A Knowledge Based Expert System for Analyzing Welded Structures

by
Regina A. Middaugh

B.S. Materials Science and Engineering
Massachusetts Institute of Technology
(1993)

Submitted to the Department of Ocean Engineering
in Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE
in Ocean Engineering

at the Massachusetts Institute of Technology
February, 1995

Author Signature_
Department of Ocean Engineering
February, 1995

Certified
Koichi Masubuchi
Kawasaki Professor of Materials Science and Ocean Engineering
Thesis Supervisor

Accepted by_____
Professor A. Douglas Carmichael
Department Graduate Committee
Department of Ocean Engineering

A Knowledge Based Expert System for Analyzing Welded Structures

by

Regina A. Middaugh

Submitted to the Department of Ocean Engineering on
January 20, 1995, in partial fulfillment of the requirements
for the degree of Master of Science in Ocean Engineering

ABSTRACT

In this study, I explored the merits of a Knowledge Based Expert System (KBES) as both an organizational and analytical tool for investigating ship grounding events and grounded ship structures. This involved exploring expert system capabilities and ship grounding phenomena to justify the development of such a system. This exercise showed that the complex and inexact nature of ship grounding events lends itself to KBES representation. Using a commercial logic shell (Neuron Data's Nexpert Object 3.0) for KBES development, I designed the Global KBES for Ship Grounding, an expert system that addresses a broad spectrum of ship grounding issues that interest the shipping community. To demonstrate the application of the design, I created the Fillet Weld KBES, a working expert system for analyzing fillet welds exposed to grounding conditions. The Fillet Weld KBES also stands on its own as the "Fillet Weld Expert" with regard to ship grounding events. It combines weld-related theories and data from the Joint MIT-Industry Project for Tanker Safety.

This document includes an in-depth description of common expert system features, and the tools and strategies required for developing such a system. Several existing knowledge based systems demonstrate successful KBES application in technical fields, and a wave of recent KBES proposals suggest an increasing acceptance of these systems. Although the application of knowledge based expert systems is still relatively new, they offer distinct advantages over traditional computer systems and trend towards wide-scale future use.

ACKNOWLEDGMENT

I would like to thank Professor Masubuchi for his role as both advisor and thesis supervisor. His enthusiasm for new ideas has made this KBES project possible. I would also like to thank Professor Francisco Fernandez-Gonzalez for freely sharing his extensive shipping knowledge. His practical experience and ties to the international shipping community have advanced this project from an academic exercise to a practical and potentially useful tool.

Also, thanks to Professor Thomasz Wierzbicki, Professor Frank McClintock, and Chad Brooks for their feedback on expert systems in general. Special thanks to Li Liang and Mike Binnard for their hours fussing with the system's front end and their support. All members of the MIT Joint-Industry Tanker Safety Project should also be recognized, as they contributed either directly or indirectly to this thesis; my KBES houses their theories and data. I hope my design and implementation effectively captures the quality work generated within the Joint MIT-Industry Program for Tanker Safety.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1. Introduction

A. Joint MIT-Industry Program for Tanker Safety

Increased public anxiety over environmental damage, coupled with a string of substantial oil spills from recent tanker groundings, prompted creation of the *Joint MIT-Industry Program for Tanker Safety* in 1992. The Tanker Safety Project joined MIT's Ocean Engineering Department and the international shipping community in a three year study to establish methods and tools for improving ship hull survivability during grounding incidents. This project is one of several current attempts to rigorously characterize ship grounding events. Several early Tanker Safety reports defend the need for improved grounding analysis and describe the tasks for the three year project **[Wierzbicki, Rady, and Peer 1990] [Joint Project Addendum 1992].**

Because of the Exxon Valdez disaster, the U.S. Coast Guard enacted the OPA 1990 ruling requiring double hulls for all oil tankers operating in U.S. waters by 2015. Currently, ships are not designed to resist grounding loads, and few analytical tools exist for addressing grounding events and predicting hull damage **[Wierzbicki, Rady, and Peer 1990].** Although many people recognize the fact that better ship structures could result in less oil spillage, the shipping community disagrees as to which design changes optimize hull strength, manufacturing cost, and shipping operation. The Tanker Safety Project addresses grounding issues at a crucial time in response to these concerns. The project promotes the development of grounding-resistant ships by providing analytical methods and design tools for characterizing moderate to severe grounding events **[Wierzbicki, Rady, and Peer 1990].** Primary tasks for the Tanker Safety project include:

- Identifying relevant grounding parameters, hull components, and failure modes.
- Developing closed form or semi-empirical solutions to describe the response of hull components exposed to grounding loads.
- Producing a computer aided design (CAD) program to help designers analyze existing ship designs, predict grounding damage, and design grounding-resistant ship hulls.

- Writing a grounding handbook that guides ship designers and engineers through damage calculations for individual hull components.

[Joint Project Addendum 1992]

Tanker Safety reports thoroughly present the above tasks, the research procedures used to accomplish these tasks, and the most recent results. These reports are kept by Professor Thomasz Wierzbicki (Principal Investigator for the Joint MIT-Industry Program for Tanker Safety) in MIT's Ocean Engineering Department.

B. KBES for Ship Grounding Analysis

For ship design, manufacturing, and operation, the shipping community historically relies on qualitative information **[Gonzalez 1994] [Masubuchi 1994]**. The qualitative information commonly used in this field typically includes judgment experience, expert knowledge, and description. At present, research studies and engineering methods supplement this information with quantitative and analytical approaches to shipping issues. For example, studies for the Tanker Safety Project address grounding events using closed form analytical models (semi-empirical expressions), research data, and computer modeling techniques (Finite Element Analysis, or **FEM**). For grounding analysis and damage prediction, a fully representative, quantitative analysis is ideal; and the Tanker Safety Project works towards this end. New methods such as these require time to development, test, and verify. Because complete grounding models do not yet exist, near future shipping practices will likely rely heavily on traditional practices, while cautiously incorporating new methods after they have proven themselves in practice **[Masubuchi 1994] [Gonzalez 1994]**.

Within the Tanker Safety Project, I designed a Knowledge Based Expert System (**KBES**) that consolidates MIT grounding research and characterizes grounding events. The KBES is a user-friendly, "intelligent" database that incorporates multiple forms of information, both qualitative and quantitative. It can combine existing grounding knowledge with research data developing at MIT, and present it in a format that directly addresses questions from the shipping community. This ability makes it particularly attractive for ship grounding problems.

The KBES can also be flexible. It will deliver suggestions and analysis using various types of input. Information can be formatted to suit the user's needs and technical background. In this way, the expert system complements the Tanker Project's CAD

program and handbook. While the CAD program and handbook cater to ship designers, engineers, and related technical experts, the KBES provides grounding information to non-experts.

Knowledge Based Expert Systems developed in the field of Artificial Intelligence. They are becoming increasingly popular for industrial applications as knowledge representation and search methods continue to improve. Commercial packages for KBES construction are now available for most computer platforms at increasingly reasonable prices **[Price 1990]**. I will discuss key features of current expert systems that relate specifically to ship grounding characterization.

To demonstrate my global KBES design, I developed an example KBES that delivers fillet weld information related to ship grounding loads. Because joint failure is a critical concern in grounding situations, this system focuses on weld design analysis and weld failure prevention. All information contained in this example system comes directly from work conducted within the Tanker Safety Project.

C. Thesis Organization

This paper is divided into three parts:

- **Part I** -- Provides background information about Knowledge Based Expert Systems. Chapter 2 describes basic expert system components and current technical applications. Chapter 3 addresses artificial intelligence concepts and tools required to create, operate, and maintain an expert system.

- **Part II** -- Justifies the development of a ship grounding KBES to embody information generated by the Tanker Safety Project (Chapter 4); and presents my design of the Global KBES for Ship Grounding (Chapters 5 and 6). I provide the foundation, organization, and data required for the construction of this design using the concepts and tools discussed in Part I.

- **Part III** -- Presents my Fillet Weld KBES, a working expert system (Chapter 7). This system demonstrates the implementation and operation of the Global KBES design presented in Part II (Chapter 8). The Fillet Weld KBES is also a unique combination of cutting edge research on fillet welds (T-joints) exposed to grounding loads. Information for this system comes directly from the Tanker Safety Project.

Part I

Background Information

Part I provides background information about Knowledge Based Expert Systems. This information defines the terms and concepts used in Parts II and III.

Chapter 2 describes basic KBES components, guidelines for using knowledge based systems, advantages of KBES development, and current technical applications. Chapter 3 covers expert system tools and concepts, commercial KBES development packages, and design methodologies. To clarify these concepts, I describe features of Nexpert Object 3.0, the commercially available software package used in Part III for KBES development **[Neuron Data 1987]**.

Chapter 2. KBES Basics

This chapter describes the basic components, common functions, and current applications of Knowledge Based Expert Systems. It also provides general information that supports the materials presented in later chapters.

Although no two expert systems are identical, the following information presents attributes common to most expert systems. This information is particularly useful for understanding how these systems manipulate data, for designing KBES's, and for choosing between commercial KBES packages and customized systems.

A. Definitions and General Description

A Knowledge Based Expert System (**KBES**) is an interactive computer program that combines expert knowledge with an iterative search method to address complex issues **[Neuron Data 1987]**. The expert knowledge is any heuristic information used by people who are knowledgeable in the appropriate field. This information includes rules of thumb, quick calculations, estimations, solution procedures, and site-specific practices **[Waterman 1986] [Siddall 1990].** For ship grounding analysis, experts would include individuals familiar with ships, shipping practices, or grounding events. These people might include: ship designers, builders, and operators; international and national regulators; damage investigators; and ship repairers.

The KBES formats expert knowledge as simple rules and stores the rules in a Knowledge Base. Databases, which are common computer applications, operate in a similar manner; they store information and return it with a defined search method. The primary difference between Knowledge Bases and databases is that databases stores data in a table format, while a KBES stores knowledge in a rule format **[Waterman 1986]**. Chapter 3 thoroughly addresses these distinctions, and explains why a KBES is preferable to a database for collecting ship grounding information.

Knowledge bases store information as simple rules. Rules typically take an **If/Then** form:

If Conditions A, B, and C exist, *Then* Conclusion D is true.

Most forms of knowledge fit easily into this format **[Agapakis and Masubuchi 1988]**. There are a number of search methods, specific to rule based representation, used to pull rules from the Knowledge Base according to analytical needs. These techniques are iterative search methods, and they attempt to mimic human reasoning processes with standard problem solving methods. Good search methods sift through the Knowledge Base efficiently, extracting and evaluating rules which are pertinent to the immediate task. Common search methods for current expert systems include: backward chaining, forward chaining, and combinations of both backward and forward chaining **[Maher 1987]**. Search methods usually develop within the fields of Computer Science or Artificial Intelligence, and the most applicable methods make their way into commercial packages **[Siddall 1990]**. Chapter 3 discusses KBES search methods more thoroughly.

Components common to most expert systems include: a User Interface, a Knowledge Base, and an Inference Engine (**Figure 2.1**) **[Siddall 1990]**. Optional features, such as database links, spread sheet applications, or custom software options, enhance the basic KBES system. Chapter 3 further describes the optional KBES components relevant to ship grounding problems.

Basic KBES Components

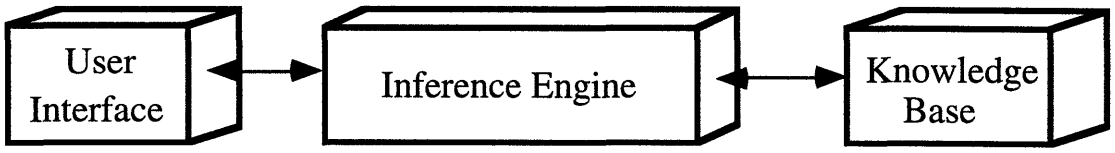

Figure 2.1. Standard components for Knowledge Based Expert Systems include the User Interface, Inference Engine, and Knowledge Base. The User Interface accepts information from the User. The Inference Engine controls procedures and relays information between User Interface and Knowledge Base. The Knowledge Base holds information and methods for solution processes.

A-1 User Interface

The User Interface displays computer screens that allow the user to interact with the expert system. The user participates in inquiries and problem solving processes by:

- selecting the desired problem or analysis type,
- establishing initial values and conditions,
- altering conditions and providing additional information during the solution process, and
- requesting solution explanations and inquiring about reasoning pathways.

The first few computer screens provided by the User Interface identify the problem or analysis and establish initial conditions. This process narrows the search domain to related tasks only. The following screens request additional information as new search paths develop. Depending on the expected questions and available information, the KBES designer can construct a User Interface that allows different levels of interaction between the user and the expert system. For example, for a given analysis, the user can provide specific information or generalized estimates. The final screens of the User Interface give solutions, explain how the system reached a conclusion, and inform the user if external sources were used.

A-2 Knowledge Base

The Knowledge Base contains a collection of rules in the form of hypotheses and supporting conditions. Rules evaluate the user's input to establish whether or not a hypothesis is true or false. Both the hypothesis and its related conditions originate from expert knowledge, common practices, generalizations, rough calculations, and educated guesses **[Waterman 1986]**. Typical knowledge representation involves rules with a standard **If/Then** (or When/Do) format. **Figure 2.2** shows this format.

Common KBES Rule Format

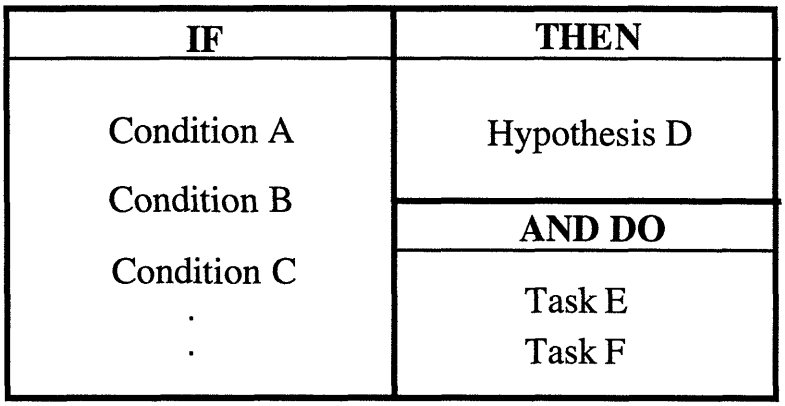

Figure 2.2. The simple rule format used by Nexpert Object, a commercial software package for KBES development **[Neuron Data 1987].** The rule contains a hypothesis, conditions that make the hypothesis true, and related tasks for the rule to perform if invoked.

