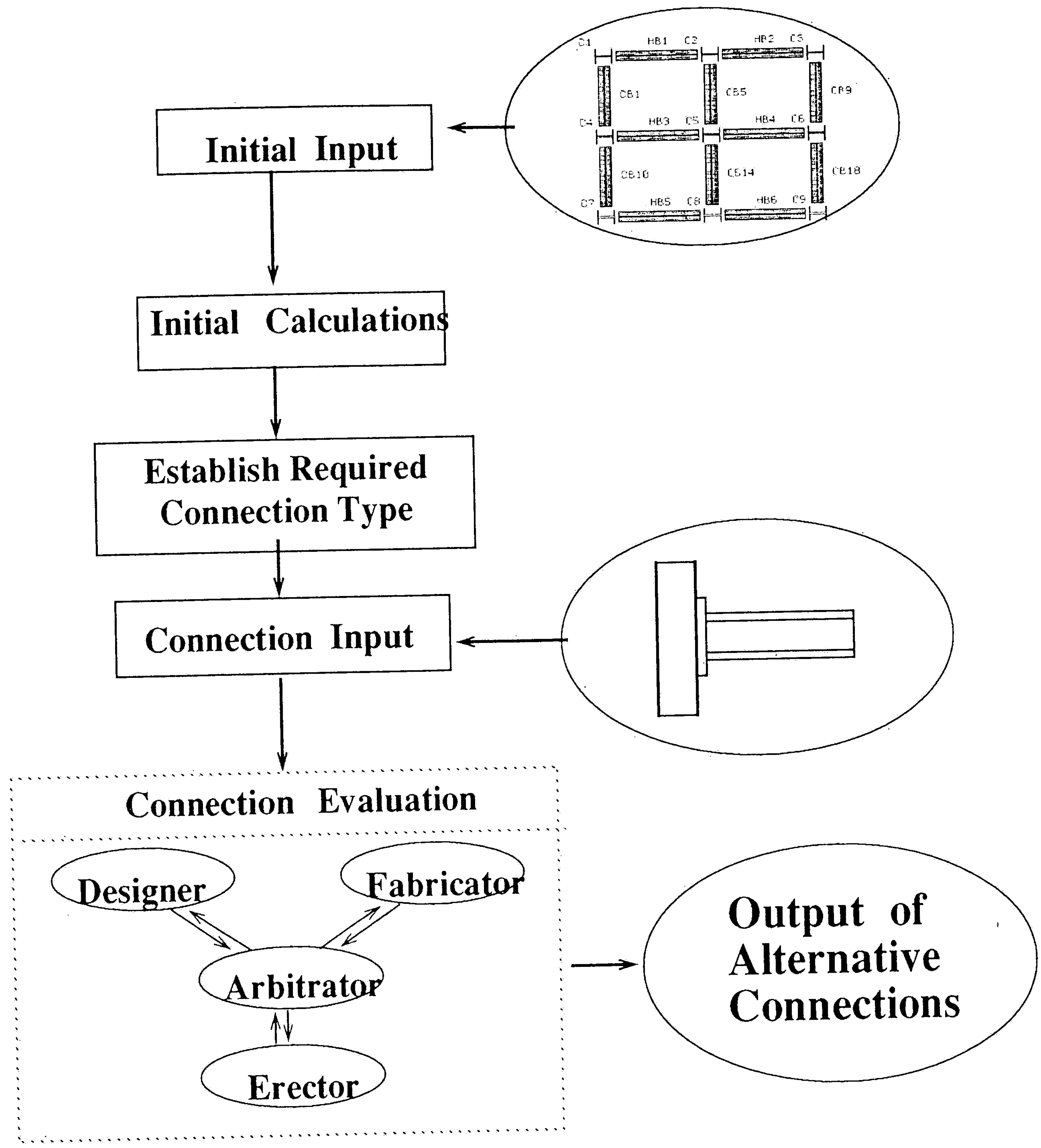


Figure 2-1

the input is constrained to only feasible connections [Barone 90]. Next the user selects a key issue. This key issue directs the evaluation process by restricting from consideration those connections which have a less than acceptable rating for the key issue.

The dashed box depicts the entire evaluation process where agents evaluate proposed connections and respond accordingly. At the end of the evaluation process, the user is presented with the original connection configuration and three potentially different connection configurations proposed by each agent (designer, fabricator, erector). It is then up to the user to review the results and the agent's explanation of his selection, and make an informed decision about the final connection configuration.

2.3 Agent Communication

A message or blackboard area was developed so the agents can post various messages during the evaluation process. This information is also provided to the user.

When conflicts between agents arise, an independent arbitrator agent is invoked. The arbitrator serves two purposes. He checks on which agent has been worst affected by the proposal, and allows that agent to go next. The arbitrator also intervenes when there is a stalemate in the evaluation process. The arbitrator uses shared knowledge about the connection and past proposals to convince other agents to accept a previously rejected connection. A much more detailed description of the negotiation process, agent communication, and the role of the arbitrator can be found in [Werkman 90].

2.4 Evaluation Process

The evaluation process begins with the designer evaluating the input connection according to his own issues. Each connection has associated with it a Connection Information Form (see Figure 2-2) which lists ten issues along with a rating of 1 to 5 (least desirable to most desirable). Connections are evaluated by taking the worst (lowest value) issue and improving it with alternate configurations. Prior to selecting an alternate connection, the evaluating agent must search the connection database for all of the connections which have a greater value than the worst issue, but also maintains a minimum value of 3 for the key issue. Once this set of connections is determined, the evaluating agent selects the connection which has the highest composite score. This selection is posted to the blackboard.

The arbitrator checks the blackboard for which agent is worst affected by this proposed connection, and gives that agent control to evaluate it next. The agent repeats the procedure described above by searching the database and selecting the connection with the highest composite score. The arbitrator halts the evaluation process when each agent has had a turn and when at least two agents agree on a connection. The Connection

CONNECTION INFORMATION FORM

endplate

1. TITLE BLOCK

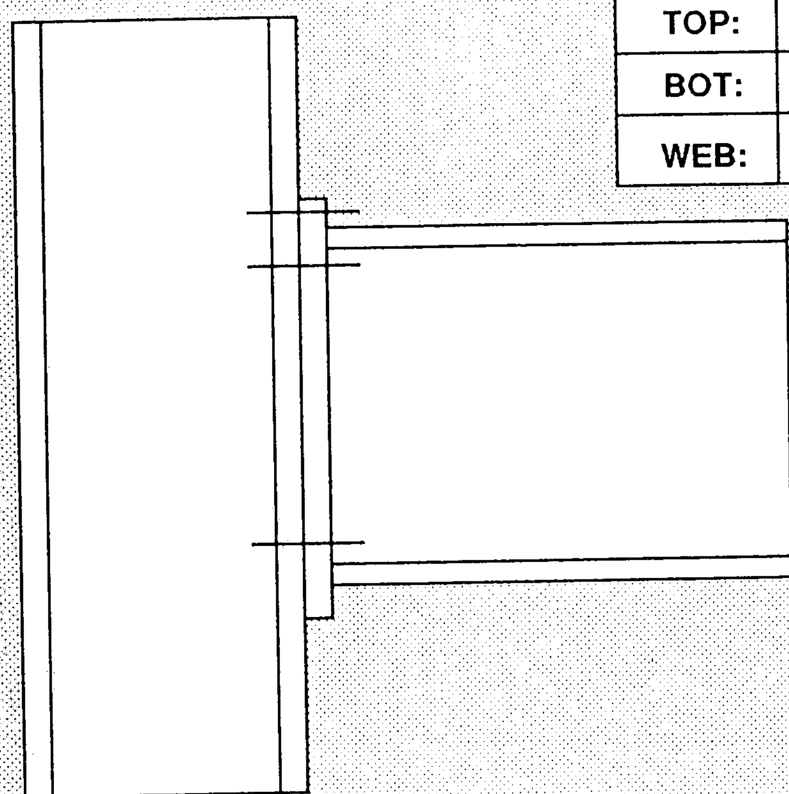
CONNECTION NAME: ***ENDPLATE***

TYPE: ***1*** ORIENTATION: ***FLANGE*** SHEET: 1 of 1

2. CONN DETAIL TABLE

	DETAIL MAT.	COL. FAST.	BEAM FAST.
TOP:	ENDPLATE	FIELD BOLT	SHOP WELD
BOT:	ENDPLATE	FIELD BOLT	SHOP WELD
WEB:	ENDPLATE	FIELD BOLT	SHOP WELD

3. PROFILE VIEW



4. RATING FACTORS TABLE

	DESIGN	FAB.	EREC.
STRENGTH	3		
STIFFNESS	3		
RELIABILITY	3		
VERSATILITY	2		
FAB. COST		4	
FAB. EASE		3	
MAT. COST		4	
EREC. COST			2
EREC. EASE			2
SAFETY			3

5. COMMENTS:

D: Check if column stiffeners are required. If stiffeners are required they need not exceed one-half the column depth when the beam is on one side only.

D: For fillet weld > 1/2" to attach the endplate consider using full or partial penetration groove welds with reinforcement.

F: Endplate connections require dimensional control to tight fit-up to column flanges, which is affected by column flange-to-web squareness, beam camber and squareness of the beam end.

F: Shim space may be provided for accommodating mill and fabricating tolerances. Use "finger" shims entered from each side where feasible.

E: Field bolts must be furnished long enough to accommodate any shim allowance.

Figure 2-2

Information Form and the ratings provide the basis for this evaluation process. For more detail about the connection information form and how these ratings were derived, see [Barone 90]. An example of an evaluation session can be found in Appendix A.

2.5 Explanations Using DFI Relational Network

The explanation facility was developed around the DFI relational network. This relational network models interactions between designers, fabricators, and erectors during the evaluation process [Barone 90]. The network is centered around a connection which is made up of functional and physical aspects. The functional aspects deal with rigidity and performance of the connection. The physical aspects involve the actual parts that make up the connection and the operations required to assemble the connection. Relations exist that link one agent's issue to another agent's issue through the functional or physical aspects of a connection.

The specific relations between one agent issues were developed by Marc Barone. (For a more detailed description of how these inter-agent and intra-agent relationships were developed, see [Barone 90].) The author took the initial set of relationships and expanded it to include one additional level of detail, that of the subissue, provided by the issue decomposition discussed in [Barone 90]. An example of a new relation is shown here:

```
relation(expense,
[agent_issue(fabricator,fab_cost,fab_proc,physical,
        [construction_schedule(duration,[fabrication_schedule
                erection_schedule,stability_analysis]),
         shop_operations(method,[cutting,drilling,welding,shop_assembly])]),
 agent_issue(erector,erec_cost,con_proc,physical,
        [construction_schedule(duration,[fabrication_schedule,
                erection_schedule,stability_analysis])
 field_operations(method,[fastening_methods,temporary_bracing])])]).
```

When a connection is being considered and a relation is found to exist between two issues of importance, an explanation is formed consisting of the information contained in these relations. An example of explanation text (derived from the above relation example) is given here:

This connection's limiting "Key Issue" is Strength (4, Designer issue) based on the user's Endplate connection.

Refusal by Erector to Designer:

The Erector has rejected the Endplate because from his perspective, the connection's Erection Cost is unacceptable due to Expense of Physical aspects.

Erection Cost is influenced by construction procedures properties:

- Construction Schedule, a time factor of construction procedures, includes fabrication schedule, erection schedule, stability analysis.

- Field Operations, a method of construction procedures, includes fastening methods, temporary bracing.

2.6 System Deficiencies

2.6.1 Limited Connection Database

The current DFI system is limited to evaluating Type 1 or moment resisting connections only. The database consists of 13 connections, each of which is rated according to the ten issues: strength, stiffness, reliability, versatility, fabrication cost, fabrication ease, material cost, erection cost, erection ease, and safety. Because of this limited database, in many of the evaluations an agent simply runs out of connections from which he can choose. Developing additional connections was not within the scope of "demonstration of concept" prototype.

2.6.2 Control Strategy

Any distributed artificial intelligence system must have some form of control strategy to efficiently guide the problem-solving process. Within DFI, the control is in the form of an arbitrator. The arbitrator monitors the current state of all agent proposals and reviews each proposal for any problems it might cause another agent. If the arbitrator detects a problem (in the form of a low composite score), the arbitrator gives control to that agent.

The arbitrator also reviews the discourse history to check for a halting condition or a deadlock. A halting condition exists when two or more agents agree on a connection, in which case control is returned to the user. A deadlock condition exists when there are no connections left to propose. In this case, the arbitrator intervenes by analyzing which connections have been rejected and tries to convince an agent to reconsider.

This scenario of a third-party arbitrator taking control of the evaluation process is not an intuitively satisfactory solution to a real-world connection evaluation process.

Although it is agreed that one person must be in control, in this particular situation it would probably be the designer.

2.6.3 Evaluation Process

The system currently begins with the evaluation of the connection which the user has selected. This connection is developed through prompts to the user represented in the hierarchy shown in Figure 2-3. An agent evaluates this connection based on his own issues; for example, the fabricator only looks at the ratings for fabrication cost, fabrication ease, and material cost. If he is not satisfied with the ratings, he rejects the connection and proposes one that has improved ratings for his issues.

One of the problems with this evaluation process is the level at which the negotiation takes place. Most of the detail material, and thus the fastening methods have already been decided upon. This fact makes negotiation difficult as opinions about connections may be very different.

Also, proposed connections are made without consideration of what connections have already been proposed or of any specific constraints. The resolution or agreement on a connection configuration can be pure luck.

The evaluation process also does not take into consideration the context of the connection within the building. This contextual information could increase the importance of some of the issues, like stiffness, while minimizing some others. This difference in the relevance of certain issues could help select more appropriate connections.

Another criticism of the current system is that it does a qualitative evaluation only. In an actual evaluation process, quantitative numbers for some of the issues like strength and cost would be developed in order to make a more accurate analysis.

2.6.4 Limited Agent Knowledge

The present system contains only the individual agent issues pertaining to a specific connection. This information is part of the global connection database. The reasoning behind the connection ratings is found in another global database - the DFI relational network. This relational network serves as a good medium for agent communication regarding how one issue may affect another. However, for an agent's own issues, his ratings and comments should reside in some sort of agent model.

DFI Connection Hierarchy

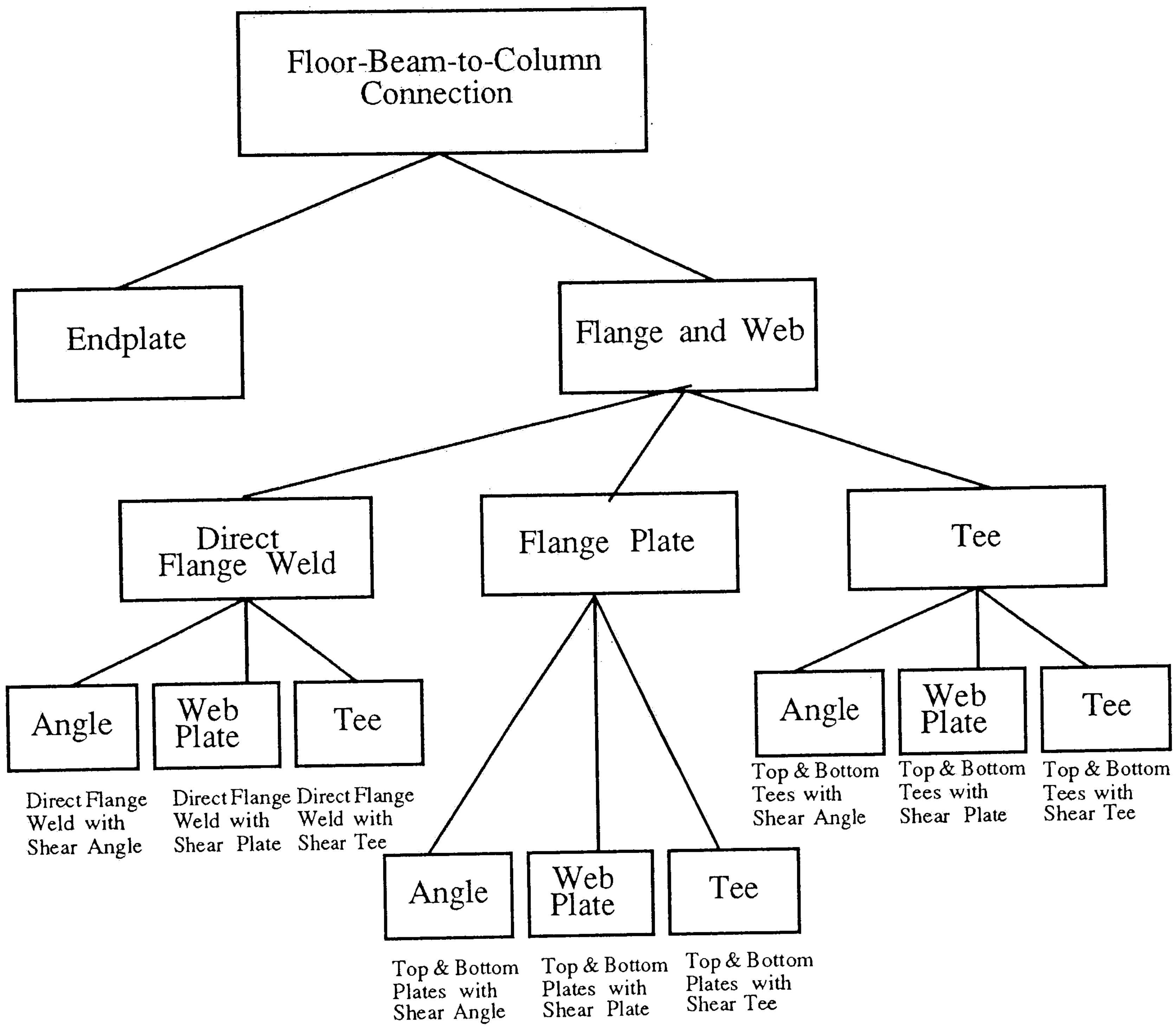


Figure 2-3

3. Proposed Architecture for Extended DFI

This chapter describes the proposed architecture of a designer-driven system and is the combined effort of the research team listed in the acknowledgments. The system consists of two distinct phases which are described in Sections 3.1 and 3.2. Discussed in Section 3.3 are the system "building blocks" which describe the knowledge and its organization. Section 3.4 contains flow diagrams for Phase I and Phase II which show the sequence of activities and the databases that are accessed during each activity. This research deals specifically with the qualitative analysis taking place in Phase I. Phase II briefly describes how the qualitative data of Phase I would be utilized, and is beyond the scope of this thesis.

3.1 Phase I

Within Phase I the agent models are consulted and connections are evaluated qualitatively, according to agent issues. One improvement over the original DFI is the inclusion of initial goals and initial constraints. This additional feature requires that the connections selected for evaluation must meet the requirements set forth by these goals and constraints. They will be obtained by querying the user for information such as connection type, moment, and strength requirements. The set of feasible connections is initially reduced by applying these initial goals and constraints. The reduced connection set is then evaluated according to specific criteria which are also derived by the initial goals and constraints. Responses are obtained from both the fabricator and erector, inappropriate connections are discarded, and the further reduced connection set is then ranked.

3.2 Phase II

The output of Phase I, a reduced, ranked set of connections, would serve as the input to Phase II. Phase II is characterized by a quantitative evaluation. Each connection in the set that proves to be a viable alternative would be fully designed according to the <u>AISC Manual of Steel Construction</u> [AISC 80]. This fully designed connection can be evaluated using the quantitative data which becomes available when the connection is designed. Some examples of new (Phase II) quantitative information are the specific data that represent strength, material cost, fabricator cost and erection cost. The connections are again ranked and a connection selection is returned to the user.

3.3 System Building Blocks

The system "building blocks" contain both data and knowledge that is required in the connection evaluation process. Besides the contents of the database, it is noted what knowledge is available to whom in terms of being global or local to an agent.

3.3.1 Connections Database

The set of all connections which the system knows about is represented as groups of attribute-value pairs. Connections will be logically grouped by a method of operation (shop weld, field bolt) and by its detail material (tee, angle). A set of 81 different combinations of shop and field operations have been identified as feasible connections and will form the basis of the database. (A discussion of how 81 were arrived at with a complete listing can be found in Appendix B.) This is a substantial increase from the original DFI connections database which contained only 13 connections. The number of connections was limited due to the difficulty in developing the ratings for the ten issues needed for each connection. In the new system, the ratings will be derived automatically from the agent models which contain ratings on the specific operations. An example of an entry in the new connection database follows:

```
(Endplate
    (orientation              flange)
    (type                     1)
    (moment_connection        field_bolt/shop_weld)
    (shear_connection         field_bolt/shop_weld)
    (category                 field_bolt/shop_weld/field_bolt/shop_weld
    (detail_material          endplate)
)
```

Note that individual ratings which were contained in the original connections database are no longer present in the new representation. This data is derived by consulting the agent models, which contain ratings based on operations, i.e. shop bolt, shop weld, field bolt, field weld. The models also contain ratings based on the connection detail material.

3.3.2 Queries

A set of questions will be asked of the user. These queries will derive the initial goals and initial constraints (IG/IC), which will be global. The IG/IC will be applied to the connections database to reduce the number of categories. Therefore, the queries must be aimed at obtaining the information necessary to determine which connection types would be appropriate (Type 1 or Type 2) and appropriate strength requirements. The IC/IG will also be used to derive the specific tasks and sequence to be followed by the designer, i.e. the operational model. The IC/IG will also serve to place more importance on specific issues according to the specific design situation. This will be accomplished by assigning appropriate "salience" or priority values to issues. These salience values will help resolve conflicts between two agents by making one issue a higher priority than another issue.

3.3.3 Type A Knowledge

Type A knowledge is common to all agents and shared. Given an operation and attribute and an issue of connection design, all three agents are within a level of tolerance and can accept the proposal. Type A knowledge will be retrieved by the designer when more information is needed per a design task.

3.3.4 Type B Knowledge

Type B knowledge is not common, and not previously shared or agreed upon. When dealing with a specific attribute or operation of connection design, there exists an inter-agent relationship which affects an agent(s), and a level of negotiation takes place. This consultation is invoked by the designer when the design task indicates that more information is required, and no Type A solution exists. The agent responds to this consultation with preferences and rating factors.

3.3.5 Type C Knowledge

Type C knowledge is distinct to an agent, not shared. Within this system, Type C knowledge can be found in each of agent models. For example, within the designer/user model are design tasks and heuristics for decision-making. Within all the models are biases and preferences which are distinct to that agent.

3.3.6 Designer/User Model

The following is a list of both the derived and embedded knowledge which resides in the designer/user model. This model contains the operational model (or set of design tasks) that drives the process.

<u>Derived Knowledge</u>

A specific task list to be followed - operational model

Salience values for issues which reflect "situation specific" priorities

<u>Embedded Knowledge</u>

Knowledge of pre-coded, generic tasks of connection design process.

Heuristics which use IC/IG to select appropriate tasks and their sequence

Heuristics which apply salience values to issues

Knowledge to deal with conflicts between fabricator and erector suggestions, possibly through the combination of salience factors of issues and their ratings

Features (operations and attributes) of connections which fulfill or violate agent's criteria (reflected with a rating factor) with respect to an issue

Criteria (issues) with default salience values indicating priority

Design biases and/or preferences

3.3.7 Agent Models

The following is a list of knowledge which resides in the fabricator and erector agent models. This knowledge is all embedded.

Features (operations and attributes) of connections which fulfill or violate agent's criteria (reflected with a rating factor) with respect to an issue

Criteria (issues) with default salience values indicating priority

3.3.8 Discourse History

The discourse history is a global database that provides a record of the session. It is consulted by all agents in order to avoid proposing connection categories that have already been evaluated. It contains a list of the connections that have been proposed and rejected, and the current connection and issue being discussed.

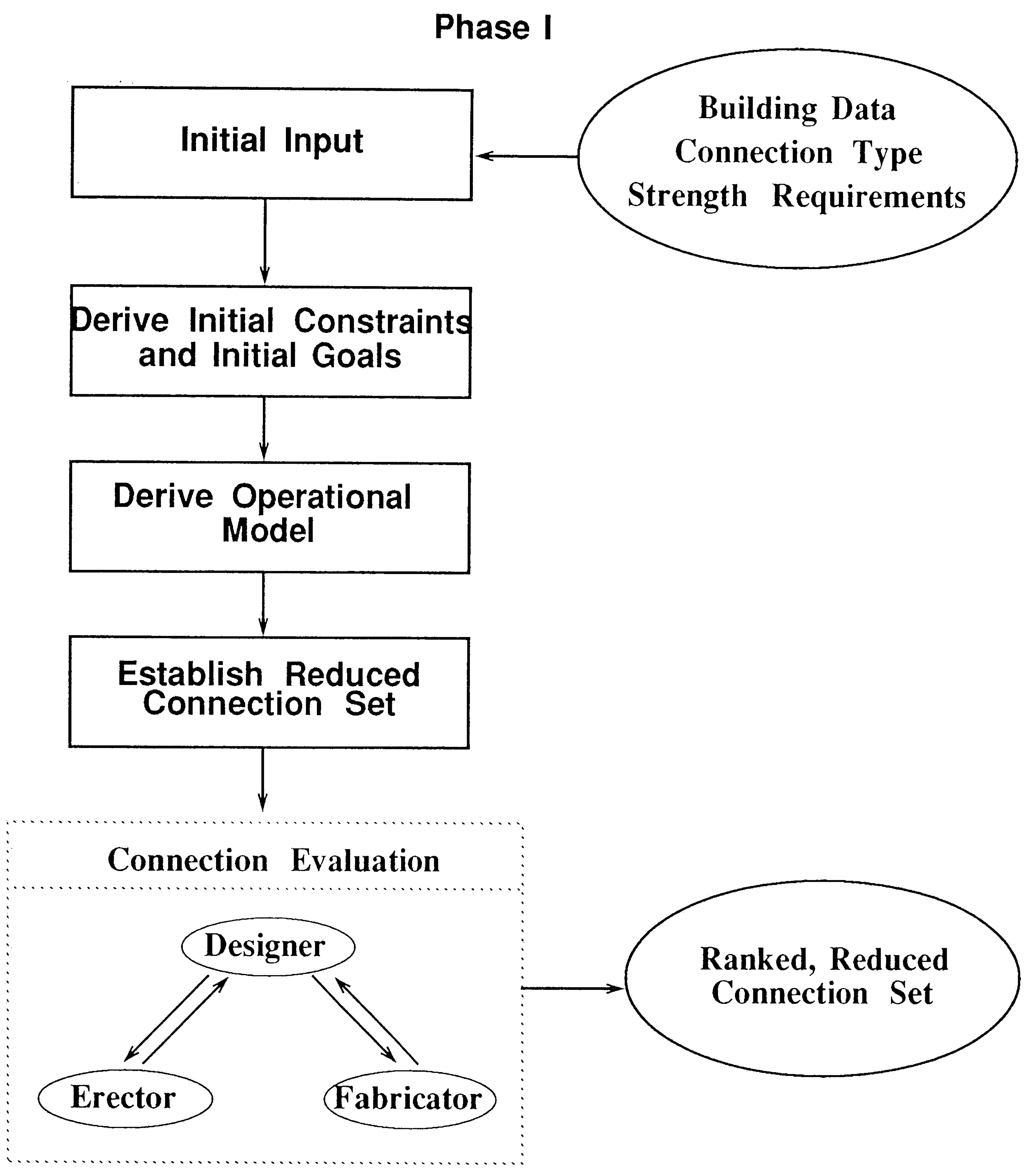


Figure 3-1

3.3.9 Tasks

Tasks are the specific procedures which make up the operational model. The operational model is derived from the generic task list via 1) initial constraints and goals; or 2) intermediate results of the connection evaluation/elimination process.

3.3.10 Issues Network

This is a global database of issues, subissues, and inter-agent relationships used in communication with the designer agent. This is the same relational network which was discussed in Section 2.5. A complete set of relations among issues and subissues is found in Appendix C.

3.4 Information Flow

Figure 3-1 illustrates the information flow of Phase I, which consists primarily of a qualitative evaluation.

The first step involves querying the user, who then inputs the contextual information such as building data, connection type, and strength requirements. This data is then used to derive the initial constraints and initial goals. All connections being evaluated must comply with these constraints and goals. An example of this may be an instance where field welding is not available. In this case, the hard constraint of no field welded connections would prevent field-welded connections from being considered.

The next processing that takes place is that which derives the operational model of the designer, which then becomes part of the designer/user model. The operational model contains the list of tasks to be followed. The constraints and goals are also used to adjust the salience values on the issues according to what the specific design situation dictates. All of this data resides in the designer/user model.

Next the initial constraints and goals are applied to the entire set of 81 connections. This is the first cut, so to speak, and narrows the selection by connection type at the very least; and at most by certain operations, depending on what the initial input was.

The dashed box shown in Figure 3-1 represents the evaluation process initiated by the designer. The designer consults the fabricator and erector, in turn, to get their input on the connection types remaining in the set. The agent models respond with ratings as well as explanatory comments about why a category is favorable or unfavorable. The designer must them assimilate this data into a ranked, reduced connection set with the help of the designer user model. Again, the priorities dictated by the initial goals and constraints help the designer to establish relevance to the responses generated by the agents. Please note that a complete description of the agent model response is given in Chapter 4.

Figure 3-2 illustrates the information flow of Phase II, which consists primarily of a

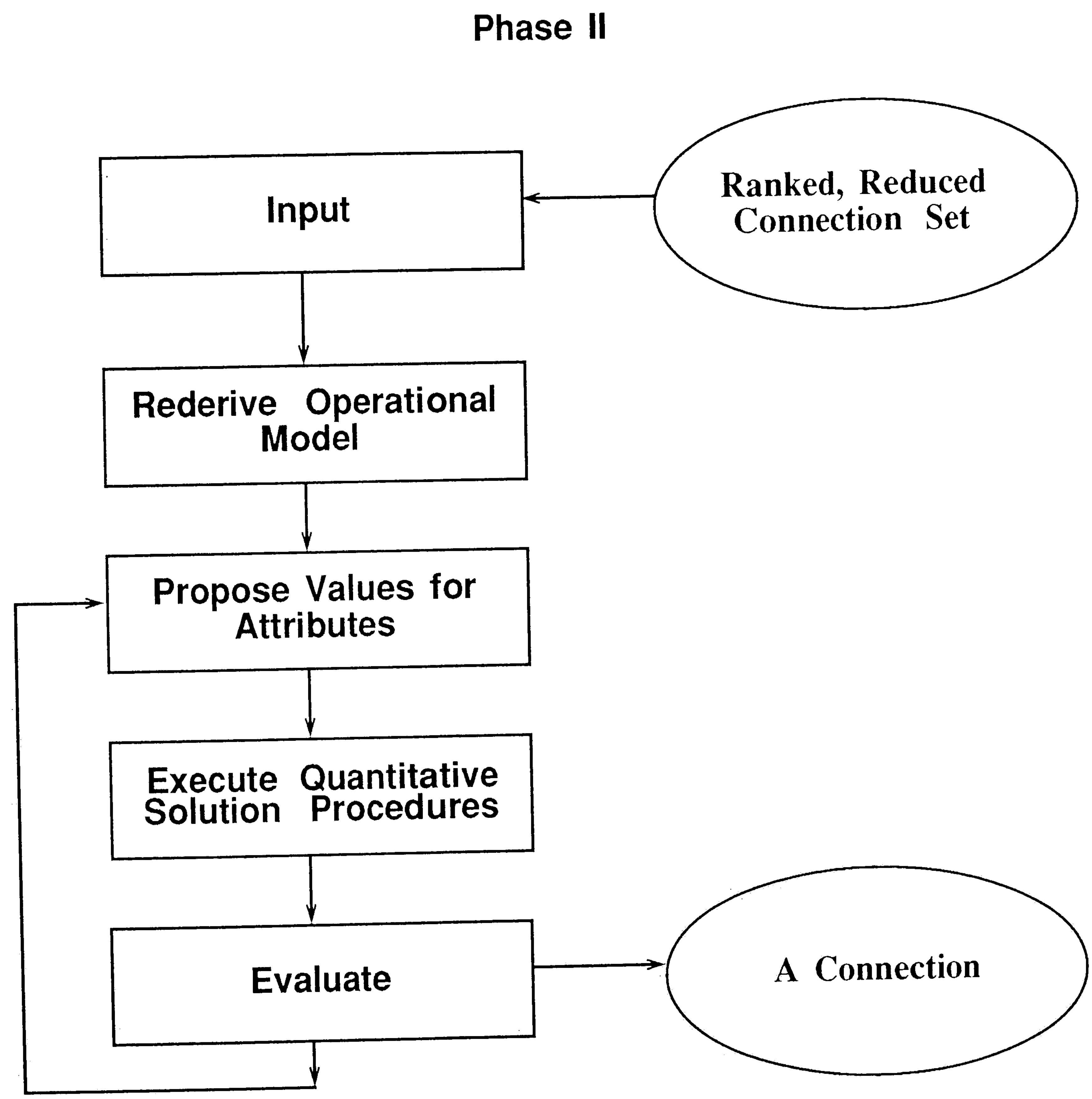


Figure 3-2

quantitative evaluation.