The rule's left side (**Figure 2.2**) is a list of conditions related to the hypothesis, located on the rule's right side. Satisfying all of these conditions proves that the hypothesis is true and invokes the tasks listed in the lower right under the AndDo statement. The newest Nexpert Object version expanded the rule format to include: If/Then ... AndDo/ElseDo, which invokes alternative actions for false hypotheses **[Neuron Data 1994]**. Rule representation works well for most knowledge types, but rules are inefficient for some purposes. Alternative formats exist for these cases, and designers often combine these formats with rules for a broader knowledge representation. Chapter 3 discusses several rule alternatives.

A-3 Inference Engine

The Inference Engine, shown in **Figure 2.1** controls the problem solving process, or information search through the system's rules. It relays information between the User Interface and the Knowledge Base, selecting pertinent rules and attempting to verify their hypotheses by matching input values to rule conditions. The Inference Engine uses an iterative search pattern. If an initial search produces a partially satisfied rule, the second search requests more information from the user. When the user cannot provide the missing information, the Inference Engine may search other sources, such as external databases, tables, or other software applications. Different search methods are appropriate for

different types of knowledge structures; search pattern combinations have become increasingly popular. Chapter 3 uses the software application Nexpert Object to demonstrate the most popular search methods. **[Waterman 1986]**

The interactive and iterative processes used by expert systems mimic standard engineering problem solving methods. Like an engineer, expert systems are flexible because they work with incomplete data, estimations, and descriptive information. They fill in information gaps with their own knowledge. The engineer's or KBES's solutions would vary according to the available information and their own background experiences. For example, a KBES analysis could result in a highly probable conclusion if provided with complete and specific input; or, it could produce several less definite recommendations using less specific input. In this respect, the knowledge based system is compatible with engineering processes where unknowns and educated guesses are unavoidable **[Bedard and Gowri 1990]**.

B. Appropriate Use of the KBES

Because knowledge based systems are relatively new, especially in technical fields, formal design procedures and methodologies have not yet been formalized **[Edwards 1991]**. KBES designers suggest using expert systems to address problems when:

- No formal procedures govern a practice.
- Knowledge and experience governs a practice.
- Human experts in that field are scarce or unavailable.
- Information is widely dispersed or informally stored.

[Siddall 1990]

There is also the issue of choosing between a knowledge based system and a standard database. The former offers flexibility, while the latter is more common **[Waterman 1986]**.

B-1 When to Use the KBES

Knowledge based expert systems are most appropriate in fields without formalized procedures. This situation is common in traditional professions such as shipping, railroad construction, and architecture **[Arockiasamy 1993]**. Many of the guiding principles for these industries come from years of experience.

For example, architects have few strict laws or procedures for designing buildings. The architect draws ideas from a combination of techniques learned in school, design examples, past experience, and personal preferences. There is considerable freedom within these guidelines. The designer can put anything on paper, but problems arise when these designs transfer from paper to practice. The contractor who constructs the building must respect equipment capabilities, material availability and performance, construction time, safety regulations, and construction costs. The designer and contractor must compromise, and this process may require a number of design alterations. After years of this routine, experience tells the designer that certain structural arrangements work better than others, that contractors refuse to build certain features, and that building codes ban several structures. An expert system that incorporates this experience, as a knowledge base of successful structures and a list of basic construction limitations, could effectively reduce the revision time spent between less experienced designers and contractors **[Bedard and Gowri 1990] [Ghosh and Kalyanaraman 1993]**.

Expert systems prevail when human experts are unavailable or do not exist **[Agapakis and Masubuchi 1988]**. This occurs in remote work locations like ocean and space environments. It also occurs when large distances or unrelated professional fields separate or conceal experts from those needing expert knowledge. In both situations, information from the expert is either limited or unavailable. In space environments, for example, there are no experts on space welding procedures **[Nakamura 1994]**. There are, however, many welding experts, space experts, and supporting literature in both areas. A second problem is that the information, accessible on earth, may not be easily accessible from space, unless the astronauts are the experts. Taka Nakamura has created a KBES that combines information from both fields, and suggests welding procedures applicable to space conditions **[Nakamura 1994]**. It serves as the easily accessible space welding expert.

Compressing expert knowledge into a computer program saves time and money, especially when experts must be consulted frequently. Of course, the KBES cannot completely replace human experts because computers cannot provide equivalent intuition,

creativity, or the background experience required to answer some questions **[Harvard College 1986]**. These systems can deliver portions of the human expertise on demand and in most locations. They can also collect knowledge from multiple experts, forming unique knowledge combinations to create "computer experts" when no human counterpart exists.

B-2 Knowledge Base vs Database

A knowledge base is similar to a database; they both store information and retrieve parts of it on demand. Knowledge bases and databases, however, are not interchangeable. Each has a different method for storing and retrieving information. For example, a database typically stores numerical data and text in a table format. It retrieves data by searching its tables until it finds the column and row that matches the user's input criteria. A knowledge base stores knowledge in a rule format and retrieves it using heuristic (inferential) procedures. For most circumstances, one format will be more appropriate than another **[Waterman 1986]**.

System flexibility distinguishes knowledge based systems from databases. Compared to a KBES, databases have fairly stringent input requirements. To pull a particular piece of information from a table cell, database input must be exact. The search process involves matching user input to table headings, columns, and rows. If search criteria are too general, incomplete, or incorrectly entered, the solution will result in too many answers, no answer, or an incorrect answer, respectively. For example, to find the title of a book in a library's computer card catalog, the user must know some specific information about the book, like title, author name, or subject. If the user gives a broad subject or indicates a popular author's name, he or she may receive many titles. The user must then decide which title is appropriate. If the user misspells the author's name, the result may be no titles or titles by a different author. To extract the desired information, the user must know exactly what they are looking for, have an idea of how to find it, and then recognize a correct solution. If the user has limited information, the database's inflexibility may be frustrating.

A knowledge base usually stores less data than a database in its immediate working environment. Instead of storing data, the KBES stores knowledge and methods as rules for manipulating data. Rules store complete bits of knowledge and provide pathways that link these bits to form a final solution. The system accommodates less specific user input, and uses inferences to help locate related solutions. In contrast to a database, uncertainties

and probabilities are permissible. **Table 2.1** summarizes the critical differences between data and knowledge bases **[Waterman 1986]**.

Table 2.1. Differences Between Knowledge Bases and Databases

Data Processing	Knowledge Engineering
stores numbers or text in multiple tables	stores symbolic information and methods in rules
algorithmic search	heuristic search
difficult to modify structure	easy to modify structure
repetitive process	inferential process
inflexible input and output format	user friendly and lenient with input and output formats

from **[Waterman 1986] [Edwards 1991]**

C. KBES Development

Besides the benefits associated with *having* a knowledge based system, there are also advantages accrued from *developing* such systems. These benefits include:

- **Locating Research Gaps** -- The KBES *design process*, even in its early stages, locates important information and procedural gaps. KBES search paths resemble logic trees, where a path to the final solution may include a number of intermittent subroutines. Design involves linking user questions to relevant analytical tools, specific tasks, and supporting data. Missing components along the solution chain make a final solution difficult, if not impossible, to reach. Discovering these gaps guides future research.

- **Combining and Verifying Data** -- The expert system can accept multiple forms of information, such as numeric and non-numeric data, equations, procedures, graphs, and pictures **[Neuron Data 1994]**. This property is particularly useful for problems with multiple solution approaches **[Peers, Tang, et al. 1994]**. It is also useful for comparing and contrasting results from different studies.

- **Creating a Computer Expert** -- A KBES serves as a "computer expert" for a particular area of interest. It will not have the flexibility and common sense possessed by human experts, but it will provide some expert knowledge for a specified range of topics **[Bedard and Gowri 1990]**. Access to this knowledge is convenient when human experts are unavailable or too expensive **[Agapakis and Masubuchi 1988]**. If the KBES answers the user's questions, it saves the human expert's efforts for more difficult problems **[Harvard College 1986]**.

- **Formatting and Saving Knowledge** -- Knowledge based systems format knowledge and technical information **[Agapakis and Masubuchi 1988]**. Knowledge, or "rules of thumb" practiced by experts, come from experience. Experts use this knowledge, but seldom format it as heuristic rules. When an expert leaves a company, much of their knowledge could easily disappear. A knowledge based system provides a framework for formalizing this experience **[Harvard College 1986]**.

D. Current and Future KBES Applications

Knowledge based systems are currently used in many technical fields. **Table 2.2** lists several current KBES applications and the services they provide. These include systems for both expert and non-expert use. For example, the Welding KBES helps non-expert welders determine appropriate materials and welding techniques for a given joint configuration **[Fukuda 1987] [Agapakis and Masubuchi 1988]**. The Medical KBES, however, helps a medical expert (doctor) sift through numerous ailments with common symptoms **[Waterman 1986]**. In the latter system, the doctor must be an expert to interpret the conclusion and decide which diagnosis is most likely.

Expert systems have different development goals depending on who develops it and for what purpose. Many existing systems serve only as research tools where the primary emphasis may be developing knowledge representations and search methods, as opposed to evaluating actual data. KBES's for company use often address very specific issues. For these systems, primary emphasis lies on formalizing data, company procedures, available expert knowledge, and current problems. They care less about the actual developmental tools, as long as they work. Because the KBES is for in-house use, developers may operate the system with programming bugs still present. Commercial KBES's reach the highest level of completeness. To attract and keep customers, the designers take great pains to provide user friendly interfaces, easy editing capabilities, and very few system bugs **[Waterman 1986]**.

Table 2.2. Examples of Current KBES Applications

Field of Use	Reference	KBES Tasks
Offshore Engineering	**[Bertini 1994] [Peers, Tang, et al. 1994] [Reddy, Arock-iasamy, et al 1988]**	site planning, design, cost optimization, material selection, risk assessment, failure analysis, maintenance control
Weld and Soldering Technology	**[Agapakis and Masubuchi 1988] [Fukuda 1987] [Harvard College 1986] [Nakamura 1994]**	procedure selection, material selection, fault analysis, process evaluation and control
Architecture and Structural Engineering	**[Bedard and Gowri 1990] [Cavanaugh and Billatos 1992] [Ghosh and Kaly-anaraman 1993] [Malaureille 1987]**	design assistance, strength analysis, material selection, regulation compliance, cost analysis, site planning and regulation
Medicine	**[Waterman 1986]**	symptom diagnosis, disease
Biology	**[Waterman 1986]**	DNA interpretation, identification
Electrical Eng.	**[Waterman 1986]**	program and circuit debugging
Shipping Industry	**[Poudret, Huther, et al. 1981] [Sen and Gerigk 1994]**	design assistance, manufacturing process control, life cycle analysis, ship route planning, database management

As of 1991, there were very few operational KBES's in use **[Edwards 1991]**. My literature survey indicates that few existing expert systems are at commercial levels or full implementation stages because KBES's are still relatively new. Early development tools and design procedures made designing, formatting, debugging, and maintaining a complete system time intensive. Also, the early systems were simple and inexact, making them inappropriate for technical problems. As commercial programs reduced development time and improved system accuracy, user confidence gradually increased. KBES's are rapidly becoming acceptable tools for analyzing complex technical problems **[Bedard and Gowri 1990] [Agapakis and Masubuchi 1988]**. There are currently many proposals for creating new knowledge based systems in technical fields. Soon, more systems will reach the application stages of development.

Commercial software packages exist for KBES development. Many of these originated from the first successful knowledge based systems, like Dendral (1970) and Mycin (1970) **[Adeli 1988]**. Software companies stripped early KBES's of their data and sold the remaining logic shells to the public **[Adeli 1988]**. These logic shells provide a framework and the necessary tools (including the Inference Engine) for similar KBES construction. Current development packages offer comprehensive knowledge representation, more intensive search methods, and friendlier interfaces.

Instead of buying commercial software, a KBES designer can build an entire system from scratch (custom designed). The advantage of this method is that the final product directly addresses the user's needs in the most efficient manner. There are no excessive functions and data is stored and analyzed as desired. The disadvantage of a custom designed KBES is that it takes substantially longer to program and debug. Developmental costs will be higher because programming expertise is required. Typical development time for a commercial level KBES is approximately five years. When time, facilities, and programming ability are limited, commercial packages may be a more cost effective approach **[Price 1990] [Edwards 1991].**

Chapter 3. KBES Tools and Concepts

This chapter presents the artificial intelligence (AI) concepts and functions useful for ship grounding characterization. I describe several knowledge processing abilities of Nexpert Object, a commercial KBES software package, to demonstrate their application. These tools are used in Parts II and III for KBES design and development.

A. Knowledge Representation

As stated in Chapter 2, most expert systems store knowledge in a rule format for convenient representation and retrieval. Many newer expert systems have incorporated other representation forms for information poorly suited to rule format. Complex structures, for example, would require many rules to define such properties as geometry, dimensions, mechanical properties, and visual appearance **[Neuron Data 1987]**. Hierarchical systems, such as ship classification, also require more efficient representation formats than the rule.

Frames and *semantic nets* are two common methods for representing objects and classified systems **[Edwards 1991]**. They are more practical than the rule for defining complex structures with numerous components and intricate relationships. Frame-based representation provides a location and a name for each object. Objects properties and associated tasks stay with the object. For semantic nets, the developer defines the object and its properties as one entity, and associated tasks and relationship to other objects as another entity. This allows the same relationships to be used over and over, without having to redefine it for each object. For both types of object representation, the object's unique and non-unique attributes are recognized. Unique traits establish the object as a single entity; non-unique traits describe group relationships between objects. By creating unique and non-unique attributes, the designer establishes a classification system for a group of objects.