The input to Phase II is the ranked, reduced connection obtained as the output of Phase I. The first processing to take place in Phase II is that which rederives the operational model. The operational model is now influenced not only by the initial goals and constraints, but by the intermediate results obtained in Phase I.

Next, attributes would be proposed for each connection remaining in the set. When attributes are selected, the connection would be fully designed according to [AISC 80]. Once designed, a quantitative evaluation can take place which looks at numerical values for strength, material cost, fabrication cost, etc. These three steps are to be iterative so that all detail materials appropriate for that category are considered. Each connection that remains in the ranked, reduced set would be designed and evaluated in this way.

The outcome of this more in-depth and quantitative evaluation would be a completely detailed connection.

4. Agent Models in DFI

This chapter begins with a brief overview of the DFI agent models and their goals. It then describes the stereotypical aspect of the model as well as the hard constraints which represent the dynamic aspect of the model. Two different representation schemes are discussed. The first is a record-like structure which uses rating factors similar to those used in the current system. The second representation uses a relational scheme with preference statements. Examples of models are given in each representation, as well as the advantages and disadvantages of each.

4.1 Overview

Agent models support problem decomposition, which is the breaking of an overall goal into subgoals. Each agent solves his individual goal which contributes to the achievement of the overall goal. The agents are cooperative in that they may need a piece of information from another agent before they are able to generate their own preference. This occurs in the form of a cross issue, i.e. an issue which is made up of several subissues, at least one of which is shared by another agent. The active agent must then send a message to the agent who shares the subissue for data that is pertinent to his preference statement.

Statements and rules that reside in the agent models are considered to be base-level knowledge. This means that the knowledge in the model represents the capability and purpose of that agent only. Agent preferences can be represented directly as statements, or derived through rules contained at the base level. The preferences are passed to the meta-level for consideration by the designer in the decision-making process.

The meta-level contains various design strategies, or soft constraints, as well as the hard constraints. A hard constraint is a criterion which cannot be violated, and represents the dynamic conditions of connection design which vary for different situations. They represent clear-cut and well-defined situations. An example of this may be an instance where field welding is not available and the construction of the building is on a tight time schedule. In this case, the hard constraint of no field welded connections would overrule any agent preference for this method of connection, and would become the driving factor in the designer's strategy. There are many such constraints that can enter into connection design. Other examples include a specific fabricator that requires all bolting operations, extreme timing considerations or extreme cost considerations.

Soft constraints are the decision strategies necessary to choose one best preference over the individual preferences supplied by the models. Other examples of soft constraints, which reside at the meta-level, are negotiation strategies which will resolve ambiguity and conflict between agents, and communication strategies that will enable the designer to gather individual preferences and other relevant information from the agents.

4.2 Goals

The overall goal of the agent models within the DFI system is to assist in the problem-solving process by supplying an important viewpoint early in the preliminary design stage. This consideration of designer and fabricator issues can help foster communication among the agents, and help avoid costly delays and rework in the construction process.

As discussed in Section 1.2, there are additional goals of models in knowledge systems that can enhance both efficiency and effectiveness. One such goal is to improve the overall system performance by running several processes in parallel. This occurs when the designer simultaneously requests data from two separate agents. Another goal is to increase the variety of solutions by allowing agents to form a local solution without influence by other agents.

4.3 Agent Models as Stereotypes

The three agent models - designer, fabricator, and erector are patterned after the stereotypical models found in GRUNDY [Rich 79a]. The designer, fabricator and erector models contain static information, i.e. viewpoints, preferences, and issues believed to be typical of that agent. These initial static preferences will serve as default values only, and can be overridden by information the user inputs. Thus, instead of the model changing to reflect different users as in GRUNDY, the model changes to reflect different design situations.

4.4 Models as Record-Like Structures

4.4.1 Description

Many of the models discussed in the literature are represented by record-like structures [Morik & Rollinger 85, McCoy 85, Rich 79a], and this is one of the proposed representations for the DFI agent models. The contents of the models will be similar to those found in [Morik & Rollinger 85] which contain the user's criteria, and specific features that fulfill or violate the criteria. Within DFI, the criteria correspond to that agent's issues while the features correspond to the method of connection (referred to as category) and the detail material. The degree to which features fulfill agents' criteria is measured by ranking (from 1 to 5) each category with respect to a specific issue. This method of ranking allows for a measure of how much better (or worse) one connection method is than another with respect to an issue. It also supplies the user with a measure of undesirability with the lower values.

Within a specific category (field bolt, shop weld), the detail material is ranked as well. Detail material is looked at within a category since it does not make sense to evaluate detail material independently. The selection of detail material for a connection is very much dependent on its method of operation or category.

The last data item found in the model is a priority measure of that agent's issues, i.e. how important is that issue to him. These are used as default values, ranging from low, medium, or high, and assist in the decision-making process when there are no hard constraints to dictate selection.

4.4.2 Example of a Fabricator Agent Model

Figure 4-1 is an example of the fabricator model containing data derived from the current DFI connection database. Five of the 81 categories are included in this connection database. The first ten statements represent rankings on categories with regard to an issue, i.e. FB/SW/FB/SW (FB = field bolt; SW = shop weld) being the category, 4 is the ranking on a scale from 1 to 5, and fab_cost is the issue. The next nine statements represent the ranking of the detail material within a category. The format of these statements is the same, but introduces the detail material options available within that category. The next three statements represent the fabricator's priorities with regard to his own issues, with HP being high priority, MP being medium priority, and LP being low priority.

```
(fabricator
    (FB/SW/FB/SW        4        fab_cost)
    (FB/SW/FB/SW        3        fab_ease)

    (SW/FB/SW/FB        2        fab_cost)
    (SW/FB/SW/FB        3        fab_ease)

    (SW/FW/SW/FB        2        fab_cost)
    (SW/FW/SW/FB        2        fab_ease)

    (FW/FW/FW/FW        3        fab_cost)
    (FW/FW/FW/FW        2        fab_ease)

    (FW/FW/SW/FB        3        fab_cost)
    (FW/FW/SW/FB        4        fab_ease)

    (SW/FB/SW/FB   2   flange_plate_w_tee         mat_cost)
    (SW/FB/SW/FB   3   flange_plate_w_angle       mat_cost)
    (SW/FB/SW/FB   4   flange_plate_w_webplate    mat_cost)
    (SW/FB/SW/FB   1   tee_w_angle                mat_cost)
    (SW/FB/SW/FB   1   tee_w_tee                  mat_cost)
    (SW/FB/SW/FB   2   tee_w_webplate             mat_cost)

    (FW/FW/SW/FB   2   flange_weld_w_tee          mat_cost)
    (FW/FW/SW/FB   4   flange_weld_w_angle        mat_cost)
    (FW/FW/SW/FB   5   flange_weld_w_webplate     mat_cost)

    (mat_cost      HP)
    (fab_cost      HP)
    (fab_ease      MP)
)
```

Figure 4-1

The connections database representation must include the same features that are listed in the agent models, in order for that agent to evaluate it. On the following page is an example of the entry for the endplate. A complete connections database, with current DFI connections mapped into this new representation is found in Appendix D. The database entry for the endplate connection follows:

```
(endplate
    (orientation          flange)
    (type                 1)
    (moment_connection    FB/SW)
    (shear_connection     FB/SW)
    (category             FB/SW/FB/SW)
    (detail_material      endplate)
)
```

Note that the category is broken into the moment and shear connection. It is anticipated that in the future rules can be developed that will help derive some of the preferences rather than simply listing preference statements. These rules would be located in the model at the base-level.

4.4.3 Model Responses

Presently, there are a variety of queries that could be asked of the agent. The first group of queries involves a single choice, i.e. the designer asks the agent for a preference on one item only. A second group of queries involves multiple choices.

The examples given show only what the system response would be to the designer. It is anticipated that a separate response in the form of an explanation would be provided to the user as well. This would consist of the rating factor translated to a linguistic variable along with explanatory comments. What follows are some recommended linguistic variables.

$4.5 \leq x \leq 5$	extremely desirable
$3.5 \leq x < 4.5$	very desirable
$2.5 \leq x < 3.5$	more or less desirable
$1.5 \leq x < 2.5$	less desirable
$0 < x < 1.5$	least desirable

Single Choice

The first query could be that agent's overall view of a connection category, without respect to a specific issue. In this case, a composite score for that category with respect to all agent's issues would be calculated. The response is generated by obtaining the values 4 for fab_cost and 5 for fab_ease, and calculating their average.

query(fabricator, FB/SW/FB/SW)
response(fabricator, FB/SW/FB/SW, 4.5)

A more common query would be to obtain the agent's view on a connection with respect to a particular issue. This would occur more frequently since the designer will have an overall design strategy that is driven by the ten issues identified in the original DFI. When this strategy indicates that more information is needed about an issue, a message is sent to the appropriate agent. The response reflects the rating of that category with respect to the specific issue. This query and response follows:

query(designer, FW/FW/FW/FW, strength)
response(designer, FW/FW/FW/FW, strength, 5)

Another query to the agent could be for a preference regarding a combination of operations at either the moment or shear connection. Of course, how much information is requested of the agents is dictated by the designer's strategy. This strategy is, in turn, influenced by the hard constraints introduced by the user. This query and response follows:

query(erector, moment, SW/FB, erec_ease)
response(erector, moment, SW/FB, erec_ease, 5)

Another query to the model could be with respect to the detail material. Detail material must be discussed within a specific category. This query and response follows:

query(fabricator, SW/FB/SW/FB, flange_plate_w_tee, mat_cost)
response(fabricator, SW/FB/SW/FB, flange_plate_w_tee, mat_cost, 2)

Another single choice query to the agent is that regarding an entire connection configuration. Since rankings are based on connection categories rather than designed connections, the response must be formulated by consulting the connection database and retrieving the category and detail material associated with that connection. This query on a connection and the response follows:

query(fabricator, endplate)
response(fabricator,
[SW/FB/SW/FB, fab_cost, 4],
[SW/FB/SW/FB, fab_ease, 3],
[SW/FB/SW/FB, endplate, mat_cost, 2])

The last single choice query that can be made to the model relates to the agent's own priorities. If the designer has no basis for making a selection among contradictory agent preferences, he may opt to make the decision based on issue importance. This query on an issue and the response follows:

query(erector, erec_ease)
response(erector, erec_ease, MP)

Multiple Choices

Depending on the design strategy, a designer may require a response from an agent on a set of connection categories or on a set of detail material. Since a decision about a specific connection indicates the end of the negotiation process, it is not anticipated that a request would include a set of specific connections.

The first example is a query for preferences on several categories with respect to an issue. The response is in the form of an ordered set of selections according to preference, along with corresponding ranking values. The inclusion of the rankings within the ordered set gives an indication of the degree of preference. This query and response follows:

query(fabricator, [FB/SW/FB/SW, SW/FB/SW/FB], strength)
response(fabricator,
    [FB/SW/FB/SW, strength 4],
    [SW/FB/SW/FB, strength, 3])

Another query involving multiple choices is that of detail material. Once a category has been selected, a decision needs to be made on the detail material. The designer may want all the choices considered at one time. This query and response follows:

query(fabricator, FW/FW/SW/FB, [flange_weld_w_tee, flange_weld_w_web_plate,
    flange_weld_w_angle], mat_cost)
response(fabricator, FW/FW/SW/FB,
    [flange_weld_w_web_plate, mat_cost,4],
    [flange_weld_w_angle,. mat_cost,4])
    [flange_weld_w_tee, mat_cost, 2],

4.4.4 Advantages

The model which uses rating factors gives a "complete" picture of an agent's preferences; complete meaning there are no gaps in the knowledge. A preference on one item can be formulated without necessitating comparison among other items. There is also a measure of quantity associated with these ratings, i.e. a value of 4 is better than a value of 3, but a value of 5 is much better than a value of 3. The ratings also supply a measure of acceptability. If a category has a rating of 1, it is not a desirable selection from that agent's point of view.

4.4.5 Disadvantages

The individual ratings of all categories and detail material by issue can result in a very complex model. The example given in Section 4.4.2 contains preference statements for only 5 out of the 81 possible connection categories. Preferences on categories alone would result in 243 statements (81 categories x 3 issues). The further inclusion of preferences on detail materials would result in at least 81 more statements. The collection of this amount of data is very labor intensive and prone to errors.

Another problem with this method is the difficulty of assigning a ranking value of 1 to 5. There will most likely be a great deal of disagreement among the experts as to what that value should be. This is especially true in the domain of connection categories and detail material, which is not particularly well-defined.

4.5 Models as Preference Statements

4.5.1 Description

Another representation for the agent models proposed by [Wong 91] is via preference relations and integrity constraints. These relations state either a preference or an indifference about two categories with respect to an issue. The preference relation, written P(issue, category1, category2) states that category 1 is preferred to category 2 with respect to that issue. The indifference statement, I(issue, category1, category2) indicates indifference between category 1 and category 2 with respect to that issue.

There are similar preference and indifference statements about the detail material. Since the detail material of a connection cannot be considered separately from the category, the category becomes an additional parameter within the statement. An example of material preference is MP(category, issue, material1, material2), which means that within this category material 1 is preferred to material 2 with respect to that issue. Similarly, the material indifference is MI(category, issue, material1, material2).

There are also a set of relations regarding the prioritizing of agent issues. The statement for issue preference, IP(issue1, issue2), means that issue 1 has a higher priority than issue 2. The issue indifference statement is II(issue1, issue2).

In addition to these preference relations, each agent contains a set of integrity constraints which will derive additional relations and check for consistency of data. These integrity constraints are listed below. A similar set of rules would be included for the material preferences and the issue preference statements.

Asymmetry

(1) P(issue, cat1, cat2) implies ~P(issue, cat1, cat2)

(2) P(issue, cat1, cat2) implies ~I(issue, cat, cat2)

(3) I(issue, cat1, cat2) implies ~P(issue, cat1, cat2) & ~P(issue, cat1, cat2)

Symmetry

(4) I(issue, cat1, cat2) implies I(issue, cat2, cat1)

Reflexive

(5) I(issue, cat1, cat1)

Transitivity

(6) I(issue, cat1, cat2) & I(issue, cat2, cat3) implies I(issue, cat1, cat3)

(7) P(issue, cat1, cat2) & I(issue, cat2, cat3) implies P(issue, cat1, cat3)

(8) P(issue, cat1, cat2) & P(issue, cat2, cat3) implies P(issue, cat1, cat3)

(9) I(issue, cat1, cat2) & P(issue, cat2, cat3) implies P(issue, cat1, cat3)

The first five rules check for consistency of data. The next four transitivity rules derive additional preference and indifference statements.

4.5.2 Example of Fabricator Agent Model

Figure 4-2 is an example of the fabricator model with preferences which were derived from the current DFI connection database. Five connection categories are included. The first half of the model lists all of the preferences that could be taken from the connection database. Its content is similar to the earlier example. The first eight statements list category preferences, the next nine statements list material preferences, and the next two statements list issue priorities. The second half of the model represents derived statements, which fill in the gaps of the missing data. These additional preferences were derived using the transitivity rules listed above.

For ease of reading the model and understanding the derived statements, the categories are represented with these names:

category 1 - FB/SW/FB/SW

category 2 - SW/FB/SW/FB

category 3 - SW/FW/SW/FB

category 4 - FW/FW/FW/FW

category 5 - FW/FW/SW/FB

FB = field bolt
FW = field weld
SB = shop bolt
SW = shop weld

Fabricator

P(fab_cost, category1, category4)
P(fab_cost, category4, category2)
I(fab_cost, category4, category5)
I(fab_cost, category2, category3)

P(fab_ease, category5, category2)
P(fab_ease, category2, category3)
I(fab_ease,category2, category1)
I(fab_ease, category3, category4)

MP(category2, mat_cost, plate, tee)
MI(category2, fab_cost, plate, tee)
MI(category2, fab_ease, plate, tee)

MP(category5, mat_cost, plate, angle)
MP(category5, mat_cost, angle, tee)
MI(category5, fab_ease, plate, angle)
MI(category5, fab_ease, angle, tee)
MI(category5, fab_cost, plate, angle)
MI(category5, fab_cost, angle, tee)

IP(fab_cost, fab_ease)
II(fab_cost, mat_cost)

Derived statements concerning categories via transitivity rules

P(fab_cost, category1, category2)
P(fab_cost, category1, category5)
P(fab_cost, category4, category3)
P(fab_ease, category5, category3)
P(fab_ease, category5, category1)
P(fab_ease, category2, category4)

Figure 4-2

Again, the connections database representation reflects both these categories and materials, so preferences on specific connections can be obtained. See Appendix D for a complete listing of the connection database.

4.5.3 Model Responses

There are a variety of queries that can be asked of the agents. In this representation, the type of query and response is somewhat limited. Responses given will always be comparisons with another category. Single choice queries can still be made, but the response is in relation to other categories.

The examples given show only the system response which is passed to the meta-level for processing. It is anticipated that a textual response would be supplied to the user as well, i.e. fabricator prefers category1 to category2 with respect to fab_cost. Explanatory comments would supply the reason for the preference.

Single Choice

The only single choice request that might be useful is to get all preferences and indifferences with respect to one category of interest. Although the responses compare category1 to others, it helps put category1 in perspective among the other possibilities. This query and response follows:

```
query(fabricator, category1)
response(fabricator)
    [P(fab_cost, category1,category2)
    P(fab_cost, category1, category5)
    P(fab_cost, category1, category4)
    I(fab_ease,category2, category1)
    P(fab_ease, category5, category1)]
```

Multiple Choice

Most of the queries to the model by the designer will be about multiple categories under consideration. What follows is a query regarding two categories with respect to an issue and the response:

```
query(designer, strength, category1, category2)
response(designer)
    [P(strength, category2, category1)]
```

Of course queries about more than two categories can be made in the same way. In this case, preference or indifferences for each category would be returned. If no relation exists between these categories, no statements would be returned.

A similar type of query can be made regarding detail material. This query and response follows:

query(fabricator, mat_cost category2, plate, tee)
response(fabricator)
    [P(mat_cost, category2, plate, tee)]

The last query that could be made is about the priorities of two issues. This could be done when there are conflicting preferences, and a decision needs to be made according to which issues have the highest priority. This query and response follows:

query(erector, erec_ease, erec_cost)
response(erector)
    [P(erec_cost, erec_ease)]

4.5.4 Advantages

As can be seen by the example model, there is quite a bit less work in data gathering on the part of the knowledge engineer. The model is completed by the system by first checking the integrity of the statements listed, and then deriving additional preferences which fill in the gaps.

There is also less work involved on the part of the expert, who now needs only to supply preferences of one category over another. These preferences would most likely be agreed upon among the experts, whereas the rankings probably would not.

Also with this format, it is more likely that rules could be written to generate all preferences. This is desirable since it supplies the reasoning behind the preference statements, lending the model more credibility and supplying an automatic generation of explanation text.

4.5.5 Disadvantages

This model is less "complete," with less information available. There are gaps in the knowledge, i.e. cases where no relation exists between two categories of interest. There are also less queries that can be made to the model. Therefore, more meta-level decision-making strategies must be incorporated into the system to assist when no relations exist.

Another disadvantage of preference statements is that there is no quantitative measurement associated with them. These measurements also supply a degree of acceptability which can be important in decision-making.

4.6 Summary

This chapter provided two proposed representations for agent models with advantages and disadvantages of each. The decision on model representation will be driven by domain data, although preference relations are recommended when data is not well defined. These representation schemes will be investigated in future work.

5. Summary and Future Extensions

This chapter provides a summary of the thesis along with some ideas for future extensions to the models. The future extensions discussed include communication between agents, the addition of rules to formulate preferences, the addition of viewpoints that can represent a bias of a specific agent, and a similarity measure that can measure how alike connection configurations are.

5.1 Summary

This thesis began with a general description of user models, and a review of the systems that can be enhanced by exploiting models. One system which was described was that of cooperative problem-solving, which represents its users as distinct agents (Chapter 1). An overview of DFI, a cooperative problem-solving system, followed with a description of the connection evaluation process (Chapter 2). The next chapter proposed a new architecture for DFI which addresses some of the deficiencies discussed in the previous chapter (Chapter 3). Next, examples of agent models with two different representation schemes were discussed (Chapter 4).

The goal of this thesis was to describe the wide variety of user models being used by many different systems, and how these ideas can be adapted to form agent models that assist in problem-solving. The user model found in most expert systems and tutoring systems serves only to enhance dialog between system and user, or to help tailor explanation text to the user's level of expertise. Within DFI, there was a need to represent knowledge and views of three distinct agents, in order to consider during preliminary design the issues associated with downstream construction processes. These viewpoints are represented by a stereotypical model that attempts to capture the preferences of most designers, fabricators, and erectors. As in most systems employing models, a static stereotypical model provided a good foundation, but was not sufficient to handle special circumstances, especially the many peculiar situations that can arise in connection design. The concept of "hard constraints" was added which represents specific details that cannot be compromised. These hard constraints overrule existing agent preferences.

Besides hard constraints, there is a need to represent an agent bias which, for the current evaluation session only, differs from that found in the stereotypical model. One way of representing this temporary view is by introducing the notion of a viewpoint. How a viewpoint is represented and used is discussed further in future extensions.

Thus in DFI, the dynamic situation surrounding connection design needs to be represented as well as the dynamic aspects of the user. The overall goal of the models within DFI differs somewhat from that of models in other systems in that they contribute directly to decision-making. The decision-making process within DFI has added complexity because of its multi-agent scheme. The agent models assist in the cooperative problem-solving process by solving subgoals, which are the result of a decomposition of the overall goal of connection design.

5.2 Future Extensions

5.2.1 Knowledge of and Communication Between Other Agents

As mentioned in Section 4.1, there may be a need for agents to communicate with each other for additional information needed during problem-solving. One situation that requires interagent communication is in the case of cross issues. A cross issue is an issue that is made up of several subissues, at least one of which is shared by another agent. An example of a cross issue is given in Figure 5-1.

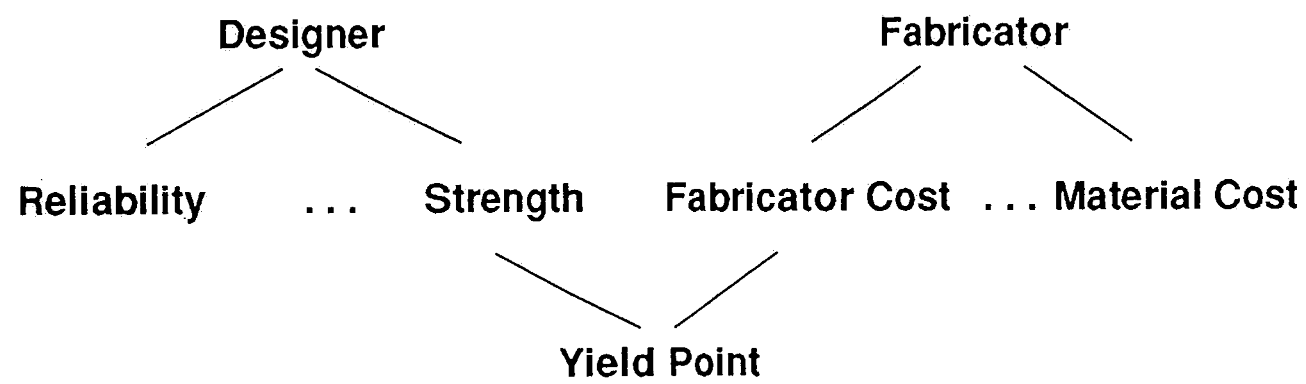


Figure 5-1

In this example the two cross issues, strength and fabrication cost, share the same subissue of yield point. In the case of such a cross issue, the agent must contain knowledge about what the other agents know, i.e. the fabricator must know that knowledge about yield point resides in the designer model. Secondly, the fabricator must query the designer for information about yield point before being able to form his preference on a configuration with respect to fabrication cost. It is anticipated that there are many such interdependencies among the issues in DFI.

5.2.2 Rules to Formulate Preferences

As mentioned in Section 4.5.4, it would be desirable to include a set of rules at the base level that would derive the preference statements. Rules of this type require a deeper knowledge than that contained in the current DFI system. Deeper knowledge refers to the reasoning behind a preference, what makes Category 1 preferable to Category 2 with respect to fabrication cost.

Although this level of knowledge was not available at the time this thesis was written, a study of the current data and discussions with Ira Hooper seemed to support these four general rules [Hooper 91].

Connections with field-welded components are preferable to those without field welding with respect to stiffness.

Connections that are bolted are preferable to connections that are not bolted with respect to reliability.

Connections that are not field-welded are preferable to those that are field welded with respect to erection ease and safety.

Connections that are field-bolted are preferable to those that are not field bolted with respect to erection cost.

These types of rules can be used to fill in the gaps when no specific preference relations exist between two categories of interest. The existence of these rules would also lend the preferences a degree of credibility by supplying the reason that such a preference exists.

5.2.3 Viewpoints

Since the agent models represent a static viewpoint considered stereotypical of that agent, it may be necessary to represent a temporary viewpoint or perspective other than that contained by the model

Kathy McCoy has done much work in creating different perspectives within the ROMPER System, a natural language interface to an expert system that contains knowledge about investment securities [McCoy 85]. Within ROMPER, objective perspectives were created which highlight a certain group of attributes while suppressing others. This highlighting is achieved by creating a perspective which assigns salience or importance values to object attributes. Thus, a precoded perspective will assign higher salience values to attributes viewed as important, and will assign lower salience values to attributes viewed as less important.

This notion of perspective can be applied to connections as well. The objects are the connections in the database, and the features used to describe them are the attributes. When no specific viewpoint is enabled, the salience values of all features are zero, indicating that salience is not being used. However, with the instantiation of a specific viewpoint, salience values of 1 or .5 are assigned to features to highlight their importance, while those of lesser importance remain 0. An example of a temporary bias is that of a fabricator shop which has a specialty involving shop welding. A viewpoint is then created which assigns salience values of 1 to the shop-welding attribute of connections, while the other non-shop-weld attributes remain at 0. This viewpoint is in effect only during this evaluation session, and overrides the default stereotypical viewpoint contained in the agent model.

5.2.4 Similarity Measure

In any cooperative problem-solving situation, it would be useful to know if the agents are close to reaching a consensus. This could be accomplished by determining how similar the objects are that the agents prefer.

McCoy also introduces in her work Tversky's similarity measure [McCoy 85]. It works by looking at the two objects of interest and comparing the attributes that they have in common and the attributes that are distinct, and assigning a final value that is a measure of how similar they are.

The Tversky metric is as follows [Tversky 77]:

$$s(a,b) = \Theta f(A \cap B) - \alpha f(A - B) - ß f(B-A)$$

This metric states that the similarity of two objects is some function of their common features minus some function of their disjoint features. In this equation, Θ, α, and ß are parameters which alter the importance of each part of the equation, depending upon the application. Compare the connection objects listed below with f being the length of the list and assuming Θ, α, and ß to be 1.

Object 1	Object 2	Object 3
SB FW SB FW	SB FW SB FB	FB SB SB FB

s(1,2) = (SB SB FW) - (FW) - (FB) = 3 -1 -1 = 1

s(1,3) = (SB SB) - (FW FW) - (FB FB) = 2 - 2 - 2 = -2

Based on the above equation, we can conclude that Object 2 is more similar to Object 1 than Object 3 is to Object 1. Note that we are comparing only two objects at one time and using the largest number to select the most similar object.

This type of evaluation would assist decision-making at the meta-level, especially in the case of conflicts. If two agents have distinct preferences that are in conflict with each other, a similarity measure could help choose objects that were at least similar to those preferred by another agent.

References

[AISC 80]
AISC, Manual of Steel Construction, 8th ed., revised, American Institute of Steel Construction, 1980.

[Beale & Finlay 89]
Beale, Russell and Janet Finlay, "User Modeling with a Neural System," Proceedings of the Fourth International Symposium on Methodologies for Intelligent Systems: Poster Session Program, pp. 177-186, 1989.

[Barone 90]
Barone, Marcello, Designer Fabricator Interpretor: A Step Towards Computer Integrated Construction, Master's Thesis, Department of Civil Engineering, Lehigh University 1990.

[Burton 82a]
Burton, Richard R., Diagnosing Bugs in a Simple Procedural Skill, in Intelligent Tutoring Systems, D. Sleeman and J. S. Brown, eds., Academic Press, New York, 1982.

[Burton 82b]
Burton, Richard R. and John S. Brown, An Investigation of Computer Coaching for Informal Learning Activities, in Intelligent Tutoring Systems, D. Sleeman and J. S. Brown, eds., Academic Press, New York, 1982.

[Carberry 88]
Carberry, Sandra, "Modeling the User's Plans and Goals," Computational Linguistics, Vol. 14, pp. 23-37, September 1988.

[Carberry 83]
Carberry, Sandra, Tracking User Goals in an Information-Seeking Environment, Proceedings of the National Conference on AI, pp. 59-63, Washington, D.C., August 1983.

[Chin 88a]
Chin, David N., KNOME: Modeling What the User Knows in UC, in User Models in Dialog Systems, A. Kobsa, ed., Springer-Verlag, Berlin-New York, 1988.

[Chin 88b]
Chin, David N., "User Models and Discourse Models," Computational Linguistics, Vol. 14, pp. 86-88, September 1988.

[Cohen 88a]
Cohen, Robin, "On the Relationship between User Models and Discourse Models," Computational Linguistics, Vol. 14, pp. 88-90, September 1988.

[Cohen 88b]
Cohen, Robin, Producing User-Specific Explanations, Proceedings of the AAAI 88 Workshop on Explanations, pp. 44-47, August 1988.

[Daniels 86]
Daniels, Penny J., The User Modelling Function of an Intelligent Interface for Document Retrieval Systems, in Intelligent Information Systems for the Information Society, B. Brookes, ed., North-Holland, Amsterdam, 1986.

[Durfee et. al. 87]
Durfee, Edmund H., Victor R. Lesser, and Daniel D. Corkill, Cooperation Through Communication in a Distributed Problem-Solving Network, in Distributed Artificial Intelligence, Morgan Kaufmann Publishers, Inc., Los Altos, CA, 1987.

[Finin 88]
Finin, Timothy W., GUMS - A General User Modeling Shell, in User Models in Dialog Systems, A. Kobsa, ed., Springer-Verlag, Berlin-New York, 1988.