Nexpert Object, for example, organizes objects into a network of classes, subclasses, objects, and sub-objects. *Inheritance paths* defined by the designer, dictate which object properties are shared and with whom. Shared properties can pass between all levels of classification. Object related *methods* are tasks associated with a particular object or group of objects, so that when certain conditions are true, the objects will inherit value

for some property. Object-oriented knowledge representation will produce a full representation of structures and intricate systems. **[Neuron Data 1994]**

Figure 3.1 is a graphical representation of a frame or semantic net. Each circle represents a class, subclass, object, or sub-object that has unique and non-unique properties **[Siddall 1990]**. Lines between circles define relationships, along which non-unique properties travel. Frames have *slots*, or subroutines, associated with object properties. Semantic nets have specially defined linkages that further characterize relationships between objects. The two are similar. Using rules over either of these methods is far less efficient for object-oriented purposes because the repetition consumes computer memory. Frames and semantic nets complement rules in problem solving tasks, and together they provide a more comprehensive knowledge representation **[Siddall 1990]**.

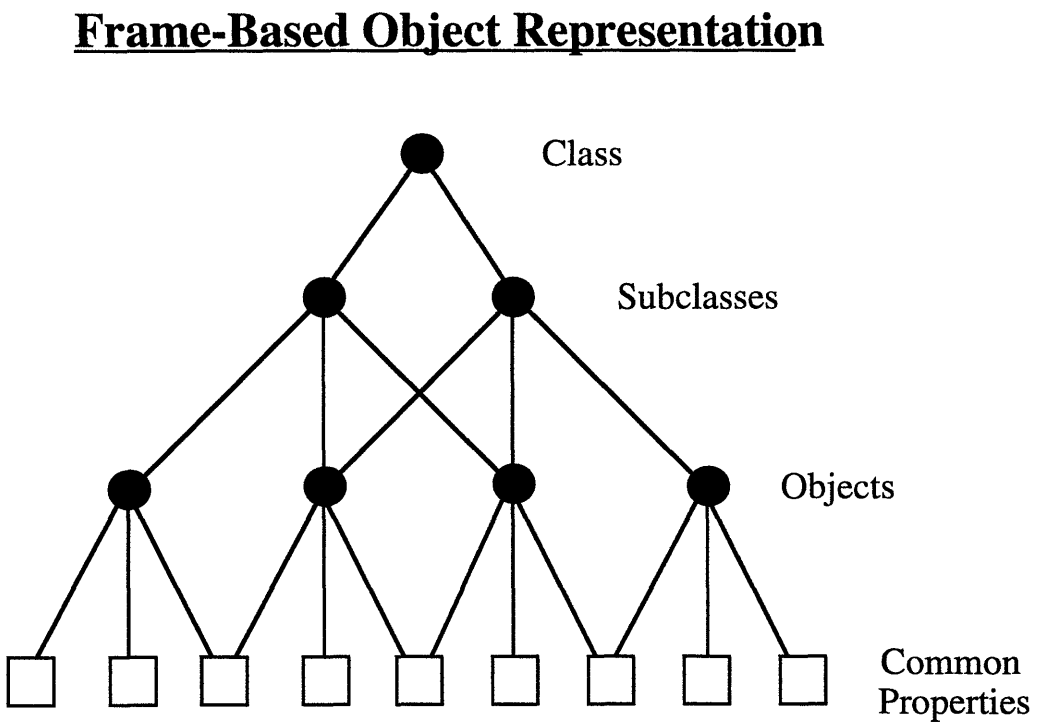

Figure 3.1. Frames and semantic nets are frequently used to represent complex structures and hierarchical systems. The circles signify objects, classes, and subclasses, and squares indicate common properties shared between them. The lines show the inheritance paths, or the routes along which properties travel. Each object also has unshared properties (not shown) that distinguish it from other objects.

The programming language used to create a KBES ultimately affects knowledge representation. Languages commonly used for expert systems include: LISP, Prolog, Basic, C, FORTRAN, and Pascal. Simple languages, like Basic and C, increase the system's flexibility because the programmer defines all functions and because they are compatible across platforms **[Price 1990]**. The disadvantage is that these languages require more development time because they developer must define everything. In contrast, more complex languages like LISP and Prolog offer predefined functions which speeds development time but reduces system flexibility. If a KBES designer wants to design a custom system, selecting the proper programming language must be given proper consideration.

B. Knowledge Search and Acquisition

There are a several common search methods currently used for expert systems. These include: forward chaining, backward chaining, and combination chaining. Early KBES's used to employ either forward or backward chaining methods, but more recent systems often use a combination of both.

B-1 Forward Chaining

Forward chaining, also called a data-driven search method, uses objects with defined values (usually defined by the user usually) to investigate all rules referring to that object. If one of these rules is satisfied, and its hypothesis is true, the inference machine adds that hypothesis value to the list of defined objects. New rules that refer to this satisfied hypothesis are now evaluated. The chaining ends when all rules referring to defined values have been checked. The satisfied hypotheses are the solutions. Theoretically, if one rule fires, forward chaining has produced an "answer" to the user's input. There may be more than one answer from this search process, all of which will be true for the conditions given. Forward chaining gives the "best" answer for the conditions, but this answer may not directly address the user's concerns. The processing time for this search is longer as all rules are checked **[Neuron data 1994]**.

B-2 Backward Chaining

Backward chaining, called a data-seeking search method, starts with suggested hypotheses, or guessed solutions. The system tries to satisfy the conditions for each suggested hypotheses to prove it true or false. If a hypothesis is a condition of the investigated rule, then its conditions will also be tested. All suggested hypotheses and related hypotheses get tested. This search method is focused on the chosen topic; it will directly answer the suggested conclusion. Unrelated rules are not checked. This method ensures that the user gets an answer related to their question, although this answer may not be the "best" answer for the existing condition. Backward chaining is faster than forward chaining because the system checks fewer rules **[Neuron data 1994]**.

B-3 Combined Searches

Combined searches use both forward and backward chaining. For example, if the user suggests a conclusion, the system might first backward chain to satisfy its conditions. When these rule have been exhausted, the system forward chain to find related rule to investigate. On the other hand, if the user provide initial values, the system first forward chains to find relevant rules and then backward chains to thoroughly investigate these rules. The combination of forward and backward chaining optimizes speed and thoroughness. **[Neuron data 1994]**

C. Nexpert Object Logic Shell

Nexpert Object is one of many commercial software packages that provides a logic shell for expert system development. Similar to most knowledge based systems, it has the basic components introduced in Chapter 2. Nexpert also has some of the more powerful internal functions mentioned above. Because it is programmed in C, Nexpert is compatible with most external software and runs on most computer platforms (Macintosh, IBM PC/AT, UNIX work stations, and IBM mainframes). Nexpert Object offers both rule and object-oriented knowledge representation, and it searches with forward and backward chaining. **Figure 3.2** shows possible components for a Nexpert system. **[Neuron Data 1987] [Price 1990]**

Nexpert Object Components

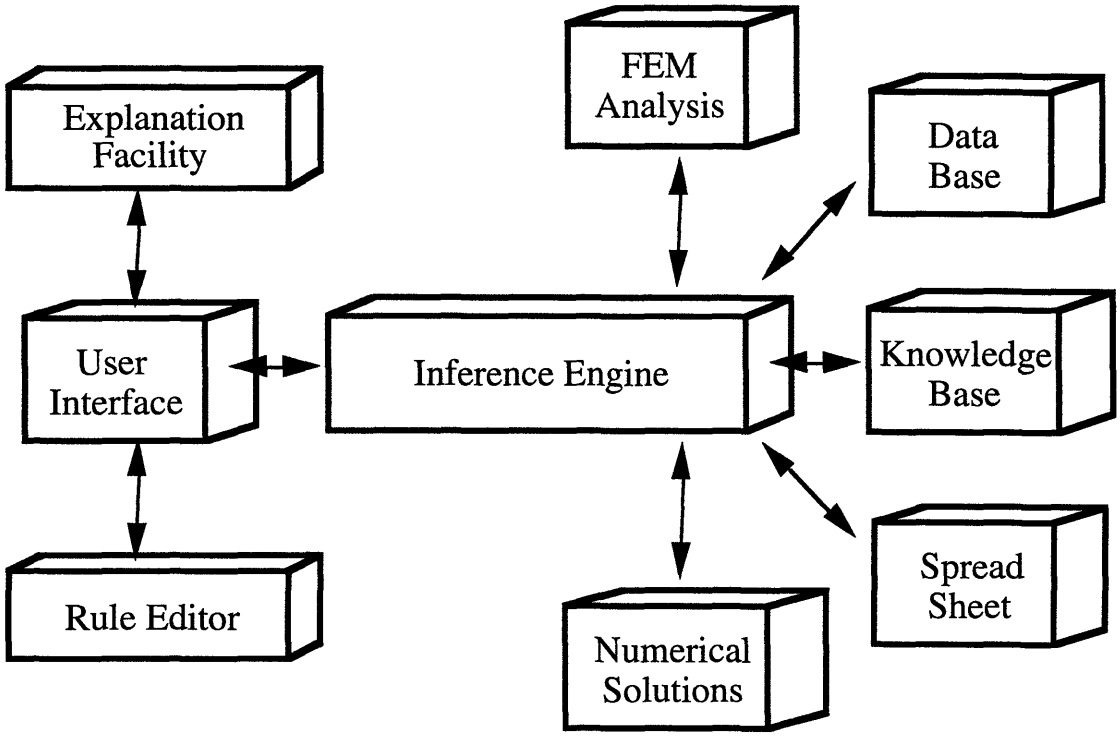

Figure 3.2. The Nexpert system contains the basic KBES components: the User Interface, Inference Engine, and Knowledge Base. The system also includes links to external information sources, such as databases, spread sheets, and software applications.

C-1 Internal Nexpert Features

NEXPERT is well suited to engineering tasks and structural analysis because of several internal features:

- **Rule based representation** -- The Nexpert rule has an **If/Then/Do** format (**Figure 2.2**). This format is called an augmented rule because the Inference Engine uses both forward and backward chaining patterns between right and left sides. (combination search method). For engineering tasks, this combination provides the added flexibility of inductive and deductive reasoning **[Neuron Data 1994] [Price 1990].**

- **Object representation** -- Nexpert provides frame-based knowledge representation. The designer defines classes, subclasses, objects, sub-objects, and their respective properties. Objects have *property slots*, or predefined locations, that hold *property values* for objects and classes. These reserved spaces are the tools that fully characterize an object. For example, a KBES that determines if the color of a ball is red, does not know the ball color *a priori*. The designer reserves a location for this property so that the system can ask for or determine the ball's color. Because it uses rules and frames, Nexpert is a *hybrid system*. To an engineer, this name means that complex objects and expert knowledge are efficiently represented in one system. **[Neuron Data 1994] [Price 1990]**

- **Rule and Object Editors** -- *Rule* and *Object editors* allow the KBES developer and the user to easily expand or alter the system without introducing syntax errors. This expandability is a major advantage to using commercial packages. Automatic editing reduces debugging time and encourages non-programmers to build and adapt their own expert systems. For an engineer, the emphasis shifts from a programming task to an engineering task **[Neuron Data 1994] [Price 1990]**.

- **Explanation Facility** -- The explanation facility operates during the solution process. Upon request, it provides both visual and written explanations of logic sequences used for a conclusion. The explanation facility is particularly important for engineering tasks. By recreating the process, the KBES either justifies its answer or blatantly displays knowledge gaps and conflicting information. The engineer gains confidence in the system when all evidence is available for scrutiny **[Neuron Data 1994]**.

- **Strong mathematics facilities** -- The NEXPERT system itself handles all mathematical functions defined by the C language. For complex mathematical routines, Nexpert can call external analysis programs **[Neuron Data 1994] [Price 1990]**.

- **Graphics abilities** -- The NEXPERT system has strong graphic facilities and will provide visual data in conjunction with analysis procedures. Photographs, charts, and pictures are often used for describing and requesting information **[Neuron Data 1994]**.

C.2 External Nexpert Features

Nexpert was designed to interact with other computer programs **[Price 1990]**. This feature means that computationally intensive programs and large data banks can lie outside the expert system, to be called by the Inference Engine only when needed. As a consequence, the KBES is relatively focused and efficient, holding only those rules needed to process information. This feature makes it suitable for engineering and industrial problems, which commonly involve a wide variety of resources **[Price 1990]**.

Table 3.1 lists internally supported software for Nexpert. Internal support means that pre-defined commands exist to access data from outside sources. Unsupported software is also accessible, but requires additional programming. Spreadsheets and databases hold data in simple table or list format. The relational database uses multiple tables and has its own query languages for navigating the tables. **[Neuron Data 1987]**

Table 3.1. External Software Supported by NEXPERT

Function	Supported Programs
Spreadsheet	Excel Lotus 1-2-3 Nexpert Spreadsheet
Database/Tables	Excel Lotus 1-2-3 Nexpert Database dBase III
Relational Database	Oracle SQL Digital's RDB

[from Neuron Data 1987]

Easy linkage to external software also allows other computer programs to invoke Nexpert. For example, Nexpert could nest inside an FEM program and conduct input/output routines. In this case, the user sketches a structure and the KBES asks questions about the kind of analysis desired, the loading conditions, boundary conditions, materials, and degree of accuracy. The KBES defines the number and size of the elements for the structure and sends the information to the FEM routine. After the analysis is complete, the KBES returns the FEM results with suggestions for design improvements back to the user. For this application, the KBES serves as a buffer between the user and a difficult software application.