[Finlay & Beale 90]
Finlay, Janet and Russell Beale, "Neural Networks in Human-Computer Interaction: A View of User Modelling," IEE Colloquium Digest, pp. 179, 1990.

[Goldstein 82]
Goldstein, Ira P., The Genetic Graph: A Representation for the Evolution of Procedural Knowledge, in Intelligent Tutoring Systems, D. H. Sleeman and J. S. Brown, eds., Academic Press, New York, 1982.

[Greenberg 85]
Greenberg, S. and I. H. Witten, "Adaptive Personalized Interfaces - A Question of Viability," Behavior and Inform. Tech., Vol. 4, No. 1, pp. 31-34, 1985.

[Hooper 91]
Hooper, Ira, STV/Seelye Stevenson Value & Knecht, notes of meetings, September 1990 - April 1991.

[Huhns 87]
Huhns, Michael N., Distributed Artificial Intelligence, Morgan Kaufmann Publishers, Inc., Los Altos, CA, 1987.

[Kass & Finin 88]
Kass, Robert and Tim Finin, Acquiring User Models for Tailoring Explanations, Proceedings of the AAAI 88 Workshop on Explanation, pp. 51-54, August 1988.

[Kass & Finin 87]
Kass, Robert and Tim Finin, Rules for the Implicit Acquisition of Knowledge about the User Proceedings of the 6th National Conference on Artificial Intelligence, pp. 295-300, Seattle, Washington, July 1987.

[Kidd 88]
Kidd, Alison, Cooperative Problem Solving as a Framework for Describing and Understanding the Role of Explanations, Proceedings of the AAAI 88 Workshop on Explanations, pp. 34-36, 1988.

[Kobsa & Wahlster 89]
Kobsa, A. and W. Wahlster, User Models in Dialog Systems, Springer-Verlag, New York, NY, 1989.

[Kobsa 88]
Kobsa, Alfred, "User Models and Discourse Models: United they Stand," Computational Linguistics, Vol. 14, pp. 91-94, September 1988.

[Mason & Thomas 84]
Mason, M. V. and R. C. Thomas, "Experimental Adaptive Interface," Information Technology: Research, Design, Applications, Vol 3., No. 3, pp. 162-167, 1984.

[McCoy 88]
McCoy, Kathleen F., "Reasoning on a Highlighted User Model to Respond to Misconceptions," Computational Linguistics, Vol. 14, pp. 52-63, September 1988.

[McCoy 85]
McCoy, Kathleen, F., Responding to User's Misconceptions, Ph.D dissertation, University of Pennsylvania, Philadelphia, PA, 1985.

[McKeown & Weida 88]
McKeown, Kathleen R. and Robert A Weida, Highlighting User Related Advice, Proceedings of the AAAI 88 Workshop on Explanation, August 1988.

[McKeown et al 85]
McKeown, Kathleen R., Myron Wish, and Kevin Matthews, Tailoring Explanations for the User, Proceedings of the 9th International Joint Conference on Artificial Intelligence, pp. 794-798, Los Angeles, CA, August 1985.

[Morik 88]
Morik, Katharina, "Discourse Models, Dialog Memories, and User Models," Computational Linguistics, Vol. 14, pp. 95-97, September 1988.

[Morik & Rollinger 85]
Morik, Katharina, and Claus-Ranier Rollinger, "The Real Estate Agent - Modeling Users by Uncertain Reasoning," AI Magazine, pp. 44-52, summer 1985.

[Norcio & Stanley 89]
Norcio, Anthony F. and Jaki Stanley, "Adaptive Human-Computer Interfaces: A Literature Survey and Perspective," IEEE Transactions on Systems, Man and Cybernetics, Vol. 19, No. 2, March/April 1989.

[Paris et al 88]
Paris, Cecile L., Michael R. Wick and William B. Thompson, The Line of Reasoning Versus the Line of Explanation, Proceedings of the AAAI 88 Workshop on Explanation, pp. 4-7, 1988.

[Rankin et al 88]
Rankin, Ivan, Sture Hagglund, and Yvonne Waern, Generating a User-Centered Explanations in a Critiquing Context, Proceedings of the AAAI 88 Workshop on Explanation, pp. 48-50, August 1988.

[Rich 83]
Rich, Elaine, "Users are Individuals: Individualizing User Models," International Journal of Man-Machine Studies, Vol. 18, pp. 199-214, 1983.

[Rich 79a]
Rich, Elaine, Building and Exploiting User Models, Ph.D dissertation, Carnegie-Mellon University, Pittsburgh, PA, 1979.

[Rich 79b]
Rich, Elaine, "User Modeling via Stereotypes," Cognitive Science, Vol. 3, pp. 329-354, 1979.

[Robertson et al 81]
Robertson, G., D. McCracken and A. Newell, "The Zog Approach to Man-Machine Communication," International Journal Man-Machine Studies, Vol. 14, pp. 461-488, 1981.

[Schuster 88]
Schuster, Ethel, "Establishing the Relationship between Discourse Models and User Models," Computational Linguistics, Vol. 14, pp. 82-85, September 1988.

[Shaw 90]
Shaw, Michael J., Mechanisms for Cooperative Problem Solving and Multi-Agent Learning in Distributed Artificial Systems, Proceedings of the 10th International Workshop on Distributed Artificial Intelligence, Bandera, TX, October 1990.

[Sleeman 85a]
Sleeman, D., "UMFE: A User Modeling Front-End Subsystem," International Journal of Man-Machine Studies, Vol. 23, pp. 71-88, 1985.

[Sleeman et al 85b]
Sleeman, D., Doug Appelt, Kurt Konolige, Elaine Rich, N. S. Sridharan, and Bill Swartout, User Modeling Panel, Proceedings of the 9th International Joint Conference on.AI, pp. 1298-1302, Los Angeles, Ca, August 1985.

[Sparck Jones 88]
Sparck Jones, Karen, "User Models, Discourse Models, and Some Others," Computational Linguistics, Vol. 14, pp. 98-100, September 1988.

[Sparck Jones 86]
Sparck Jones, Karen, Issues in User Modelling for Expert Systems, in Artificial Intelligence and its Application, John Wiley & Sons Ltd., New York, 1986.

[Suthers 88]
Suthers, Dan, Providing Multiple Views for Explanation, Proceedings of the AAAI 88 Workshop on Explanation, pp. 22-25, 1988.

[Tversky 77]
Tversky, A., "Features of Similarity," Psychological Review, 84:327-352, 1977.

[Van Lehn 88]
VanLehn, Kurt, Student Modeling, in Intelligent Tutoring Systems, M. C. Polson and J. J.Richardson, eds., Lawrence Erlbaum Assoc., Inc. 1988.

[Vilnat 85]
Vilnat, Anne, Be Brief, Be to the Point...Be Seated or Relevant - Responses in Man/Machine Conversation, Proceedings of the 9th International Joint Conference on AI, pp. 852-854, Los Angeles, Ca, August 1985.

[Wahlster 88]
Wahlster, Wolfgang, "Distinguishing User Models from Discourse Models," Computational Linguistics, Vol. 14, pp. 101-103, September 1988.

[Werkman 90]
Werkman, Keith J., Multiagent Cooperative Problem-Solving Through Negotiation and Sharing of Perspectives, Ph.D. Dissertation, Department of Computer Science and Electrical Engineering, Lehigh University, 1990.

[Werkman et. al. 90]
Werkman, K. J., M. Barone, S. J. Wagaman, D. J. Hillman, and J. L. Wilson, Design and Fabrication Problem-Solving Through Cooperative Agents, NSF-ERC ATLSS Report 90-05, Lehigh University, 1990.

[Wick and Thompson 89]
Wick, Michael R. and William B. Thompson, Reconstructive Explanation: Explanation as Complex Problem Solving, Proceedings of the 11th International Joint Conference on AI, pp. 135-140, August 20-25, 1989.

[Wong 91]
Wong, S. T. C., "An Approach to Cooperation among Specialist Agents in Construction Life-Cycle Process," internal memorandum, Lehigh University, January 29, 1991.

[Yallow 80]
Yallow, E., Individual Differences in Learning from Verbal and Figure Materials, School of Education, Stanford University, Aptitudes Research Project Tech. Report No. 13, Palo Alto, CA, 1980.

APPENDICES

APPENDIX A: Example of DFI Connection Evaluation

APPENDIX B: Feasible Connection Combinations

APPENDIX C: Extended DFI Inter-Agent Relations

APPENDIX D: Revised DFI Connection Database

A. Example of DFI Connection Evaluation

This section presents a representative example of DFI's connection evaluation process. An endplate connection was selected as the starting point for this evaluation. This connection configuration was chosen because of its inherent erection problems. The test case demonstrates, in part, DFI's ability to recognize the erection problem and suggest "better" alternatives.

After the user enters a connection and selects the evaluate option, he is asked for a single, most important key issue which is maintained by all agents during their proposal of alternate connection configurations. In this example, the key issue is strength.

Each agent has unique knowledge about connections including a standardized qualitative rating scheme for the issues related to each connection. The higher the value, the more acceptable it is. The agents suggest alternative connections that are of the same connection type and have the same or higher value for strength as the user specified endplate connection. Initially, the arbitrator **commands** the design agent to **accept** the user's endplate connection using strength as the positive supporting issue in the first cycle of negotiation. The design agent then **informs** all agents of the key issue and requests that the proposed connection be evaluated. The designer's request is shown graphically in the designer's window in Figure A-1.

Before each agent's evaluation, the arbitrator reviews all proposed connections and selects the most detrimentally affected agent to go next. In this case, the erector is the

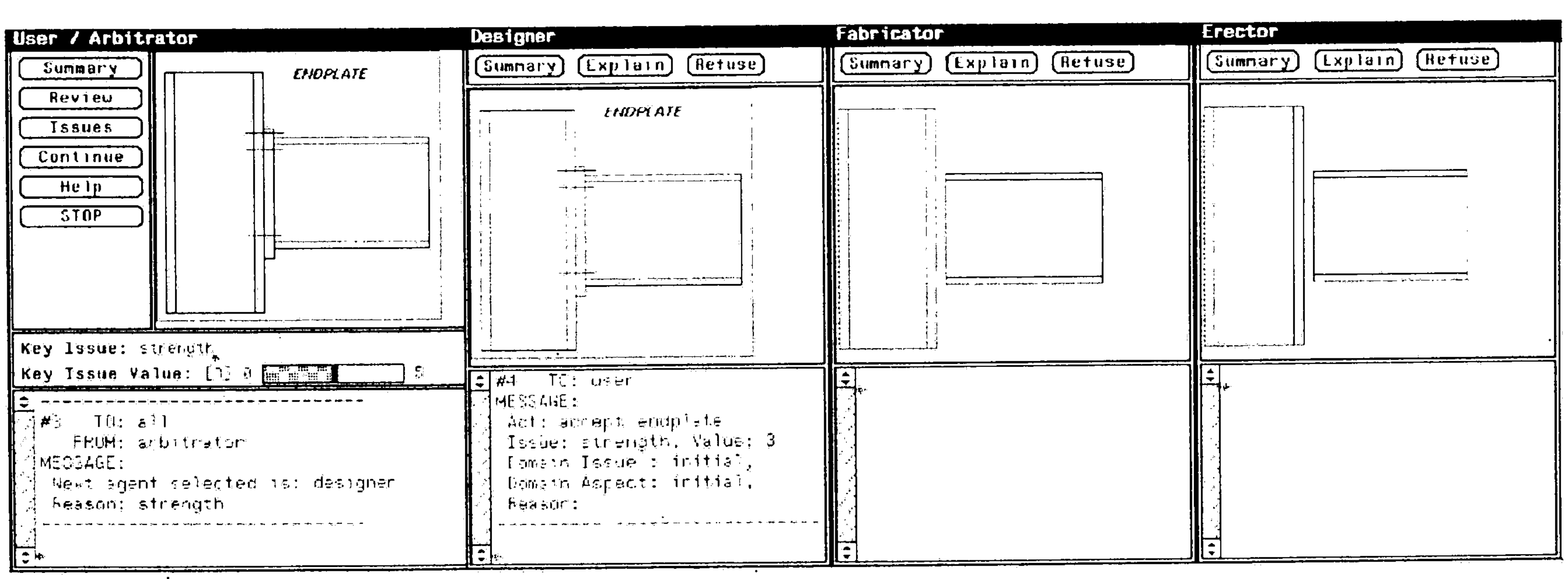


Figure A-1: Designer Accepts the User's Connection worst affected by the designer's endplate proposal. The erector determines that the designer's proposal is unacceptable because the endplate connection has a low value for erection ease. Therefore, the erector **refuses** (objects to) the designer's connection and then checks the fabricator's connection (this is a default action of the system). The evaluating agent checks with both of the other agents to determine if their connections are acceptable to him. At this stage, the fabricator has no connection proposal, so the erector selects a connection configuration from the connection database. The erector **requests** the plates_tee (top and bottom plates with shear tee) because it satisfies the erection ease issue as well as the user specified key issue. This proposal is shown in the erector's window in Figure A-2.

It is important to note that the erector has directed the proposed connection back to the designer for review. The designer **accepts** the erector's proposal because it exceeds the key issue of strength. Also, the value of the key issue has been increased to 4. The value as-

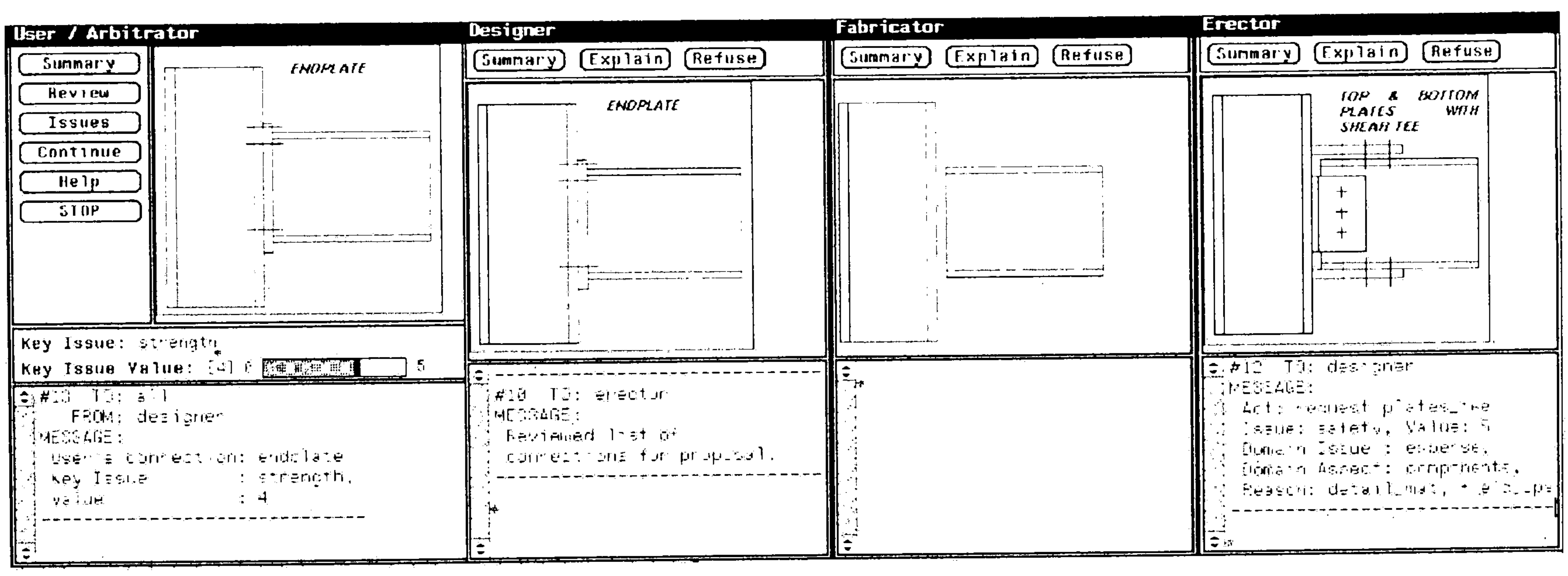


Figure A-2: Erector Proposes plates_tee Connection

sociated with the erector's proposed plates_tee connection is higher than the original value of 3 for the designer's strength key issue on the endplate connection. By increasing the value of the key issue, the search space of possible connection alternatives is reduced, thus causing the agents to converge more quickly on a set of acceptable connections. The designer's acceptance is seen graphically in the designer's window in Figure A-3.