To determine the most appropriate commercial logic shell for a project, a KBES designer should analyze system features with specific tasks in mind. I chose the Nexpert Object Version 3.0 for KBES development. Nexpert Object is adequate for the a ship grounding analysis project because of the features described in previous sections, such as: rule and object representation, forward and backward chaining, explanation facilities, automatic editing and formatting, strong mathematical capabilities, visual representation,

and convenient linkage to external data sources. Other commercial logic shells have comparable properties and would also be suitable for a tanker grounding project, but Nexpert Object was readily available and relatively inexpensive. For a review of other commercial packages, please refer to **[Price 1990] [Maher 1987].** In the interest of time, I have chose a commercial package over a custom designed system, although the ;latter would result in a more efficient system. From this section forward, all expert system features refer to those of Nexpert Object.

D. KBES Design Methodologies

Because knowledge based systems are relatively new, no formal development procedures exist **[Edwards 1991]**. Several KBES researchers suggest general methodologies for building expert systems, but most existing systems used their own techniques. The underlying steps for KBES development resembles standard design procedures. A commonly cited method comes from **[Waterman 1986]**, and includes the following steps:

- **Problem Identification**. The first step in developing a KBES is to identify the problems. This involves determining and characterizing potential users, defining project goals, limiting the project's scope, and locating resources. After defining the problems, a design should determine if the KBES is an appropriate tool for addressing them.

- **Procedure Conceptualization**. After establishing goals, users, and resources, the KBES designer, with advice from experts, determines the analysis required to reach each goal. Each analysis may break into tasks, sub tasks, and related parameters. Experts provide the insight, background knowledge, and procedures for defining and solving the problem addressed by the KBES.

- **Procedure Formalization**. The tasks above require formalization to computer language. Commercial packages typically specify formats. It is important to minimize formatting errors, because they will be difficult to detect at later stages of development .

- **Program Implementation**. This step involves instantiating the system with real data, and combining the above procedures into a comprehensive system.

- **Testing and Debugging**. Testing includes evaluating the system's accuracy, syntax, and ability to meet initial goals.

Waterman has provided a rough guide for KBES development. Many newer methods have these basics but add more details, depending on the project's goals **[Edwards 1991] [Adeli 1988]**. The design process is iterative, as the project continuously changes as it develops.

Part II

The Global KBES Design

Part II presents my Global KBES Design for Ship Grounding Analysis. Chapter 4 discusses current approaches to ship grounding and indicates why KBES development is useful. This chapter also introduces my design methods. Chapter 5 is the proposal for the Global KBES; it addresses the features expected for such a system. Following the Top-Down approach, I indicate potential users and topics for the Global KBES. Chapter 6 presents the actual Global Design, describes its components, and talks about what is required to construct and implement such a system.

It should be noted that the Global KBES for Ship Grounding has not been implemented; this task is well beyond the scope of my project. Part III, however, describes the construction and operation of a portion of the Global Design.

Chapter 4. KBES Development for Tanker Groundings

This chapter explores ship grounding as a justifiable topic for KBES representation. Grounding information comes primarily from reports from and members of the MIT-Industry Project for Tanker Safety. Following the guidelines presented in Chapter 3, I have used Tanker Safety Project's goals, tasks, and experts to design a full Knowledge Based Expert System for analyzing and characterizing grounding events.

A. Current Approaches to Ship Grounding

Currently, the shipping community does not design ship hulls to withstand grounding forces, and this practice has resulted in several recent environmental catastrophes **[Wierzbicki, Rady, and Peer 1990].** The U.S. Coast Guard's OPA 1990 ruling shows one approach to this problem: stricter hull regulations to prevent environmental damage. Other members of the shipping industry feel that harsh regulations could radically increase ship construction and operation costs; they desire less severe alternatives. This debate prompted numerous studies to characterize grounding events in order to develop both economically practical and grounding-resistant hulls.

Few analytical methods comprehensively characterize and predict hull damage from grounding incidents **[Wierzbicki, Rady, and Peer 1990]**. This task requires modeling a combination of complex structures and random variables. Grounding analysis methods must address hull characterization as well as random and non-random grounding parameters.

A-1 The Variables

A ship's hull is an intricate network of rigidly connected structures and structural supports, making hull characterization a difficult task. Many hull components serve as both structural members and reinforcements for other components **[Taylor 1985] [Eyres 1988]**. For example, a network of girders strengthens and stiffens bulkheads. The girders are strengthened in the same manner by another set of stiffeners. At a ship's bottom, where grounding events occur, the hull resembles a cross-hatched pattern of rigidly connected members. These interdependencies, and the wide variety of ship

structures, complicate structural analysis. Although rigorous analytical methods are often computationally intensive and expensive, the demand for such exercises has increased **[McDonald 1993]**.

Grounding parameters vary as widely as ship hull structures, further complicating grounding analysis development. These parameters divide into random and non-random variables. Non-random variables include ship dimensions, operating conditions (speed and trim), onboard loads (cargo), and ocean conditions (general wave height and wind speed). Other parameters are completely random and impossible to determine precisely or predict in advance. These include grounding obstacle character (size and shape), individual wind and wave effects at time of impact, ship position, grounding location, total grounding time, and stranding damage. The presence random variables means there are infinite potential grounding scenarios. Because of this uncertainty, unique solutions become difficult. Probability methods, estimations, and past experience, however, can provide value ranges for random variables. **[Gonzalez 1994]**

Although present studies address grounding parameters individually, there are few comprehensive studies exploring the interactions between hull structure, definable parameters, and random variables **[Bracco 1994].**

A-2 Simplification Methods

Simplifying grounding parameters is an important technique for reducing analytical complexity. Past studies have accomplished this simplification in several ways. One approach is to model relatively small, geometrically simple hull structures. These studies often incorporate complex loading conditions to achieve a realistic characterization. This technique emphasizes failure modes under different loading conditions. An alternative method is to use simple loading conditions and more complex structures. This technique emphasizes component interactions during structural failure.

For studies involving very large and complex structures, analyses often apply both loading and structural simplification. Loading may be modeled as point loads at computationally convenient locations. Structural simplification often involves "smearing" techniques which blend smaller components into larger ones. This practice accounts for material volume and eliminates geometric details. Although simplifications give an incomplete representation, it does provide useful information. Combining the studies mentioned above into one system would describe ship grounding behaviors more fully.

A-3 Data Formats

A system that combines analytical techniques must be compatible with multiple data formats and numeric routines. **Table 4.1** gives examples of different approaches used for grounding analysis, and the types of data they generate.

Table 4.1. Approaches to Ship Grounding Analysis

Approach	Result Format
FEM/FEA analysis	critical loads, strains, deformation models
analytical approaches	equations, numeric results
semi-empirical models	numeric data, graphs, photographs, equations, failure descriptions
database analysis	estimations, pictures, descriptions

B. Justifying a KBES for Ship Grounding Analysis

There are many alternatives for studying ship grounding events. This flexibility is demonstrated by the Tanker Safety Project, where promised deliverables include a CAD program and welding handbook. Since KBES development is time consuming, the designer must justify its use. Not only must a KBES be appropriate for a particular problem, the designer must have proper access to resources, time for development and maintenance, and a receptive audience.

B-1 Is KBES Analysis Appropriate?

Referring to conditions proposed in Chapter 2, knowledge based expert systems are useful when:

- No formal procedures govern practice.
- Knowledge and experience governs practice.
- Human experts are scarce or unavailable.
- Information is widely dispersed or informally stored.

[Siddall 1990]

At present, there are *no formal approaches* or universally accepted methods for analyzing groundings events, predicting ship hull survivability, or estimating hull damage in grounding conditions. Because of this lack, *no ship grounding experts* exist. Experts exist for different aspects of ship grounding, but they are spread among academic institutions, shipping companies, regulatory agencies, and military groups. It is unlikely that any of the experts alone could provide a comprehensive characterization of ship grounding, and accessing many experts at one time is impractical.

Because ship manufacturing and operation are old professions, *knowledge and experience* still govern many of today's practices **[Masubuchi 1994]**. This information is empirical and informal, but it is useful when more exact information is unavailable. Scientific techniques have gradually worked their way into shipping practices, joining with the older practices **[Gonzalez 1994]**.

In light of the above conditions, ship grounding characterization fits a KBES problem profile. This fact by itself, however, does not justify its use. A KBES must be practical to build.

B-2 Is KBES Development Practical?

For KBES development to be practical, the designer must have access to the necessary resources during development. Resources must also be available for maintaining and updating the system. Computer facilities, KBES developers, and computer-receptive users are required, and development time should be reasonable. Finding computer-receptive audiences is often a significant problem in traditional fields **[Masubuchi 1994]**.

A combination of past grounding-related analyses and the information generated by the Tanker Safety Project would provide a more comprehensive grounding representation.

The MIT Tanker Safety project has recognized the strength of a combined approach to ship grounding **[Wierzbicki, Rady, and Peer 1990]**. The Tanker Safety Project has participants from both scientific and shipping communities, which provides it with a wide spectrum of expertise in areas related to ship grounding. Because of this collaboration, old practices are being re-evaluated and new ones created. It is both timely and practical to develop a KBES in conjunction with this project.

B-3 What advantages result from KBES development?

As mentioned in Chapter 2, the process of developing an expert system provides its own advantages. Those particular to ship grounding include:

- comparing past, present, and future grounding analysis methods,
- locating logic and information gaps, and
- targeting areas for future grounding research.

The process of linking grounding research to shipping problems is a useful exercise, but not an easy task. For example, suppose a company asks an academic researcher for help applying his or her research to company problems. The researcher presents his or her models, tests, and final conclusions to a company engineer. Whether this information gets passed on to a manager for further consideration depends on the engineer's understanding of the subject, their evaluation of the information's merit, and whether or not the assumptions and solution forms are applicable to company processes. If the company engineer recommends research results to a manager, it passes through another evaluation. The manager must look at implementation issues such as feasibility, cost, shut down time, reliability, and benefits. Whether or not academic research reaches industrial application depends on parameters beyond its scientific merit **[Masubuchi 1994]**.

B-4 What advantages result from a ship grounding KBES?

The final KBES will be useful to many people, both providers and users of grounding information . For experts, the KBES helps orient studies towards practical results, so that it stands a better chance of reaching implementation stages. Company engineers can use the KBES to match research assumptions and conditions and research results to company problems and methods, thereby judging its merit and practicality. Upper level management can use the KBES to determine if implementation advantages outweighs the cost of equipment down time required to alter manufacturing processes.

The result of KBES development is a computer program that answers ship grounding questions, such as:

- What structural change can I make to improve hull survivability in grounding situations without significantly increasing manufacturing cost, time, or feasibility?
- What optimal ship speeds reduce total grounding damage for a 150,000 ton single hulled oil tanker?
- I have a list of failed components and their damage dimensions; how fast was the ship traveling at the time of impact?
- Last year, international regulations required tankers to use 9 mm welds to join primary hull structures; how many ships complying with this regulation suffered severe damage during grounding? Is this performance better or worse than before the rule's implementation?
- How much weld repair is necessary for a minor hull puncture, and how will repairs affect the fatigue life of the joint and the surrounding structures?

C. Design Approach

There are two approaches for designing the Global KBES, the Top-Down and Bottom-Up methods **[Maher 1987]**. There are advantages and disadvantages associated with each method. In the following sections, I describe how to use each method and its strength and weaknesses. The last section describes my approach to the Global KBES Design for Ship Grounding, which starts with a Top-Down and ends with the Bottom-Up method.

C-1 Top-Down

The *Top-Down* approach starts with a project's final goals, and works backwards to establish the processes and data required to reach these goals. This design process resembles that suggested by **[Waterman 1986]** described in Chapter 3.

In the first step of the Top-Down approach, the designer *identifies potential KBES users* and determines their interest in ship grounding events. Depending on their role in the shipping community, each user will have unique demands for a grounding oriented computer system. This first step identifies specific goals for the system. By starting with

users and their grounding questions, the designer ensures an end product that directly addresses real grounding issues.

The second step of this approach is to *determine appropriate analytical methods* for goals established in the first step. The KBES designer gets this information from experts, who choose procedures based on initial conditions, desired output, and available information. A complete solution procedure may include one or more analytical methods, and each analysis subsequently breaks into its associated tasks and sub-tasks. The entire solution procedure resembles a logic chain (discussed earlier).

The final design step involves formalizing the above procedures into KBES format and eliminating logic errors. The complete development time for a medium sized, non-commercial KBES is typically three to four man-years.

C-2 Bottom-Up

Research for much of the Tanker Safety Project follows a *Bottom-Up* approach. This method involves incorporating available information and finding useful ways of applying it. This is more of a construction approach to KBES design. For example, fracture or failure analysis of complicated ship structures begins by characterizing very basic structures. Researchers combine these characterizations in ways that simulates the behavior of more intricate structures. The KBES would store these components and offer them as building blocks for the analysis of other structures. This method is more appropriate organizing, applying, and verifying new research information when relationships between topics has not yet been established.

C-3 Approach for the Global KBES Design

The approach I used to develop the Global KBES uses the strengths of both Top-Down and the Bottom-Up methods. The primary advantage of the Top-Down approach is that the resulting KBES directly addresses specific grounding issues. The system is designed around the desired outcome. This method is particularly useful for the first stages of the Global KBES, because I am trying to make a system that answers grounding questions of the shipping community. Chapter 5 uses this method extensively.

For the developmental stages of the Global KBES, I chose a Bottom-Up approach. This method appears in Part III when I develop a working KBES for analyzing fillet welds. The Bottom-Up method is more exploratory and constructive than the Top-Down approach. It is good for organizing the fillet weld research generated in the Tanker Safety Project. The strength is that new research information gets combined into one system, and

the user can explore different ways to combine that data. Although this is often the easiest way to combine new information whose interactions have not yet been determined, it is not the best method for answering specific question about grounding. The combination of the two methods, the Top-Down for the earlier stages of design and the Bottom-Up for later stages of development, provide the strongest approach to the Global KBES for Ship Grounding.

Chapter 5. Global KBES Proposal

In this chapter, I present a proposal for the Global KBES Design for Ship Grounding, also referred to as the Global KBES or Global Design. I created this system to deliver research information, ship grounding knowledge and data, and common shipping practices to non-expert members in the shipping community. Solutions arrive in a format that directly addresses their grounding concerns. It is a comprehensive system, and the design is intended to include all grounding issues, if desired.

A. Global Perspective

To be useful to the shipping community, a ship grounding KBES must have broad capabilities. In Chapter 4, I described the complex combination of grounding events and ship structures. Ideally, a comprehensive Global System should have the tools and resources for addressing all possible grounding issues. A computer system that is excessively broad, however, becomes too diffuse to address specific issues with reasonable rigor **[Gonzalez 1994]**. A useful system is more focused and detailed, requiring system limitations. So, the question becomes "How does one design a comprehensive Global KBES that exhibits some analytical gusto without overwhelming its users?"

My solution to this problem is to design a broad system with the potential to include all grounding issues. It is important in the early stages of design not to impose too many limitations, so that the same design applies to a broad spectrum of users **[Gonzalez 1994]**. Limits imposed at later stages of development will make the system more focused for particular uses. Chapter 7 describes this technique in detail. The following sections expand the Global KBES Design for Ship Grounding to include the needs of all shipping community members.

B. Design Issues

To create a Global KBES, the design must have the ability to include all shipping and grounding related issues. I have decided, with the help of Professor Francisco

Fernandez Gonzalez **[Gonzalez 1994]** on a list of critical areas that a Global KBES must cover. It must:

- Address the needs of major shipping interests, and possess expansion capabilities to include future shipping interests.
- Refer to real ship structures using proper terminology, geometry, dimensions, and operations. It should have the expansion capability to include future hull structures.
- Consider all types of grounding occurrences, with the ability to add more.
- Include information from real grounding events, and provide links to casualty databases and external data collections.
- Provide expert knowledge, analysis, methods and research information from all fields related to ship grounding.
- Acknowledge international and national regulations, common building and operating practices, cost considerations, and common repair procedures.

The following sections explore the topics listed above, and explain why these are critical components in a Global KBES Design. This exercise is part of the Top-Down approach; it attempts to fully and directly address issues most important to the shipping community.

C. System Users

The Global KBES Design should also have the ability to include all shipping interests. The first step in the Top-Down approach is to identify all potential users, the shipping community members. This step also involves determining the priorities and concerns of these users, which help specify their interest in ship grounding events. Ship grounding information is useful to all of these users, but each may want a Ship Grounding KBES for different purposes. The goal is to design one system that will hold all grounding information in a format accessible to all interested parties. This avoids the repetition of creating separate systems for each party. Specific information can be drawn from this system according to individual needs.

The information each user can provide a Ship Grounding KBES varies with profession and technical background. For example, a ship operator may not have detailed information regarding hull plate materials. A ship repairer may not have precise information about a ship's normal operating conditions. These parameters will impact the

Global KBES's design, because alternative information sources will be required to fill in the user's knowledge gaps. **Table 5.1** lists potential Global KBES users with an emphasis on their role in the shipping community, their grounding concerns, and the information they might provide such a KBES.

Table 5.1. Potential Global KBES Users

Expected User	Priorities	Grounding Concerns	Available Information
Designer	structural stability operating performance safety cost manufacturability	hull survivability cost satisfying regulations	performance criteria structural details design techniques
Ship Builders	hull strength stability construction time cost	structural stability fabrication methods materials inspections liability	fabrication parameters material properties local dimensions quality assessment
Shipping Companies	cargo capacity transport time accident prevention	safety maximum efficiency grounding risks	cargo properties ballast load distributions trim ship speed sea conditions ship history
Insurance Companies	damage investigation liability cost estimations appraising ship value	grounding risk fault investigation damage predictions	accident reports photographs claims general ship data
Repair Companies	repair strength extent of repairs cost	damage extent repair procedures strength of repairs	damage descriptions damage dimensions repair procedures
Regulatory Agencies	construction rules operating rules promoting safe practices	pollution rule effectiveness predicting effects of new rules new hull designs	rule details operating practices hull design details building procedures

Table 5.1. This table lists potential users for a Global KBES for Ship Grounding. All users listed would benefit in some manner from a collection of grounding information. Since each user has different priorities, grounding concerns, and available information, the Global KBES Design must reflect these differences so that everyone can benefit.

One significant advantage of creating the Global KBES for Ship Grounding is that it can address the needs of many users, whose grounding interests and approaches to grounding may vary widely, as indicated in **Table 5.1**. The same Global KBES could help the designer optimize stiffener separation, weld size, low-stress geometry, structural

dimensions, costs, and manufacturing time. It could assist the manufacturer by optimizing fabrication practices, inspection procedures, repairs and reinforcement choices, hull materials, costs, and building time. A Global KBES could be used to predict potential cargo losses and ship damage for specific grounding scenarios and operating conditions. For fault investigation, a KBES could verify claimed operating speeds using energy balance analysis, damage dimensions, and claimed operating conditions. Since damage often involves ore than the immediate grounding area, a KBES could help repair companies explore damage extent, decide on proper repair procedures, and predict grounding survivability for repaired structures. For regulatory agencies, such a system could provide feedback for existing rules, and predict the effects of implementing regulations for current and future hull designs.

D. System Information

Shipping information comes from many different sources. Because shipping is an old profession, many practices have developed from experience and knowledge gathered over time **[Masubuchi 1994]**. Unformalized knowledge ("rules of thumb") should not be neglected for grounding characterization, since few reliable alternatives exist. The knowledge based system incorporates this kind of data form easily and combines it with more technical and formal information.

Grounding information often comes from empirical data collections and databases **[Gonzalez 1994]**. Agencies such as Lloyds and the U.S. Coast Guard maintain detailed casualty data bases on all major accidents **[Sinmao 1994]**. By incorporating data from spreadsheets, tables, and databases, the Global KBES could perform both a statistical and probabilistic analysis on specific types of grounding events. Real-life examples would supplement, as well as verify, experimental data and analytical grounding models.

D-1 Grounding Analyses

Techniques for grounding and hull analysis vary widely with respect to initial assumptions and final answer formats. *FEM/FEA*, an increasingly popular technique, gives a stress-strain analysis for different levels of structural detail. It involves a two or three dimensional figure, defined structural subunits, boundary conditions, loading conditions, and material properties. It determines failure loads, strain distributions, and stress concentrations; it also models structural deformation. Maintaining a high level of

structural detail as the FEM structure's size and complexity increases requires more computing power and longer computing times **[Atmadja 1994].** For large, complicated structures, FEM users typically define larger subunits and accept lower accuracy to accommodate their computing abilities and time constraints. An FEM analysis typically focuses on either a small, detailed structure; or a large, less defined structure.

Energy balances typically involve the whole ship. The solution applies energy conservation principles to account for kinetic energy dissipation during grounding events. Analysis of this type requires knowledge of ship operating conditions before and after the grounding event, a defined grounding scenario, acknowledgment of major external forces, ship structure details, and accurate damage measurements. Several energy methods exist, but they are not well suited to damage predictions, since solution procedures require the damage dimensions **[Poudret, Huther, et al. 1981]**. Energy balance methods are useful for calculating forces on the hull resulting from grounding **[Poudret, Huther, et al. 1981]**. The information generated from this kind of analysis might help guide future research on predictive methods.

Semi-empirical models or "*closed form solutions*" exist (and are currently being developed at MIT) for describing puncture, crushing, tearing, bending, and buckling of simple hull components. These analyses include simple geometry and loading conditions. Since real-life groundings will likely include a number of these components and failure modes, a combination of these solutions would provide a rigorous representation of a grounding event **[Wierzbicki, Rady, et al 1990]**. This method also falls somewhat short of a complete analytical grounding model for predicting damage, because assumptions are needed to define several critical grounding parameters (like rock size and hit locations).

Each analysis above has unique input demands and output formats. These are appropriate for some grounding questions, but not appropriate for others. The Global KBES can store these analysis methods and pre-analyzed solutions. When the user specifies a set of grounding conditions, the Global KBES chooses among its methods for the proper combination of solutions. The system retains the flexibility to incorporate new models and analysis methods over time, or to inform users when an analysis is unavailable.

D-2 Ship Components and Terminology

The Global KBES includes detailed references to current hull structures, and it allows user's to include company ships and new designs. Reference to hull components is

the common link between all forms of grounding analysis and characterization. For a KBES to include many analysis methods, it must have access to hull structure parameters for solution procedures. These details include the dimensions and geometry of very small to very large hull structures. Different grounding analysis methods will emphasize different portions of a hull, so it is important to have an extensive and detailed reference for these components. **[Gonzalez 1994]**

For example, a whole ship energy balance analysis might require references to larger structural members, such as outer hull dimensions and primary bulkhead locations. Large scale solutions commonly account for smaller components by "smearing" the volume evenly into the larger structures. A more detailed analysis, such as a weld failure, may include only the weld and immediate stiffeners; larger structures represent boundary conditions and their structural details are unnecessary.

Overall ship dimensions, bottom shell, and stiffeners and structures near the bottom are commonly involved in grounding accidents and analysis. The Global KBES should be able to characterize structural components, based on user input from any of these categories:

- **Overall dimensions** -- length, width, draft, and weight
- **Major structures** -- bulkheads, stiffeners, and machinery
- **Framing systems** -- longitudinal, transverse, or combinations
- **Hull types** -- single hull, double hull, and double bottom
- **Bottom structures** -- Base plate, keel, center girder, longitudinal margin brackets, welds, and vertical stiffeners

[Eyres 1988] [Taylor 1985]

This list is not exhaustive, but it does include the major structures near a ship's bottom. Bottom hull structures can be organized by their position in the ship. For example, bottom hull structures are spaced relatively uniformly within five regions of a ship's bottom: bow, cargo areas, machinery area, sides, and stern. Cargo areas typically have more widely spaced stiffeners compared to the other four regions. The others require extra stiffening support. International regulations specify minimum stiffener spacing and support structures depending on a ship's size, operation purpose, and general hull structure **[Taggart 1980] [Taylor 1985] [Eyres 1988]**.

Proper terminology, as established by international agencies, is critical for wide spread usage of the Global KBES for Ship Grounding. This is particularly important when referencing hull components **[Gonzalez 1994]**. Unclear labels could result in errors. Although there is no one universal terminology, the Global system should include a widely known system as a foundation, such as ISO standards.

D-3 Grounding Events

As discussed in the Chapter 4, each grounding event is unique, and random variables make the total number of possible grounding scenarios infinite. Although it is impossible and unnecessary to cover all aspects of grounding, it is useful to determine the most probable grounding events for a certain type of ship. By combining empirical and analytical data, this type of compromise analysis is possible. Combining such data is a primary strength of a KBES.

For example, casualty database records give examples of actual grounding occurrences **[Sinmao 1994]** . A survey or statistical analysis of grounding records could determine the ship structures most commonly involved in grounding accidents. This survey would narrow possible grounding events to a number reasonable for subsequent research.

One possible use of a Global KBES is to recreate grounding incidents as a "sequence of events" **[Gonzalez 1994]**. Ideally, each event would ideally be fully characterized by a closed-form analytical solution, like those developing at MIT. With this information, the Global KBES may be able to predict grounding damage for common hull locations given the operating conditions. We cannot design against every conceivable grounding event, but we should be able to design against the most likely forms of damage.

Chapter 6. Global KBES Design

This chapter presents the structure of the Global KBES Design for Ship Grounding. I review the system's components, how they operate, and how they are useful to the Global KBES. The following sections also describe formats and specific components of the Global KBES Design for Ship Grounding. They suggest one strategy for incorporating all the information presented in the previous chapter. **Figure 6.1** shows the General KBES Format, which is comprised of User Interfaces, the Inference Engine, the Structural Reference, Knowledge Bases, and External Applications.

The User Interface controls information input for queries and searches in the Global KBES. The Inference Engine relays information between User Interfaces and Knowledge Bases. The Structural Reference contains ship hull information to be used by the Inference Engine to obtain grounding analysis parameters. External Applications hold empirical data and computer software routines. The Nexpert Object logic shell links components of the Global system by creating internal and external links between files using the "C" programming language.

A. User Interfaces

The User Interface accesses parts of the Global KBES Design that are most important to a particular user. Since each user does not require all the information contained in the Global KBES, nor all possible analysis methods, each category of users (see **Figure 6.1**), would use a different interface. The interface contains computer screens to control input and output procedures. Its organization should present the contents and analysis option to the user in a user-friendly format. As mentioned earlier, the Global KBES that combines all grounding information is too broad for any one user. By customizing the User Interface, the designer produces a subset of the Global KBES that is useful to certain members of the shipping community.

For an example of User Interface screens, please refer to, Chapter 7, section C.