Next, the arbitrator reviews the agent proposals and notices that two agents have proposed the same connection. Usually, this would cause the arbitrator to **inform** all agents of a halting (agreement) condition. This is not the case here because an "unfair" evaluation

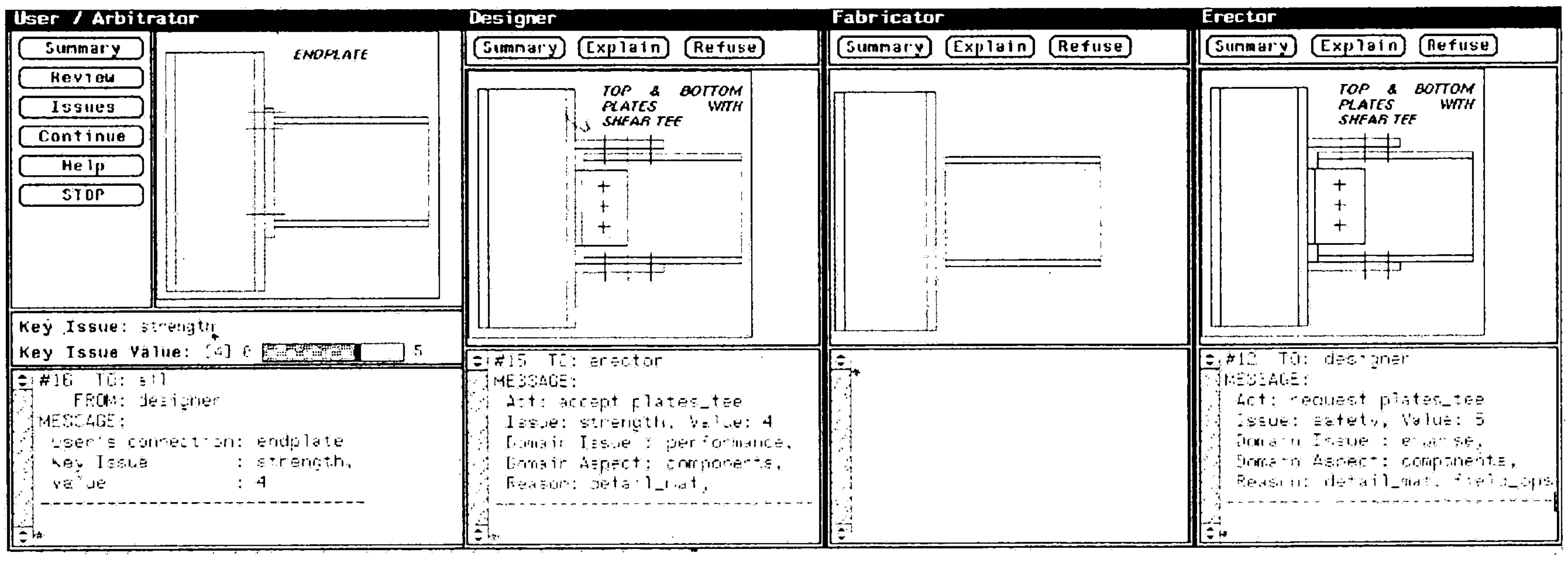


Figure A-3: Designer Accepts Erector's Proposal

has occurred - unfair in the sense that the fabricator has not yet had a chance to contribute to the evaluation. Thus, the arbitrator gives control to the fabricator who looks at the designer's connection and immediately notices that material cost is the problem issue. Since both the designer's and erector's connection are the same, the fabricator needs only to review the plates_tee connection and propose an alternative. In this case, the best connection, from the fabricator's viewpoint, that maintains the key issue value of strength as well as improving the fabricator's material cost issue is the flange_weld_plate (direct flange weld with shear plate) connection as seen in the fabricator's window in Figure A-4.

Again, the arbitrator reviews the evaluation process and notices that two agents have agreed on a connection, and that each agent has had a chance at proposing an alternative. There is also the possibility that an agent may not be able to suggest an alternative. The arbitrator **informs** the agents of a halting condition and control is returned to the user. At this point the user can ask any agent to **explain** its proposed connection or **continue**

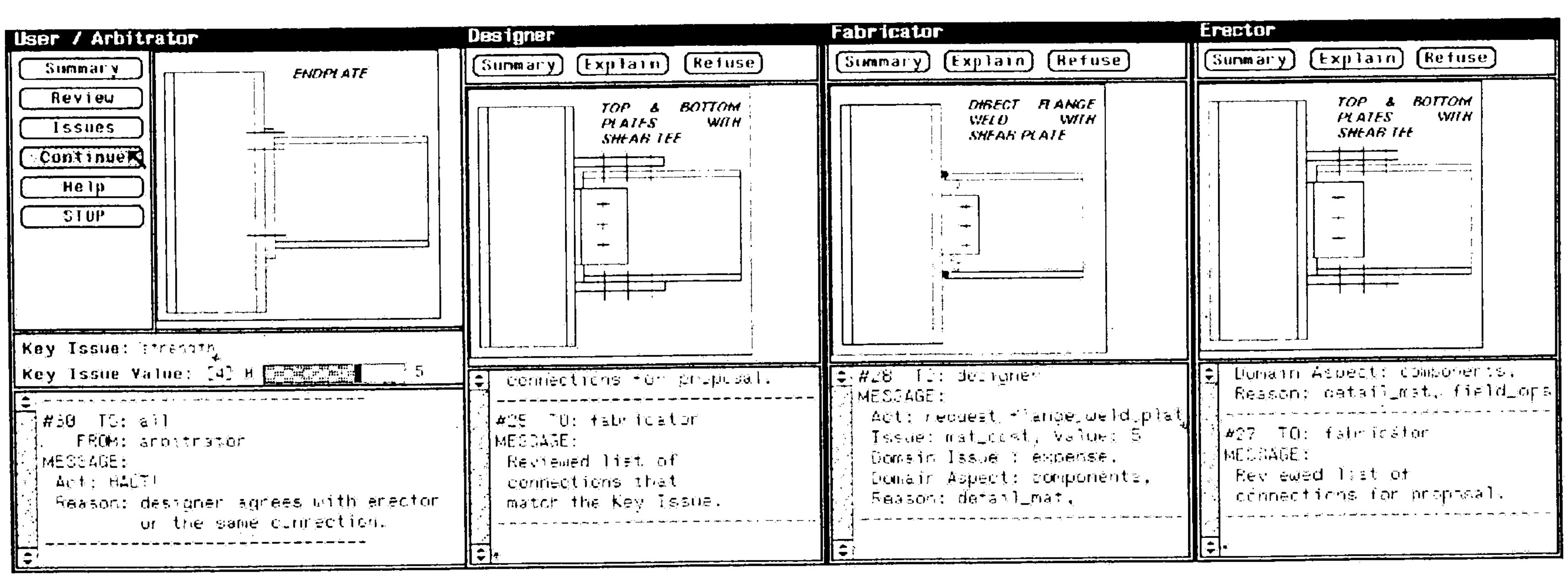


Figure A-4: Fabricator Proposes flange_weld_plate Connection with the evaluation. If the user **continues**, the arbitrator reviews the situation and notices that no particular agent is in "peril." Therefore, whichever agent received the last message is given a chance to respond to it. In this case, the fabricator proposed a connection to the designer. The design agent, upon reviewing this connection, notices that the fabricator's connection is also acceptable. Thus, the designer **accepts** the fabricator's proposed flange_weld_plate connection as seen in the designer's window in Figure A-5.

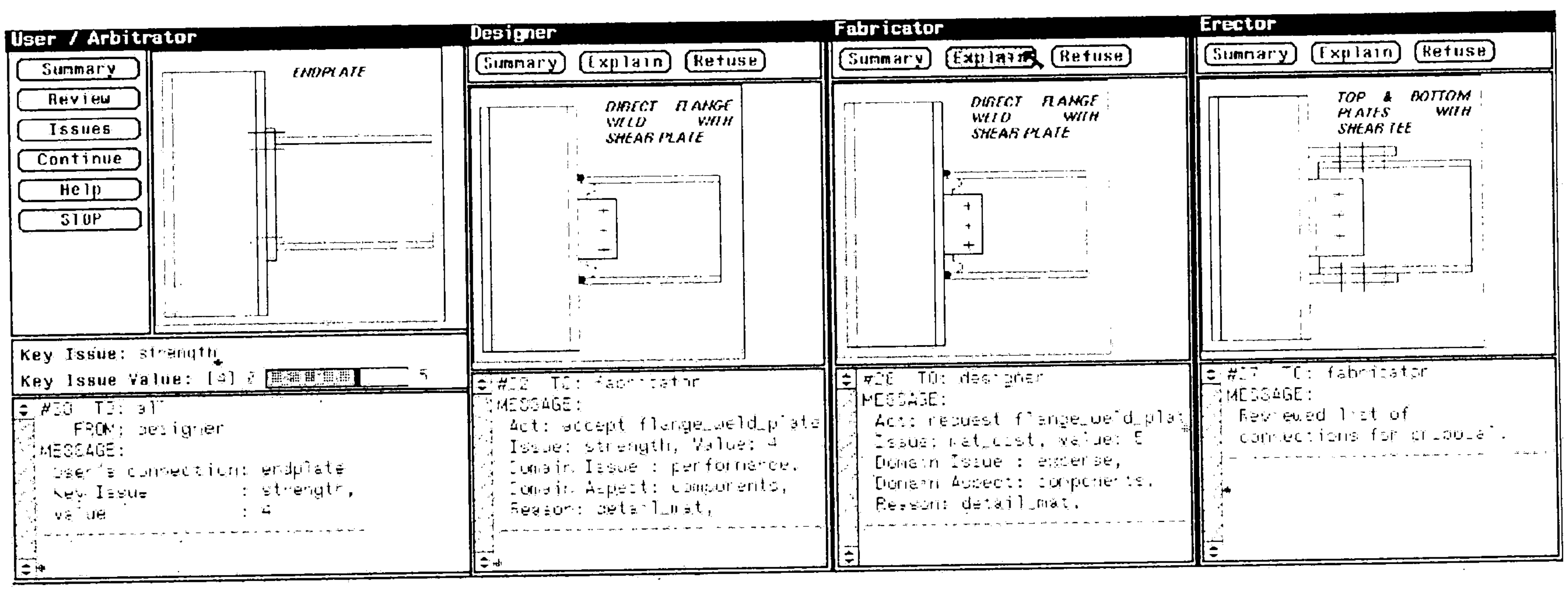


Figure A-5: Designer Accepts Fabricator's Proposal

Once again two agents agree on the same connection, thus causing another halting condition. The arbitrator returns control of the system back to the user, who makes the final decision based on the connection alternatives shown here.

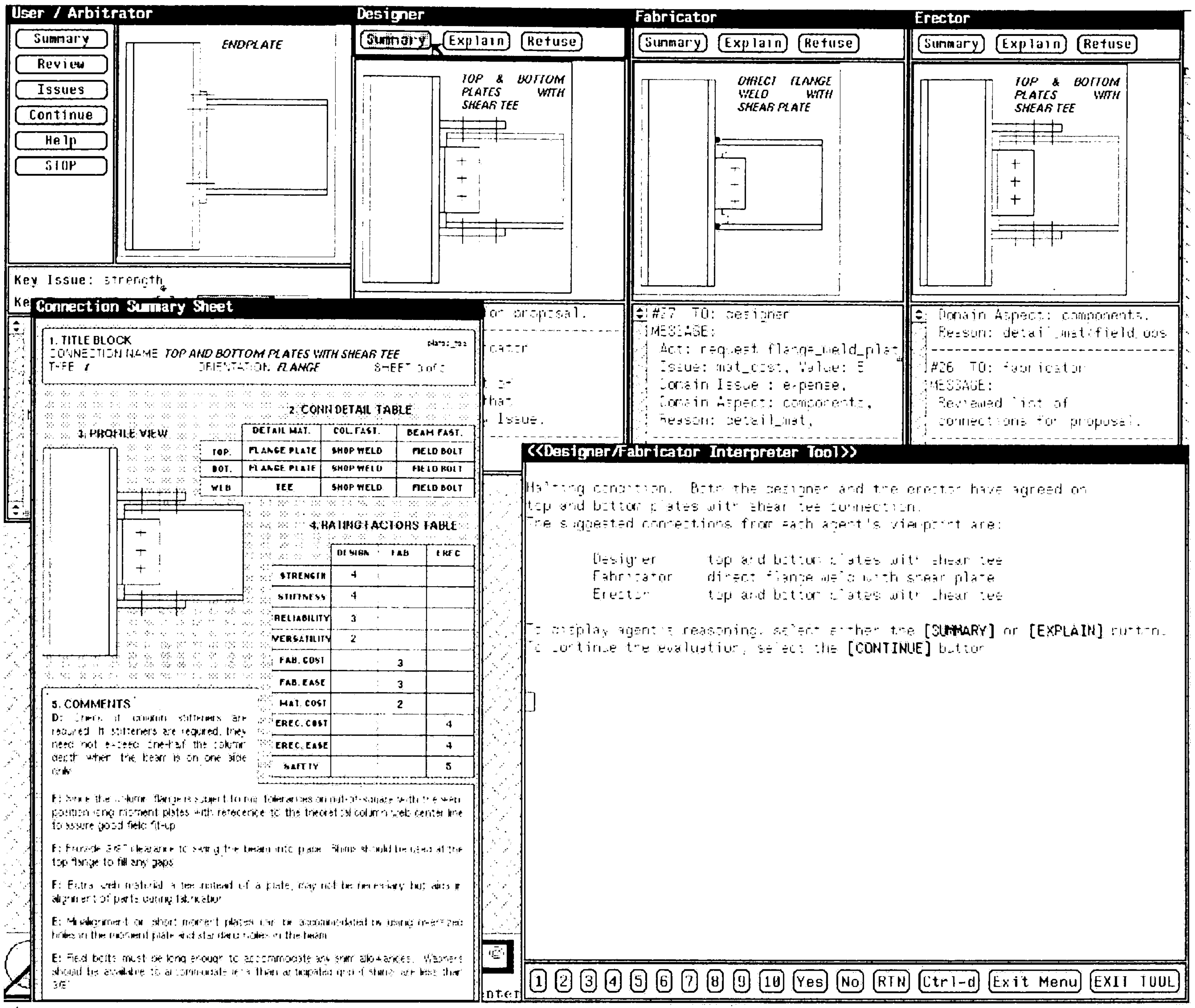


Figure A-6: Halting Condition with a Connection Summary Sheet

To assist the user in his decision-making, he can request a connection summary sheet of any of the connection proposed by the agents. Figure A-6 shows a summary sheet of the designer's connection.

The user can also request an explanation on any agent, which will summarize the actions of the agent during that evaluation. Figure A-7 shows this explanation on the fabricator agent.