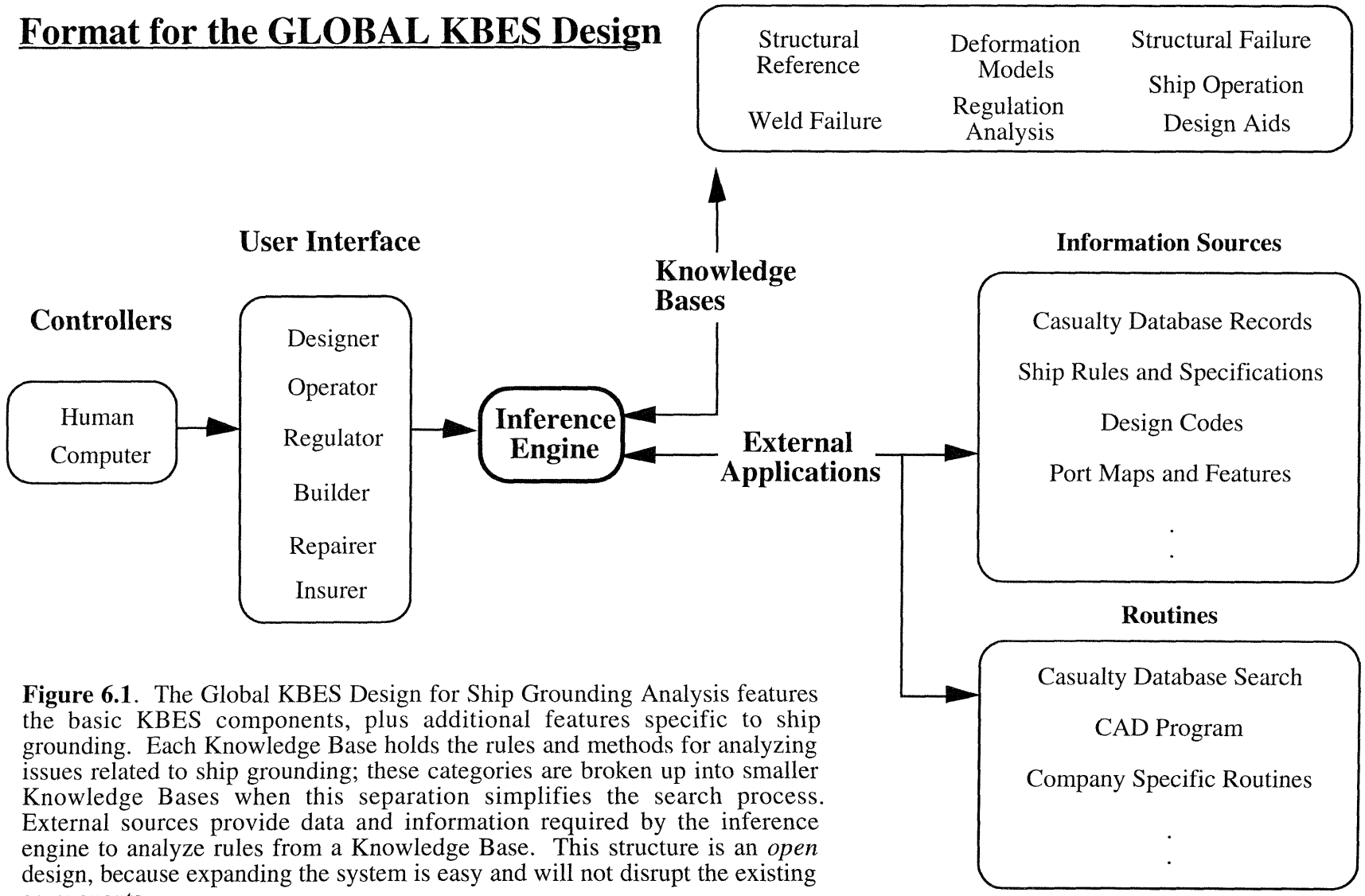

Figure 6.1. The Global KBES Design for Ship Grounding Analysis features the basic KBES components, plus additional features specific to ship grounding. Each Knowledge Base holds the rules and methods for analyzing issues related to ship grounding; these categories are broken up into smaller Knowledge Bases when this separation simplifies the search process. External sources provide data and information required by the inference engine to analyze rules from a Knowledge Base. This structure is an *open* design, because expanding the system is easy and will not disrupt the existing components.

B. System Controllers

The Global KBES can be operated by a human user or another computer. **[Neuron Data 1994].** Prior discussions assumed a human operator, who presents a problem to the User Interface, provides relevant information, and accepts offered solutions. With a human operator, the system can be interactive; the Inference Engine can request additional information or address related topics during an analysis procedure. The system's output can also be delivered in a variety of formats, such as pictures, text explanations, or numbers.

The Global KBES can also be controlled by another computer application. For example, a KBES could be use sensors on manufacturing equipment to regulate operation. KBES results and suggestions would depend on sensor readings, and could be automatically executed or sent to a plant worker. Such a system loses interaction flexibility, however, because the human is taken out of the loop. If data to the KBES is insufficient, the sensors cannot be asked for alternative information. The output of such a system must also have a specific format if the actions are for other machines.

A third method of control involves combining human and computer control of the KBES. A KBES can be nested in a CAD program to operate input and output processes. The CAD program calls the KBES and initiates most procedures, but the user is allowed to provide some information and receive suggestions from the KBES.

C. Structural Reference

Grounding analysis refers to a part, or all, of a ship's hull. Although the pertinent components vary with analysis, hull components are the common link between grounding solutions. The Structural Reference is an independent unit, called by the Inference Engine during an analysis. It takes advantage of Nexpert Object's object-oriented representation and its property inheritance paths which are explained in Chapter 3. The Structural Reference provides geometric and dimensional information for different parts of a ship. The Structural Reference has two major features, its hull classification scheme and its referencing ability. These components allow the Structural Reference to characterize ship components based on various levels of user input.

Terminology within the Structural Reference must be consistent with international standards to avoid confusion. ISO has already established standards for communication

among international shipping community members. By accepting their definitions, more users will be able to participate in a Global KBES project **[Gonzalez 1994].**

C-1 Hull Classification

To describe and define ship structures using an object-oriented representation, relationships must be established between ship hull components. Each component will have its own unique features, such as: length, height, thickness, number of welds, location. The parts will also have non-unique characteristics like: is a stiffener, is a bottom hull component, is connected to another object. Because ship hulls are an intricate, connected network of small to large stiffeners, it exhibits a natural classification that includes all of its components. The KBES can find hull specific information easily if it is represented as an object-oriented hierarchy, where the designer has determined the best classification scheme and defined the relationships between structural components. Ship hull structures lend themselves to this type of classification. **Table 6.1** shows several commonly used categorization schemes used to distinguish between different kinds of ships.

Depending on the topic of discussion, people group ships into different categories, as is shown in **Table 6.1 [Eyres 1988]**. For the Structural Reference, hull organization should be based a system that refers to hull structures most likely involved in grounding, i.e. bottom hull structures. Of the four categories, Hull Types and Framing Systems provide the most information about a ship's bottom structure. For example, a double bottomed hull indicates the presence of two horizontal plates with vertical stiffeners connecting them. Combinational stiffening indicates the presence of both longitudinal and transverse stiffeners connected to the hull bottom. Because these two categories provide pertinent information about the bottom of a ship, I have chosen them as the principal classes of my system.

Table 6.1. Ship Hull Classification System

Hull Type:	Single Hull Double Hull Double Bottom
Framing System:	Transverse Longitudinal Combination
General Dimensions:	Dead Weight Tonnage Length Between Perpendiculars Length at Waterline
Ship Name by Usage:	General Cargo Container ship Barge Carrier Bulk Carrier LNG's OBO's Tankers

Table 6.1. To accommodate different grounding analysis methods, the Global KBES must be able to reference and provide information about ship hull dimensions and geometry. The table shows different ways the shipping community commonly classifies ships. Use of these categories depends on the topic of discussion.

Based on hull descriptions from ship architecture literature **[Taggart 1980]** **[Taylor 1985]**, I constructed **Figure 6.2**. This figure shows the object-oriented classification scheme for the Structural Reference portion of the Global KBES. Starting with Hull Type as the primary class, standard ship structures subdivide easily into classes, subclasses, objects, subobjects, and their respective properties. The object tree shown in **Figure 6.2**, similar to an animal classification scheme, divides the hull into a hierarchy of independent and related objects. The designer defines how the properties of these objects are shared, inherited, or passed between other objects and classes within the system.

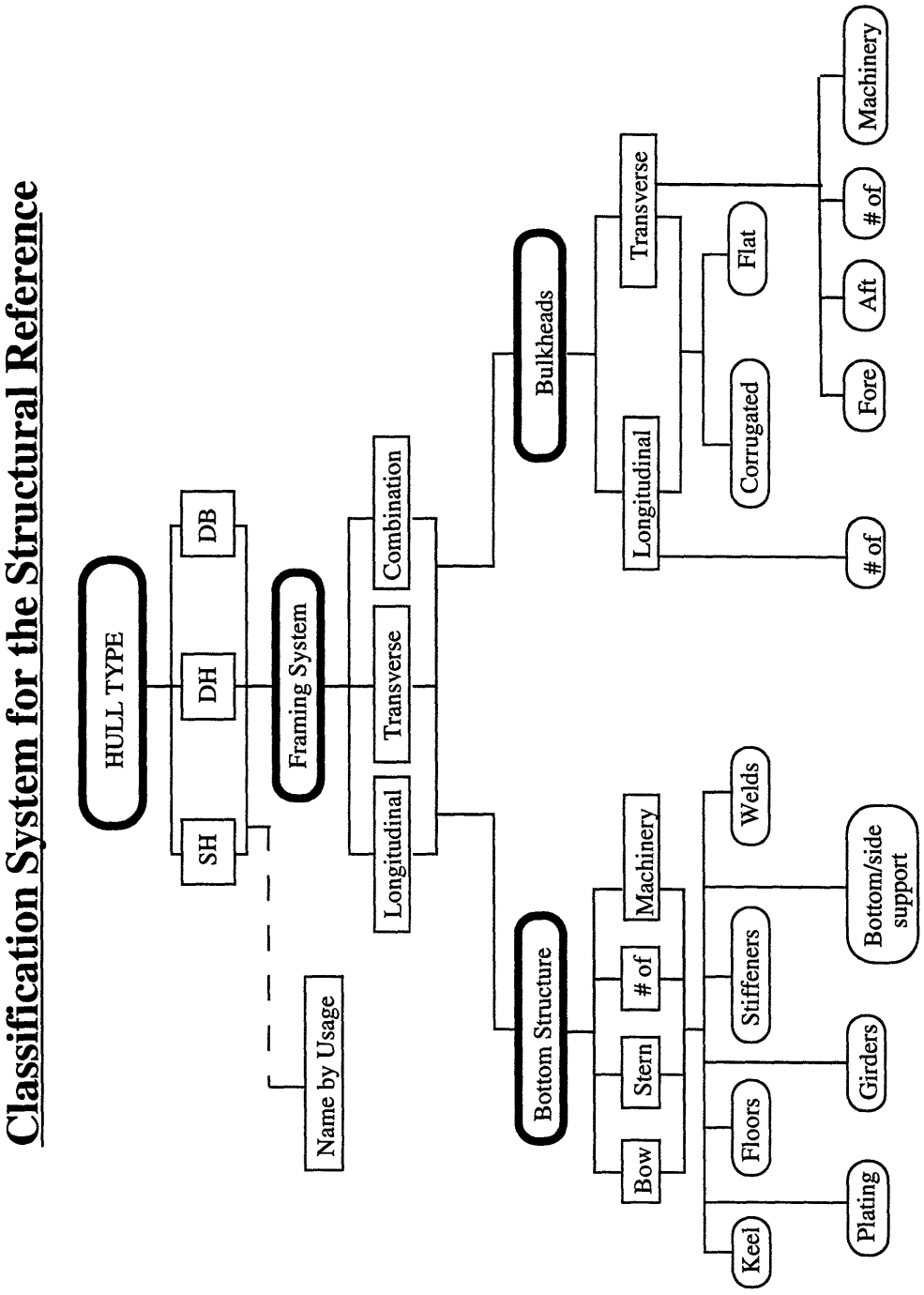

Figure 6.2. This figure represents the organization of the Structural Reference in the Global KBES Design. Its purpose is to conveniently deliver dimensional and geometric information to the Inference Engine if the user cannot, i.e. it provides default values. The strength of this organization is that the user can give general information, and the Structural Reference will determine specific values.

Referring to Figure 6.2, the principal class of Hull Types breaks into single hulled, double hulled, and double bottomed. This list is not all inclusive, but it does provide the most common hulls. After Hull Type, the first sub-class is the ship's Framing System. This includes the primary frame types: Transverse, Longitudinal, and Combination frames. Knowing a ship's framing system indicates the orientation of the primary stiffening structures at the hull bottom. Each Framing System category applies to the Hull Type above it, i.e. they are sub-classes of the primary class.

Bulkheads and Bottom Structures are objects under the Framing System sub class. The objects and sub-objects underneath have value slots that hold the information to completely describe the each of these large objects. The sub-object can be broken further into sub sub-objects depending on the level of detail required for an analysis. Actual values for different ships can be stored in files or directories near the Structural Reference, labeled by Hull Type, Name by Usage, or Framing System. The Inference Engine will pull these values into the object-tree when default values are needed. The default values can be defined as secondary references, so that the user, the primary reference, has the option to provide known values first. If the user provides a value, the defaults will not be used.

The Name by Usage class shown in **Figure 6.2** has been included because it is also a common way to refer to a ship. Usage names (**Table 6.1**) often indicate general properties of a ship's bottom structures, which also allows the Structural Reference to find the values needed for an analysis. For example, if the user does not know the Hull Type but knows it is a VLCC, then the Framing System is likely to be longitudinal.

C-2 Structural Reference Operation

Specifying a hull type characterizes the bottom of a ship in a general manner. The Structural Reference can use this information to find more specific information about a ship. For example, when the user indicates a double bottom ship, the Structural Reference knows to expect the presence of a second floor located a particular distance form the outer hull plate. It also knows that there will be stiffeners running between the to bottoms spaced at fairly regular distances. If the user can also provide the ship's size, ISO standards (international rules) dictate dimensional ranges for these structural components **[Taggart 1980] [Taylor 1985]**. The Structural Reference can choose a value within this range if the user cannot provide the information. Although this process may not send the Inference Engine the same structural dimensions as the ship being investigated by the user, it will give similar dimensions from a ship with same general characteristics. This flexibility

allows the Inference Engine to continue solution procedures when the Structural Reference unit does not have exact information.

In another case, if the user specifies hull dimensions that do not match with the Structural Reference's databanks, estimations can be made. The Structural Reference may choose values of a very similar ship. The user would be flagged by the system if such estimations are required or have been used.

The strength of the Global KBES is that ship hull properties can be accessed in many ways. The user can provide values, or the system can find default values in the Structural Reference. As described above, structural information can be kept in external files, stored by general classes like those in **Figure 6.2.** These files are loaded by the Inference Engine when the user provides general information about a particular ship. Once this information is loaded, the system has access to the dimensions required for structural analysis through the Structural Reference.

C-3 Knowledge Bases Related to the Structural Reference

Grounding analysis methods focus on structural components at different levels. For example, whole ship dynamics requires only general structural detail, such as general hull dimensions, numbers of primary structures (bulkheads, framing, bottom), and general operating conditions (speed, weight). Analysts often "smear" minor structural components into larger structures to account for material volume while simplifying the geometric details. For the Global KBES to accommodate this type of analysis, specific values can be pulled from the Structural Reference. There should be a Smearing Knowledge Base associated with the Structural Reference to receive this information and perform the appropriate level of smearing for a particular analysis. The results are then provided to the Inference Engine and the solution process continues. I have not investigated the Smearing Knowledge Base in great detail, but this feature should not be difficult to create and could be incorporated easily into the Global KBES.

D. External Applications

Large databases and complex computer routines are stored in the External Applications of the Global KBES Design (**Figure 6.1**). **Table 6.2** shows the data and routines expected to be in a Global KBES. The system should have the ability to perform any grounding-related analysis for any shipping community member, so extensive resources are required. Adding all of these Applications to the primary system, in the Knowledge Bases would slow the system's operation because it would have to search many more rules. By keeping these Applications separate, the system remains fast and efficient, accessing this information only when necessary.

Table 6.2. External Applications for the Global KBES

External Data:	Casualty databases (USCG, Lloyd's)
	ISO Rule and Specifications
	Material Property Handbooks
	Research Data
	Design Codes
	Fabrication Techniques
	Company database
	Port Features and Maps
	Ship History Records
External Routines:	Casualty Data Base Search
	Statistical Analysis Routines
	FEM Routines
	CAD Routines
	Ship Dynamic Analysis
	Probability Analysis

Links from the Inference Engine to External Applications must be designed for compatibility. For a few external software applications, Nexpert supplies these links (**Table 3.1**), in the form of internally defined commands. Additional programming in C will allow Nexpert to access unsupported software. To create unsupported links, the designer needs to know how the outside software program functions, so that he or she can access the required information correctly. **[Neuron Data 1994]**

Part III

Implementing the Fillet Weld KBES

Part III describes the Fillet Weld KBES, a working Knowledge Based Expert System created with the Nexpert Object software. This system serves two purposes. First, the Fillet Weld KBES design is similar to the Global KBES for Ship Grounding design presented in Part II. Although I cannot build the Global system by myself, I can use a similar, but smaller, system to assess the Global Design's merit. Second, the Fillet Weld KBES houses a unique collection of welding information collected from the Tanker Safety Project. It stands on its own as a "weld expert" for grounding events.

Chapter 7 describes the Fillet Weld expert system, how it is organized, what it contains, and how it works. Chapter 8 discusses the performance and merit of the Fillet Weld KBES. The chapter also draws parallels between the Global and Fillet expert systems to critique the former's design.

Chapter 7. Example KBES - MIT Welding Research

This chapter presents a working knowledge based system that analyzes fillet weld design, strength, and failure modes. I show how I designed, implemented, and operated this system. I have also listed the contents of my system and commented on its abilities and faults.

Fillet welds are important to grounding analysis, and the affiliated information lends itself well to KBES representation. The Fillet Weld KBES contains information generated by the Tanker Safety Project; this is a reasonable amount of material for this project. I use this system as an independent KBES with a design similar to the Global KBES. It functions as a "Fillet Weld Expert" for grounding conditions. The system can is also a working subset of the Global KBES; its operation demonstrates design features of the Global KBES.

Figure 7.1 shows the components of the Fillet Weld KBES. The basic structure resembles a standard expert system; it includes the User Interface, Inference Engine, multiple Knowledge Bases, and External Applications. This structure is also similar to the Global KBES Design. Parts I and II described the capabilities of these components. The following sections describe the *unique* features and of each component, and how they operate within the Fillet Weld KBES.

KBES Format for Fillet Weld Analysis

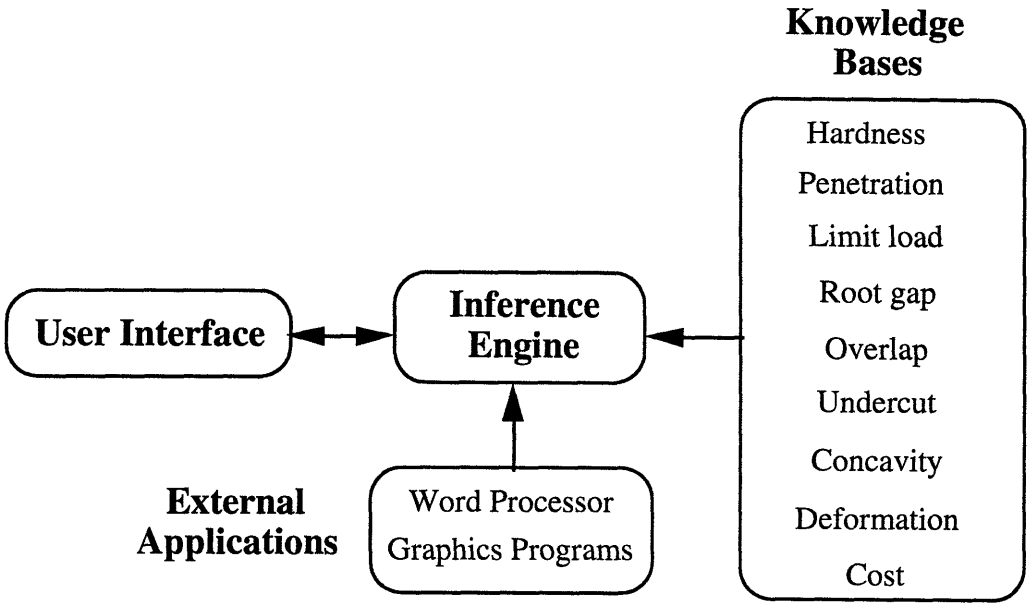

Figure 7.1. This figure shows the structure of the Fillet Weld KBES. The User Interface and Inference Engine operate as standard expert system components. Multiple Knowledge Bases hold rules concerning different aspects of fillet welds that relate to ship grounding. The External Applications for this system are common software applications for editing text and graphics.

A. KBES Design Goals and Constraints

To make development of the Fillet Weld KBES reasonable in the time available, I set two design constraints. The primary constraint is that the system covers fillet welds only, and that it contains results only from the Tanker Safety Project. Exploring other weld topics would make the system an unmanageable size. The second constraint is that the system be designed for a non-expert user who is somewhat familiar with welding, ship structures, and grounding events. For instance, a user might be a managing engineer who oversees many operations, but who is not intimately involved in any one area.

The design goals for the Fillet Weld KBES are similar to those of the Global KBES for Ship Grounding. These include:

- A user-friendly computer system that provides useful fillet weld grounding information and suggestions.
- A design that is easy to expand as new topics arise.
- A system that explains and justifies its actions and conclusions.
- A system that indicates research gaps and missing information.

B. Fillet Weld KBES - Contents

The following section describes the contents of the Fillet Weld KBES, as well as current progress in Knowledge Base development. Although the system is not yet incomplete, the extent of topics to be covered in the future has been well defined.

B-1 System Contents - General

Table 7.1 lists the current status of topics in the Fillet Weld KBES. Topics followed by a "**C**" have been completely integrated into the KBES, and the Tanker Safety Project will not generate additional information on these issues. Categories labeled with a "**P**" are partially covered by the system because this research is still in progress. These topics will be completed as the research results become available. Topics listed with an "**N**" are not yet included because they are new research projects. They will be incorporated into the KBES system in the future.

Table 7.1. Topics Covered by Fillet KBES

Category	Status
Weld Strength Analysis	
Weld Theory	**P**
Bending Tests	**C**
Lazy-L Tests	**N**
Penetration Tests	**N**
Defect Analysis	
Rootgap	**C**
Overlap	**C**
Undercut	**C**
Concave/Convex	**C**
Weld Improvement Suggestions	
Size	**P**
Penetration	**P**
Weld Geometry	**P**
Cost of Change	**C**
Number of Passes	**P**
Weld Material Analysis	
Hardness	**P**
Electrode	**P**
Heat Affected Zone	**P**
Joint Distortion Models	
Critical Loads	**N**
Critical Displacement	**N**
Deformation Sequences	**N**
Effects of Different Loads	**N**
Weld Technique Selection	
Process Choice	**N**
Electrode Choice	**N**
Standards vs Research Analysis	**N**

Table 7.1. This table lists the topics for the final version of the Fillet Weld KBES. The topics are at different stages of completion because some of the research projects are still in progress. The following codes describe each topic's current status:

C The KBES has complete information from Tanker Project in this category.
P The KBES has partial information in this category; ongoing research.
N The KBES has no information in this category; ongoing or future research.

Table 7.2 lists the topics required to fully represent results from weld studies from the Tanker Safety Project. This outline is also an initial proposal for the Welding Handbook outline, a Tanker Safety Project deliverable mentioned in Chapter 1 **[McClintock 1995] [Masubuchi 1995].** Categorizing the research results in this form is difficult, because many topics are inter-related. For example, the topics listed in Section 3 of the outline, *General Considerations*, pertain to and must be described for each weld type in section 4, *Weld Behavior for Different Loads*. This complexity indicates the interdependence of fillet weld topics and grounding issues. Creating the Weld Handbook is a substantial organizational task, which will inevitably include extensive indexing, referencing, and cross-referencing.

The primary strength of the Fillet Weld KBES is that it can cover the same topics as the Weld Handbook with substantially fewer organizational problems. In a KBES version, each topic listed in **Table 7.2** would reside in a separate Knowledge Base, and the rules of each Knowledge Base would codify the relationships between topics. Because Knowledge Bases are not confined to a physical order like a book, cross referencing is less cumbersome.

In this system, the User Interface provides indexing services, allowing the user to select an analysis to perform. In essence, this organization lets each analysis access Knowledge Base topics in whatever manner is most convenient. When a different question is posed to the system, the data in the separate Knowledge Bases can be accessed in a different way. This process is similar to a handbook where the reader removes and replaces sections as they reference it for different purposes.

Table 7.2. Initial Outline of the Welding Handbook

1. **Introduction**
   - Failure Examples
   - Current Practices
   - Current Rules and Specifications

2. **Theory and Analytical Methods**
   - Theory
   - Analytical Methods
   - Examples

3. **General Considerations**
   - Material Strength
   - Weld Geometry
   - Joint Design
   - Weld Defects
   - Deformation
   - Environmental Effects
   - Expected external loads

4. **Weld Behavior for Different Loads**
   - T-Joints
     - Longitudinal Shear Tests
     - Transverse Shear Tests
     - Web Tension
     - Web Folding
     - Hardpoint Fracture
   - Lap Joints
   - Cruciforms
   - Butt Welds

5. **Suggestions for New Practice**
   - Lazy-T
   - Beam Bending

6. **Relative Effects of Change**
   - Cost

7. **Related Topics**

Table 7.2 is an early proposal for the Welding Handbook, one of the Tanker Safety deliverables. Like the Fillet Weld KBES, the handbook will cover welding information generated by the Tanker Safety Project. Unlike the KBES, organization on paper is difficult, as most issues listed above have complex relationships.

B-2 Specific Knowledge Base Contents

Every issue related to fillet weld design, performance, or analysis is a separate Knowledge Base, as shown in **Figure 7.1**. Keeping each topic separate is a feature of the Global KBES Design. This structure allows new Knowledge Bases to be readily implemented into the overall system without disturbing the current system's organization and operation.

Table 7.3 lists Knowledge Bases of the Fillet Weld KBES and indicates the types of analysis supported by each Knowledge Base. The italicized portions of the table are topics which will be implemented soon. For a listing of current Knowledge Base rules, hypotheses, and objects, please refer to **Appendix I**.

Table 7.3. Knowledge Bases and Analysis Options for the Fillet Weld KBES

Topic	Knowledge Bases	Analysis Options
Weld Defects **[McDonald 1993]**	Root gap Undercut Overlap Concave/Convex	Fillet weld improvement Current practice vs research Effects of defects on strength
Limit Loads **[McClintock 1994]** **[Kirkov 1994]** **[McDonald 1993]**	Limit Load	Web vs weld failure Limit load calculation Tearing resistance Critical displacement Weld peeling failure Joint deformation failure Weld strength improvements Geometry analysis
Cost **[Koga 1993]** **[Kirkov 1994]**	Cost	cost of size change cost of welding technique *cost of penetration change* *cost of geometry change*
Material **[Middaugh 1994]**	Weld Metal *Base Metal*	weld homogeneity weld strength single vs multi pass over vs under match weld HAZ strength
Penetration	Penetration	weld strength improvement
Real-life Examples **[McDonald 1993]** **[Sinmao 1994]** **[Xiauo's 1994]**	*Failure Examples*	*ship failures* *weld failures* *grounding reports*
Joint Deformation **[Atmadja 1994]**	*Deformation*	*joint/weld interaction* *deformation sequence* *critical displacement* *failure loads*

Table 7.3: Knowledge Bases for the Fillet Weld KBES and the analysis options they provide. General topics and information sources appear at the top of each section. The items listed in italics are not yet incorporated.

C. Fillet Weld KBES - Operation

This section demonstrates how each component of the Fillet Weld KBES operates. Since I cannot cover all system actions, I have chosen examples to help the reader understand and appreciate the abilities of the system.

C-1 User Interface Operation

Careful consideration went into designing the User Interface for the Fillet Weld KBES. As mentioned earlier, the User Interface controls interaction between the user and the system. Clear and concise computer screens that reflect the contents of the system and indicate actions expected of the user are essential for this KBES; it is designed to be a user-friendly system for non-experts. **Figure 7.2** shows the first operating screen, Screen 1, that the User Interface displays **[Liang 1994]**. Screen 1 appears before the Inference Engine initiates a search and gives the user analysis options. The contents of this screen reflect the topics included in this system, and the menu-driven environment is self explanatory.