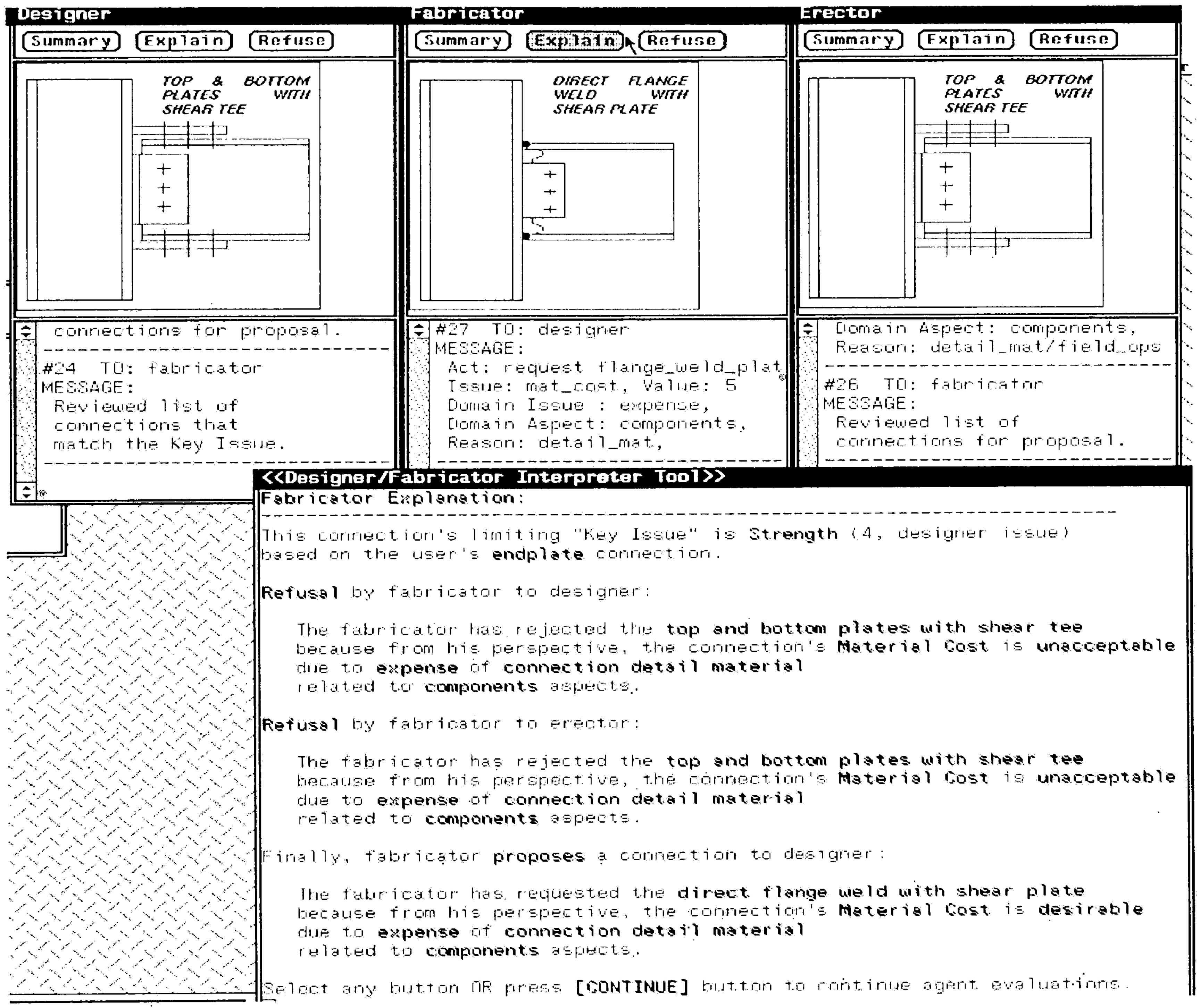


Figure A-7: Fabricator's Output of Explain Button

B. Feasible Connection Combinations

A list of 81 different connection categories was derived as a basis for the new connections database. The connection operations are divided into four categories - column flange moment, beam flange moment, column flange shear, and beam web shear. There are also four choices for each connection, i.e. shop bolt, shop weld, field bolt and field weld.

We started by considering all combinations, which is 4^4 for a total of 256. To this list of 256 combinations, feasibility rules were applied. These rules state that at the moment or shear connection, if one operation is a shop operation, then the second operations must be a field operation. For example, if shop bolt is the method of operation at the column flange moment, then the beam flange moment must be a field operation, either field bolt or field weld. The only exception to the rule is a field welded connection, which can accompany another field welded connection. This is a common practice in the industry. The same set of rules are applied to the shear connection as well.

By applying these feasibility constraints, the original list of 256 was limited to 81 feasible connection combinations listed below.

	COLUMN FLANGE MOMENT	BEAM FLANGE MOMENT	COLUMN FLANGE SHEAR	BEAM WEB SHEAR
1	shop bolt	field bolt	shop bolt	field bolt
2	shop bolt	field bolt	shop bolt	field weld
3	shop bolt	field bolt	field bolt	shop bolt
4	shop bolt	field bolt	field bolt	shop weld
5	shop bolt	field bolt	shop weld	field bolt
6	shop bolt	field bolt	shop weld	field weld
7	shop bolt	field bolt	field weld	shop bolt
8	shop bolt	field bolt	field weld	shop weld
9	shop bolt	field bolt	field weld	field weld
10	shop bolt	field weld	shop bolt	field bolt
11	shop bolt	field weld	shop bolt	field weld
12	shop bolt	field weld	field bolt	shop bolt
13	shop bolt	field weld	field bolt	shop weld
14	shop bolt	field weld	shop weld	field bolt
15	shop bolt	field weld	shop weld	field weld

	COLUMN FLANGE MOMENT	BEAM FLANGE MOMENT	COLUMN FLANGE SHEAR	BEAM WEB SHEAR
16	shop bolt	field weld	field weld	shop bolt
17	shop bolt	field weld	field weld	shop weld
18	shop bolt	field weld	field weld	field weld
19	field bolt	shop bolt	shop bolt	field bolt
20	field bolt	shop bolt	shop bolt	field weld
21	field bolt	shop bolt	field bolt	shop bolt
22	field bolt	shop bolt	field bolt	shop weld
23	field bolt	shop bolt	shop weld	field bolt
24	field bolt	shop bolt	shop weld	field weld
25	field bolt	shop bolt	field weld	shop bolt
26	field bolt	shop bolt	field weld	shop weld
27	field bolt	shop bolt	field weld	field weld
28	field bolt	shop weld	shop bolt	field bolt
29	field bolt	shop weld	shop bolt	field weld
30	field bolt	shop weld	field bolt	shop bolt
31	field bolt	shop weld	field bolt	shop weld
32	field bolt	shop weld	shop weld	field bolt
33	field bolt	shop weld	shop weld	field weld
34	field bolt	shop weld	field weld	shop bolt
35	field bolt	shop weld	field weld	shop weld
36	field bolt	shop weld	field weld	field weld
37	shop weld	field bolt	shop bolt	field bolt
38	shop weld	field bolt	shop bolt	field weld
39	shop weld	field bolt	field bolt	shop bolt
40	shop weld	field bolt	field bolt	shop weld
41	shop weld	field bolt	shop weld	field bolt
42	shop weld	field bolt	shop weld	field weld
43	shop weld	field bolt	field weld	shop bolt
44	shop weld	field bolt	field weld	shop weld
45	shop weld	field bolt	field weld	field weld
46	shop weld	field weld	shop bolt	field bolt
47	shop weld	field weld	shop bolt	field weld
48	shop weld	field weld	field bolt	shop bolt
49	shop weld	field weld	field bolt	shop weld
50	shop weld	field weld	shop weld	field bolt
51	shop weld	field weld	shop weld	field weld
52	shop weld	field weld	field weld	shop bolt
53	shop weld	field weld	field weld	shop weld
54	shop weld	field weld	field weld	field weld

	COLUMN FLANGE MOMENT	BEAM FLANGE MOMENT	COLUMN FLANGE SHEAR	BEAM WEB SHEAR
55	field weld	shop bolt	shop bolt	field bolt
56	field weld	shop bolt	shop bolt	field weld
57	field weld	shop bolt	field bolt	shop bolt
58	field weld	shop bolt	field bolt	shop weld
59	field weld	shop bolt	shop weld	field bolt
60	field weld	shop bolt	shop weld	field weld
61	field weld	shop bolt	field weld	shop bolt
62	field weld	shop bolt	field weld	shop weld
63	field weld	shop bolt	field weld	field weld
64	field weld	shop weld	shop bolt	field bolt
65	field weld	shop weld	shop bolt	field weld
66	field weld	shop weld	field bolt	shop bolt
67	field weld	shop weld	field bolt	shop weld
68	field weld	shop weld	shop weld	field bolt
69	field weld	shop weld	shop weld	field weld
70	field weld	shop weld	field weld	shop bolt
71	field weld	shop weld	field weld	shop weld
72	field weld	shop weld	field weld	field weld
73	field weld	field weld	shop bolt	field bolt
74	field weld	field weld	shop bolt	field weld
75	field weld	field weld	field bolt	shop bolt
76	field weld	field weld	field bolt	shop weld
77	field weld	field weld	shop weld	field bolt
78	field weld	field weld	shop weld	field weld
79	field weld	field weld	field weld	shop bolt
80	field weld	field weld	field weld	shop weld
81	field weld	field weld	field weld	field weld

C. Extended DFI Inter-Agent Relations

Following is a list of inter-agent relationships, which was expanded from the original list to include the subissue level. These 38 relations were developed using the original relations plus the issue decomposition work found in [Barone 90]. Within the DFI system, these relations were used to develop enhanced explanation text.

Preceding the actual relations is a template of the relation followed by a list of what each item can contain.

EXAMPLE RELATION:

```
relation(Domain_Perspective,
[agent_issue(Agent1, Issue1, Subissue1, Aspect1,
        [property_name1(prop_link,[Keywords_for_Explanation])]),
 agent_issue(Agent2, Issue2, Subissue2, Aspect2,
        [property_name2(prop_link,[Keywords_for_Explanation])]) ).
```

Domain_Perspective: {expense,performance}

Agent: {designer, fabricator, erector}

Issue: {strength, stiffness, reliability, versatility,
fab_cost, fab_ease, mat_cost,
erec_cost, erec_ease, safety}

Subissue: {structural concept, structural detailing, design methods, fabrication procedures, construction procedures, shipping operations, physical components, material properties}

Aspect: {functional, physical}

Property_name: building_topology, connection_design, construction_schedule, field_operations, shop_operations, structural_members, strength_criteria, serviceability_criteria, design_codes, analysis/assumptions, shipping_schedule, shipping_methods

Property_link: attribute, method, part, duration, measure, destination, instrument

Keywords_for_Explanation: See Appendix C of [Barone 90] which contains an extensive subissue decomposition with keywords. Specific keywords have been selected that were deemed to be the most appropriate for the specific relation.

Strength Relationships:

```
relation(performance,
1      [agent_issue(designer,strength,struc_det,functional,
            [connection_design(method,[bolt_properties, weld_properties,
                 detail_material, fastener_type])]),
        agent_issue(fabricator,fab_cost,fab_proc,physical,
            [construction_schedule(duration,[fabrication_schedule,
                 erection_schedule, stability_analysis]),
                 shop_operations(method,[cutting,drilling,welding,shop_assembly])])).

relation(performance,
2      [agent_issue(designer,strength,struc_det,functional,
            [connection_design(method,[bolt_properties, weld_properties,
                 detail_material, fastener_type])]),
        agent_issue(fabricator,fab_ease,fab_proc,physical,
            [construction_schedule(duration,[fabrication_schedule,
                 erection_schedule, stability_analysis]),
                 shop_operations(method,[cutting,drilling,welding,shop_assembly])])).


relation(performance,
3      [agent_issue(designer,strength,physical_comp,physical,
            [structural_members(part,[member_type, member_property]),
                 connection_design(part,[bolt_properties, weld_properties,
                 detail_material, fastener_type])]),
        agent_issue(erector,safety,struc_det,physical,
            [connection_design(method,[bolt_properties, weld_properties,
                 detail_material, fastener_type])])).

relation(performance,
4      [agent_issue(designer,strength,physical_comp,physical,
            [structural_members(part,[member_type, member_property]),
                 connection_design(part,[bolt_properties, weld_properties,
                 detail_material, fastener_type]) ]),
        agent_issue(erector,safety,con_proc,physical,
            [construction_schedule(duration,[fabrication_schedule,
                 erection_schedule,stability_analysis]),
                 field_operations(method,[fastening_methods,temporary_bracing])])).

relation(performance,
5      [agent_issue(designer,strength,struc_det,functional,
            [connection_design(method[bolt_properties, weld_properties,
                 detail_material, fastener_type]) ]),
        agent_issue(erector,erec_ease,con_proc,physical,
            [construction_schedule(duration,[fabrication_schedule,
                 erection_schedule,stability_analysis]),
                 field_operations(method,[fastening_methods,temporary_bracing])])).
```

```
relation(expense,
6     [agent_issue(designer,strength,physical_comp,physical,
            [structural_members(part,[member_type, member_property]),
                  connection_design(part,[bolt_properties, weld_properties,
                  detail_material, fastener_type])]),
      agent_issue(fabricator,mat_cost,physical_comp,physical,
            [structural_members(part,[member_type, member_property]),
                  connection_design(part,[bolt_properties, weld_properties,
                  detail_material, fastener_type])])).

relation(expense,
7     [agent_issue(designer,strength,struc_det,functional,
            [connection_design(method[bolt_properties, weld_properties,
                  detail_material, fastener_type]) ]),
      agent_issue(erector,erec_cost,con_proc,physical,
            [construction_schedule(duration,[fabrication_schedule,
                  erection_schedule,stability_analysis]),
                  field_operations(method,[fastening_methods,temporary_bracing])])).
```

Stiffness Relationships:

```
relation(performance,
8     [agent_issue(designer,stiffness,struc_det,functional,
            [connection_design(method[bolt_properties, weld_properties,
                  detail_material, fastener_type]) ]),
      agent_issue(fabricator,fab_cost,fab_proc,physical,
            [construction_schedule(duration,[fabrication_schedule,
                  erection_schedule,stability_analysis]),
                  shop_operations(method,[cutting,drilling,welding,shop_assembly])])).

relation(performance,
9     [agent_issue(designer,stiffness,struc_det,functional,
            [connection_design(method[bolt_properties, weld_properties,
                  detail_material, fastener_type]) ]),
      agent_issue(fabricator,fab_cost,physical_comp,physical,
            [structural_members(part,[member_type, member_property]),
                  connection_design(part,[bolt_properties, weld_properties,
                  detail_material, fastener_type])])).

relation(performance,
10    [agent_issue(designer,stiffness,struc_det,functional,
            [connection_design(method[bolt_properties, weld_properties,
                  detail_material, fastener_type]) ]),
      agent_issue(fabricator,fab_ease,fab_proc,physical,
            [construction_schedule(duration,[fabrication_schedule,
                  erection_schedule,stability_analysis]),
                  shop_operations(method,[cutting,drilling,welding,shop_assembly])])).
```

```
relation(performance,
11    [agent_issue(designer,stiffness,struc_det,functional,
          [connection_design(method[bolt_properties, weld_properties,
               detail_material, fastener_type]) ]),
      agent_issue(fabricator,fab_ease,physical_comp,physical,
          [structural_members(part,[member_type, member_property]),
               connection_design(part,[bolt_properties, weld_properties,
               detail_material, fastener_type])])).

relation(performance,
12    [agent_issue(designer,stiffness,physical_comp,physical,
          [structural_members(part,[member_type, member_property]),
               connection_design(part,[bolt_properties, weld_properties,
               detail_material, fastener_type])]),
      agent_issue(erector,safety,con_proc,physical,
          [construction_schedule(duration,[fabrication_schedule,
               erection_schedule,stability_analysis]),
               field_operations(method,[fastening_methods,temporary_bracing])])).

relation(performance,
13    [agent_issue(designer,stiffness,physical_comp,physical,
          [structural_members(part,[member_type, member_property]),
               connection_design(part,[bolt_properties, weld_properties,
               detail_material, fastener_type])]),
      agent_issue(erector,safety,mat_prop,physical,
          [strength_criteria(measure,[tensile,compressive]),
               serviceability(measure,[chemical_makeup,weldability,corrosion_
             properties,tolerances])])).

relation(performance,
14    [agent_issue(designer,stiffness,struc_det,functional
          [connection_design(method[bolt_properties, weld_properties,
               detail_material, fastener_type]) ]),
      agent_issue(erector,erec_ease,con_proc,physical
          [construction_schedule(duration,[fabrication_schedule,
               erection_schedule,stability_analysis]),
               field_operations(method,[fastening_methods,temporary_bracing])])).

relation(expense,
15    [agent_issue(designer,stiffness,physical_comp,physical,
          [structural_members(part,[member_type, member_property]),
               connection_design(part,[bolt_properties, weld_properties,
               detail_material, fastener_type])]),
      agent_issue(fabricator,mat_cost,physical_comp,physical,
          [structural_members(part,[member_type, member_property]),
               connection_design(part,[bolt_properties, weld_properties,
               detail_material, fastener_type])])).

relation(expense,
16    [agent_issue(designer,stiffness,physical_comp,physical,
          [structural_members(part,[member_type, member_property]),
```

```
        connection_design(part,[bolt_properties, weld_properties,
            detail_material, fastener_type])]),
    agent_issue(erector,erec_cost,con_proc,physical,
        [construction_schedule(duration,[fabrication_schedule,
            erection_schedule,stability_analysis]),
            field_operations(method,[fastening_methods,temporary_bracing])])).
```

Versatility Relationships:

```
relation(performance,
17  [agent_issue(designer,versatility,physical_comp,physical,
        [structural_members(part,[member_type, member_property]),
            connection_design(part,[bolt_properties, weld_properties,
            detail_material, fastener_type])]),
    agent_issue(fabricator,fab_cost,physical_comp,physical,
        [structural_members(part,[member_type, member_property]),
            connection_design(part,[bolt_properties, weld_properties,
            detail_material, fastener_type])])).

relation(performance,
18  [agent_issue(designer,versatility,physical_comp,physical,
        [structural_members(part,[member_type, member_property]),
            connection_design(part,[bolt_properties, weld_properties,
            detail_material, fastener_type])]),
    agent_issue(fabricator,fab_ease,fab_proc,physical,
        [construction_schedule(duration,[fabrication_schedule,
            erection_schedule,stability_analysis]),
            shop_operations(method,[cutting,drilling,welding,shop_assembly])])).

relation(performance,
19  [agent_issue(designer,versatility,physical_comp,physical,
        [structural_members(part,[member_type, member_property]),
            connection_design(part,[bolt_properties, weld_properties,
            detail_material, fastener_type])]),
    agent_issue(erector,erec_cost,con_proc,physical
        [construction_schedule(duration,[fabrication_schedule,
            erection_schedule,stability_analysis]),
            field_operations(method,[fastening_methods,temporary_bracing])])).

relation(performance,
20  [agent_issue(designer,versatility,physical_comp,physical,
        [structural_members(part,[member_type, member_property]),
            connection_design(part,[bolt_properties, weld_properties,
            detail_material, fastener_type])]),
    agent_issue(erector,erec_ease,con_proc,physical,
        [construction_schedule(duration,[fabrication_schedule,
            erection_schedule,stability_analysis]),
            field_operations(method,[fastening_methods,temporary_bracing])])).
```

Reliability Relationships:

```
relation(performance,
21    [agent_issue(designer,reliability,mat_prop,physical,
            [strength_criteria(measure,[tensile,compressive]),
                  serviceability(measure,[chemical_makeup,weldability,corrosion_
                properties,tolerances])]),
       agent_issue(fabricator,fab_cost,fab_proc,physical,
            [construction_schedule(duration,[fabrication_schedule,
                  erection_schedule,stability_analysis]),
                  shop_operations(method,[cutting,drilling,welding,shop_assembly])])]).

relation(performance,
22    [agent_issue(designer,reliability,mat_prop,physical,
            [strength_criteria(measure,[tensile,compressive]),
                  serviceability(measure,[chemical_makeup,weldability,corrosion_
                properties,tolerances])]),
       agent_issue(fabricator,fab_ease,fab_proc,physical,
            [construction_schedule(duration,[fabrication_schedule,
                  erection_schedule,stability_analysis]),
                  shop_operations(method,[cutting,drilling,welding,shop_assembly])])]).

relation(performance,
23    [agent_issue(designer,reliability,mat_prop,physical,
            [strength_criteria(measure,[tensile,compressive]),
                  serviceability(measure,[chemical_makeup,weldability,corrosion_
                properties,tolerances])]),
       agent_issue(erector,erec_cost,con_proc,physical,
            [construction_schedule(duration,[fabrication_schedule,
                  erection_schedule,stability_analysis]),
                  field_operations(method,[fastening_methods,temporary_bracing])])]).

relation(performance,
24    [agent_issue(designer,reliability,mat_prop,physical,
            [strength_criteria(measure,[tensile,compressive]),
                  serviceability(measure,[chemical_makeup,weldability,corrosion_
                properties,tolerances])]),
       agent_issue(erector,erec_ease,con_proc,physical,
            [construction_schedule(duration,[fabrication_schedule,
                  erection_schedule,stability_analysis]),
                  field_operations(method,[fastening_methods,temporary_bracing])])]).
```

Fabrication Cost Relationships:

```
relation(expense,
25    [agent_issue(fabricator,fab_cost,physical_comp,physical,
            [structural_members(part,[member_type, member_property]),
                  connection_design(part,[bolt_properties, weld_properties,
                   detail_material, fastener_type])]),
```

```
          agent_issue(erector,safety,mat_prop,physical,
                [strength_criteria(measure,[tensile,compressive]),
                      serviceability(measure,[chemical_makeup,weldability,corrosion_
                    properties,tolerances])])).

relation(expense,
26        [agent_issue(fabricator,fab_cost,physical_comp,physical,
                [structural_members(part,[member_type, member_property]),
                      connection_design(part,[bolt_properties, weld_properties,
                      detail_material, fastener_type])]),
          agent_issue(erector,erec_cost,con_proc,physical,
                [construction_schedule(duration,[fabrication_schedule,
                      erection_schedule,stability_analysis]),
                      field_operations(method,[fastening_methods,temporary_bracing])])]).

relation(expense,
27        [agent_issue(fabricator,fab_cost,fab_proc,physical,
                [construction_schedule(duration,[fabrication_schedule,
                      erection_schedule,stability_analysis]),
                      shop_operations(method,[cutting,drilling,welding,shop_assembly])]),
          agent_issue(erector,erec_cost,con_proc,physical,
                [construction_schedule(duration,[fabrication_schedule,
                      erection_schedule,stability_analysis]),
                      field_operations(method,[fastening_methods,temporary_bracing])])]).

relation(expense,
28        [agent_issue(fabricator,fab_cost,physical_comp,physical,
                [structural_members(part,[member_type, member_property]),
                      connection_design(part,[bolt_properties, weld_properties,
                      detail_material, fastener_type])]),
          agent_issue(erector,erec_ease,con_proc,physical,
                [construction_schedule(duration,[fabrication_schedule,
                      erection_schedule,stability_analysis]),
                      field_operations(method,[fastening_methods,temporary_bracing])])]).

relation(expense,
29        [agent_issue(fabricator,fab_cost,fab_proc,physical,
                [construction_schedule(duration,[fabrication_schedule,
                      erection_schedule,stability_analysis]),
                      shop_operations(method,[cutting,drilling,welding,shop_assembly])]),
          agent_issue(erector,erec_ease,con_proc,physical,
                [construction_schedule(duration,[fabrication_schedule,
                      erection_schedule,stability_analysis]),
                      field_operations(method,[fastening_methods,temporary_bracing])])]).
```

Fabrication Ease Relationships:

```
relation(expense,
30    [agent_issue(fabricator,fab_ease,physical_comp,physical,
            [structural_members(part,[member_type, member_property]),
                connection_design(part,[bolt_properties, weld_properties,
                detail_material, fastener_type])]),
       agent_issue(erector,safety,mat_prop,physical,
            [strength_criteria(measure,[tensile,compressive]),
                serviceability(measure,[chemical_makeup,weldability,corrosion_
               properties,tolerances])])).

relation(expense,
31    [agent_issue(fabricator,fab_ease,fab_proc,physical,
            [construction_schedule(duration,[fabrication_schedule,
                erection_schedule,stability_analysis]),
                shop_operations(method,[cutting,drilling,welding,shop_assembly])]),
       agent_issue(erector,safety,mat_prop,physical,
            [construction_schedule(duration,[fabrication_schedule,
                erection_schedule,stability_analysis]),
                shop_operations(method,[cutting,drilling,welding,shop_assembly])])).

relation(expense,
32    [agent_issue(fabricator,fab_ease,physical_comp,physical,
            [structural_members(part,[member_type, member_property]),
                connection_design(part,[bolt_properties, weld_properties,
                detail_material, fastener_type])]),
       agent_issue(erector,safety,con_proc,physical,
            [construction_schedule(duration,[fabrication_schedule,
                erection_schedule,stability_analysis]),
                field_operations(method,[fastening_methods,temporary_bracing])])).

relation(expense,
33    [agent_issue(fabricator,fab_ease,fab_proc,physical,
            [construction_schedule(duration,[fabrication_schedule,
                erection_schedule,stability_analysis]),
                shop_operations(method,[cutting,drilling,welding,shop_assembly])]),
       agent_issue(erector,safety,con_proc,physical,
            [construction_schedule(duration,[fabrication_schedule,
                erection_schedule,stability_analysis]),
                field_operations(method,[fastening_methods,temporary_bracing])])).

relation(expense,
34    [agent_issue(fabricator,fab_ease,fab_proc,physical,
            [construction_schedule(duration,[fabrication_schedule,
                erection_schedule,stability_analysis]),
                shop_operations(method,[cutting,drilling,welding,shop_assembly])]),
       agent_issue(erector,erec_cost,con_proc,physical,
            [construction_schedule(duration,[fabrication_schedule,
                erection_schedule,stability_analysis]),
                field_operations(method,[fastening_methods,temporary_bracing])])).
```

```
relation(expense,
35    [agent_issue(fabricator,fab_ease,fab_proc,physical,
          [construction_schedule(duration,[fabrication_schedule,
              erection_schedule,stability_analysis]),
              shop_operations(method,[cutting,drilling,welding,shop_assembly])]),
       agent_issue(erector,erec_ease,con_proc,physical,
          [construction_schedule(duration,[fabrication_schedule,
              erection_schedule,stability_analysis]),
              field_operations(method,[fastening_methods,temporary_bracing])])]).
```

Material Cost Relationships:

```
relation(expense,
36    [agent_issue(fabricator,mat_cost,physical_comp,physical,
          [structural_members(part,[member_type, member_property]),
              connection_design(part,[bolt_properties, weld_properties,
              detail_material, fastener_type])]),
       agent_issue(erector,erec_cost,physical_comp,physical,
          [structural_members(part,[member_type, member_property]),
              connection_design(part,[bolt_properties, weld_properties,
              detail_material, fastener_type])])]).

relation(expense,
37    [agent_issue(fabricator,mat_cost,physical_comp,physical,
          [structural_members(part,[member_type, member_property]),
              connection_design(part,[bolt_properties, weld_properties,
              detail_material, fastener_type])]),
       agent_issue(erector,erec_ease,physical_comp,physical,
          [structural_members(part,[member_type, member_property]),
              connection_design(part,[bolt_properties, weld_properties,
              detail_material, fastener_type])])]).

relation(expense,
38    [agent_issue(fabricator,mat_cost,physical_comp,physical,
          [structural_members(part,[member_type, member_property]),
              connection_design(part,[bolt_properties, weld_properties,
              detail_material, fastener_type])]),
       agent_issue(erector,safety,con_proc,physical,
          [construction_schedule(duration,[fabrication_schedule,
              erection_schedule,stability_analysis]),
              field_operations(method,[fastening_methods,temporary_bracing])])]).
```

D. Revised DFI Connection Database

```
(Endplate
  (orientation                flange)
  (type                       1)
  (moment_connection          FB/SW)
  (shear_connection           FB/SW)
  (category                   FB/SW/FB/SW)
  (detail_material            endplate)
)

(Top_and_Bottom_Plates_with_Shear_Plate
  (orientation                web)
  (type                       1)
  (moment_connection          SW/FB)
  (shear_connection           SW/FB)
  (category                   SW/FB/SW/FB)
  (detail_material            flange_plate _ w_plate)
)

(Top_and_Bottom_Plates_with_Shear_Tee
  (orientation                flange)
  (type                       1)
  (moment_connection          SW/FB)
  (shear_connection           SW/FB)
  (category                   SW/FB/SW/FB)
  (detail_material            flange_plate_w_tee)
)

(Top_and_Bottom_Plates_with_Shear_Angle
  (orientation                flange)
  (type                       1)
  (moment_connection          SW/FB)
  (shear_connection           SW/FB)
  (category                   SW/FB/SW/FB)
  (detail_material            flange_plate_w_angle)
)

(Top_and_Bottom_Plates_with_Shear_Plate
  (orientation                flange)
  (type                       1)
  (moment_connection          SW/FB)
  (shear_connection           SW/FB)
  (category                   SW/FB/SW/FB)
  (detail_material            flange_plate_w_plate)
)
```

```
(Direct_Flange_Weld_with_Shear_Plate
  (orientation                web)
  (type                       1)
  (moment_connection          SW/FW
  (shear_connection           SW/FB)
  (category                   SW/FW/SW/FB)
  (detail_material            flange_weld_w_plate)
)

(Direct_Flange_Weld_with_Shear_Angle
  (orientation                flange)
  (type                       1)
  (moment_connection          FW/FW)
  (shear_connection           SW/FB)
  (category                   FW/FW/SW/FB)
  (detail_material            flange_weld_w_angle)
)

(Direct_Flange_Weld_with_Web_Weld
  (orientation                flange)
  (type                       1)
  (moment_connection          FW/FW)
  (shear_connection           FW/FW)
  (category                   FW/FW/FW/FW)
  (detail_material            flange_weld_w_weld)
)

(Direct_Flange_Weld_with_Shear_Tee
  (orientation                flange)
  (type                       1)
  (moment_connection          FW/FW)
  (shear_connection           SW/FB)
  (category                   FW/FW/SW/FB)
  (detail_material            flange_weld_w_tee)
)

(Direct_Flange_Weld_with_Shear_Plate
  (orientation                flange)
  (type                       1)
  (moment_connection          FW/FW)
  (shear_connection           SW/FB)
  (category                   FW/FW/SW/FB)
  (detail_material            flange_weld_w_plate)
)
```

```
(Top_and_Bottom_Tees_with_Shear_Tee
  (orientation              flange)
  (type                     1)
  (moment_connection        SW/FB)
  (shear_connection         SW/FB)
  (category                 SW/FB/SW/FB)
  (detail_material          tee_w_tee)
)

(Top_and_Bottom_Tees_with_Shear_Plate
  (orientation              flange)
  (type                     1)
  (moment_connection        SW/FB)
  (shear_connection         SW/FB)
  (category                 SW/FB/SW/FB)
  (detail_material          tee_w_ plate)
)

(Top_and_Bottom_Tees_with_Shear_Angle
  (orientation              flange)
  (type                     1)
  (moment_connection        SW/FB)
  (shear_connection         SW/FB)
  (category                 SW/FB/SW/FB)
  (detail_material          tee_w_angle)
)
```

Vita

Stephanie J. Wagaman was born in Bethlehem, PA on September 11, 1956. She received her Bachelor of Science in Computer Science from Cedar Crest College in May 1989. She then continued on to graduate school and received her Master of Science degree in Computer Science from Lehigh University in June 1991.

While at Cedar Crest, Stephanie was inducted into Delphi, the honorary Scholastic Society; Alpha Sigma Lambda, a National Honor Society for Students in Continuing Education; and was awarded the Cedar Crest Board of Associates Scholarship. She graduated from Cedar Crest College summa cum laude and was awarded the Paul V. Kunkel Math Award. At Lehigh, Stephanie served as an ATLSS Center Scholar doing research in the development of agent models for the Designer Fabricator Interpreter knowledge system. She served as Chair of LU-SIGART, the Lehigh University Special Interest Group in Artificial Intelligence, and was awarded the Executive Award by Lehigh's ACM Student Chapter.

Stephanie is a member of the Association for Computing Machinery (ACM), ACM-SIGART, and the American Association for Artificial Intelligence (AAAI).