The *Index* lets the user search for specific topics, using a search domain to speed the solution process. To use the *Index*, however, the user must know exactly what topics are relevant to their query. When the user is unsure what issues are critical, or if they want a more general analysis, the *General Analysis* options provide alternatives. The *General Analysis* inquires take longer, but they automatically find and extract the correct analytical tools when the user does not know which are appropriate.

The User Interface has an open design; it allows the developer to add and subtract options when necessary without disturbing the existing structure. For example, the designer would add a new, discrete body of information about a specific weld defect to the system as an *Index* entry. It would be another separate Knowledge Base, to be called by the Inference Engine if the user wants to know about that weld defect. In another instance, the developer would add new relationships between weld defects (between multiple Knowledge Bases) as a new *General Analysis* entry. When the user chooses this new entry, the Inference Engine calls the Knowledge Bases that the developer specified for that analysis. By keeping the Knowledge Bases separate and simple, they remain available for all *General Analysis* entries. This eliminates repetition when constructing the system. The

next section describes the interaction between the Inference Engineer and the User Interface when a user selects entries from the Index and General Analysis menus.

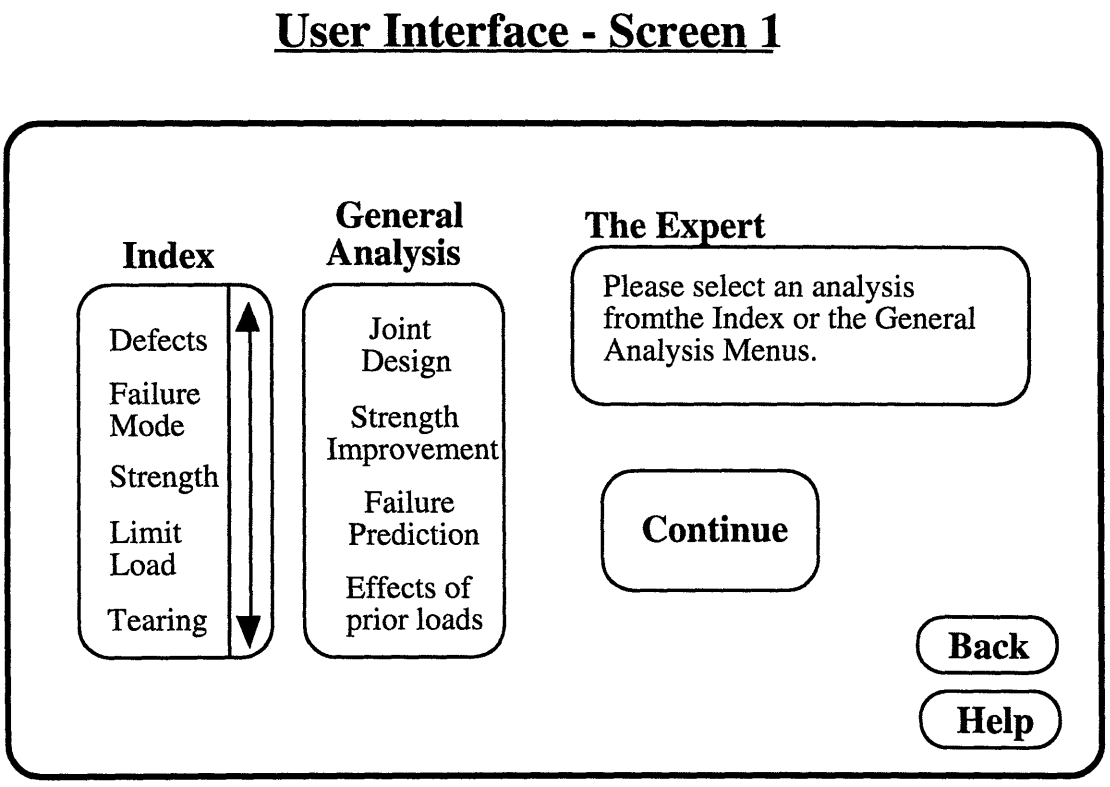

Figure 7.2. Screen 1 of the fillet weld KBES provides the user with general options at the beginning of a session. The *Index* lists each individual analysis option. The *General Analysis* menu provides more general options for users who are unsure which analysis to perform.

Figure 7.3 shows a standard operating screen during operation. *The Expert* box interacts with the user. It requests information and instructs the user during the solution process. Prompts from *The Expert* usually include requests for value inputs and prompts for analysis selections. The *Technical Support* box assists *The Expert* by clarifying and expanding on requests from the Inference Engine to the user. These messages usually take the form of pictures, references, and textual explanations. After each user response, these screens change. This particular feature of the User Interface increases the user-friendliness of the entire system.

The *Your Response* box accepts information from the user responding to Inference Engine requests. This box is designed to regulate input formats to those that are acceptable to the Inference engine. These formats include using True-False-Unknown questions,

numeric entries, or lists of possible answers. With this input design, the system minimizes errors and improperly formatted entries. The *Help*, *Back*, *Why*, and *Logic Tree* buttons are alternative user responses to question asked by the system. They produce pop-up windows, which provide additional information. The *Help* button offers suggested responses to the immediate question. The *Why* button provides explanations for current questions, so that the user knows in which direction the solution process is heading. Often, this elucidates the immediate question. The *Logic Tree* is a graphical version of the *Why* option; it shows a picture of the current rule being analyzed by the system. This feature allows the user to view the conditions and hypotheses as they are investigated. All of these function have been included for the purpose of making the system more thorough and easy to use.

User Interface - Typical Screen

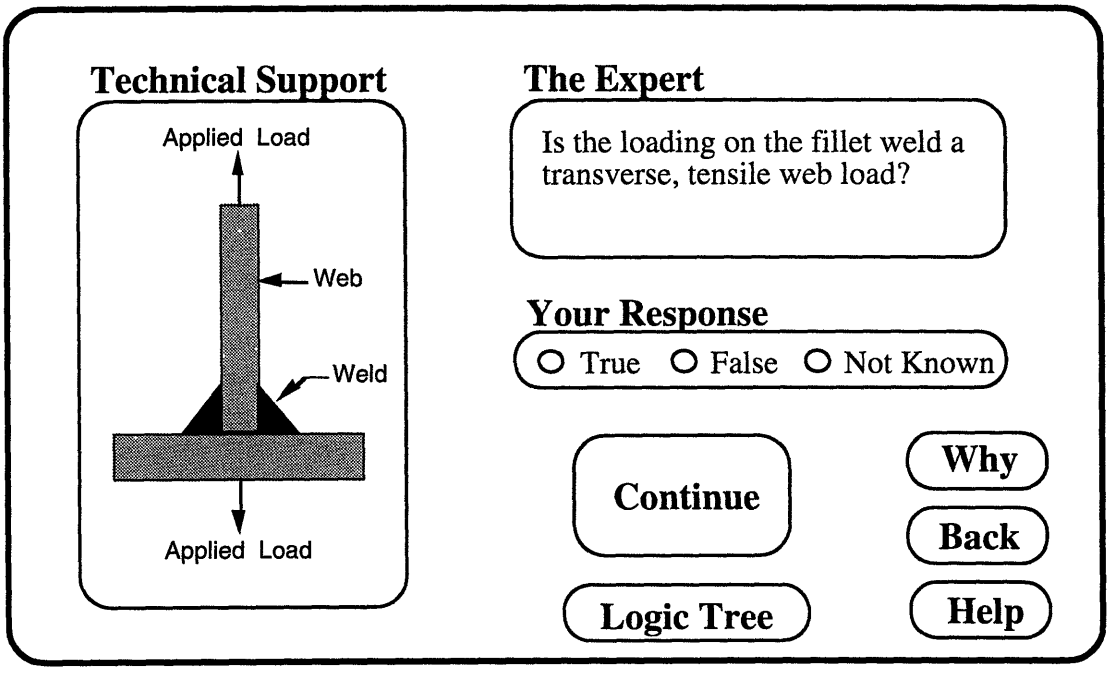

Figure 7.3. This figure shows a standard screen displayed by the User Interface. The *Technical Support* box shows pictures and text that clarify system procedures or questions; *The Expert* box prompts the user for information; and the user responds in the *Your Response* box. Buttons in the lower right corner provide additional information upon request.

C-2 Inference Engine Operation

The Inference Engine conducts the solution processing. It loads Knowledge Bases, searches through them for analysis related rules, evaluates rules, and requests information. The following paragraphs use examples to demonstrate these actions.

As an example, **Figure 7.4** shows Screen 2, displayed when the user selects *Defects* from the *Index* of Screen 1 (**Figure 7.2**). Screen 2 offers the user more specific options, to minimize the number of Knowledge Bases required for addressing the user's question. This makes the system more efficient because the Inference Engine searches through fewer rules. If the user selects *Rootgap* from the available options, the Inference Engine loads the Rootgap Knowledge Base containing rules related to weld rootgap defects. The Inference Engine now has the tools it needs to investigate this topic. The Inference Engine also chooses a possible conclusion, such as "The rootgap size is too large", and starts the search process to determine if this conclusion is true or false.

With a suggested conclusion or hypothesis, the Inference Engine uses backward chaining to test its conditions. If the conditions are hypotheses of other rules, these must also be evaluated. The Inference Engine combines known and determined values with forward chaining to find related rules which are also investigated. The are several outcomes of this search. First, the user receives an answer to the initially proposed solution (True or False). Second, the user receives a list of suggested actions for weld improvements. These suggestions are attached to a rule's hypothesis, and appear when the hypothesis is true (AndDo). Third, the user gets the values of related conclusions which may or may not be useful, but are true for the given set of circumstances.

Inference Engine Action - Screen 2

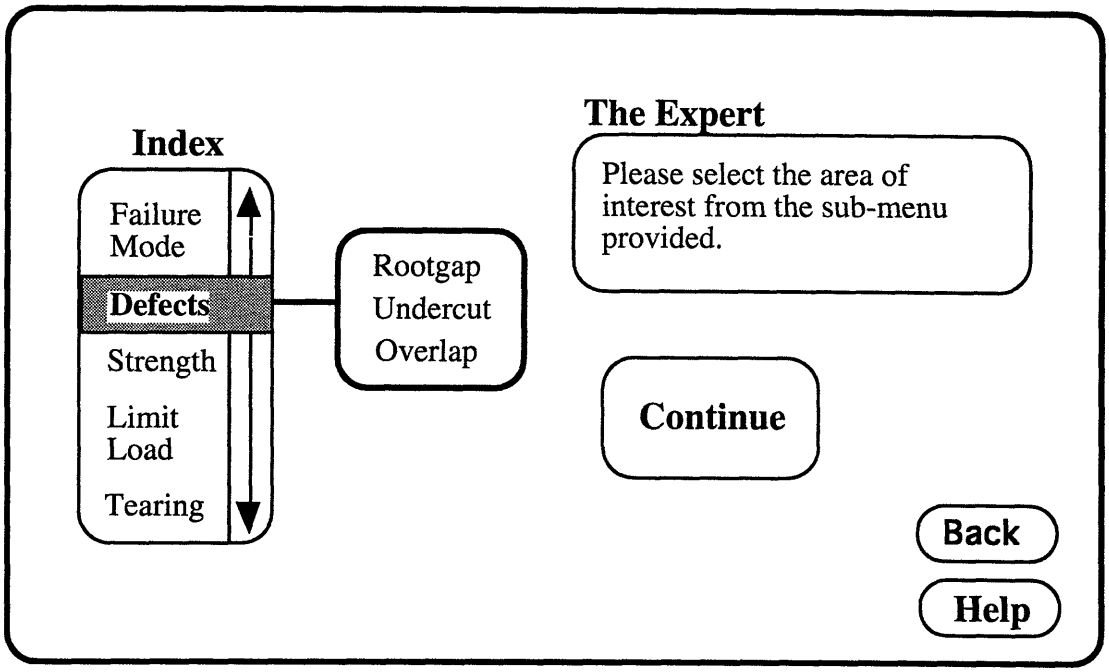

Figure 7.4. The Inference Engine attempts to narrow its search domain by prompting the user to be specific. Above, the user has chosen "Defects" which produces another list of options. If the user then selects "Rootgap", the Inference Engine loads the Rootgap Knowledge Base and starts the solution process.

C-3 Knowledge Base Operation

The Knowledge Bases hold the rules necessary for KBES analysis. These Knowledge Bases are loaded by the Inference Engine when necessary. The Fillet Weld KBES has a specific rule organization scheme. Each Knowledge Base contains rules for a very discrete body of information. These bases have a very loose organization by general category, like Strength analyses, weld defects or deformation models. The advantage of separate and discrete Knowledge Bases is that this organization speeds up solution processing and reduces logic conflicts. I have tried to minimize the number of rules within each Knowledge Base.

As an example, the Rootgap Knowledge Base mentioned in the last section, contains only three primary conclusions:

- The rootgap size is inadequate.
- The rootgap size is adequate.
- The rootgap size is borderline.

The suggested actions generated by the Inference Engine, (see section C-2), are each attached to only one conclusion. The conclusions use the same information from the user, but they are mutually exclusive; only one of the three will be true at the end of a solution process. The advantages of such an organization are that the user provides information only once (instead of three times, once for each conclusion), the system will not reach conflicting conclusions, and all information pertaining to rootgaps resides in one location.

The size of a Knowledge Base depends on the amount of information available for a particular topic and how logically it fits together. For Knowledge Bases that require rules and routines from other Knowledge Bases, these links must be established by the designer within the former Knowledge Base. When two Knowledge Bases are loaded, special care is required to ensure a common terminology. The system slows when multiple Knowledge Bases are loaded because more rules need investigation. The solution to this problem is to use a clearing, or unloading, function to remove Knowledge Bases when they are no longer required for a solution process.

Selections from the *General Analysis* list (**Figure 7.2**) often require several Knowledge Bases. An example of this might be an investigation of weld defects. All defect related Knowledge Bases will require investigation. The most efficient way to do this is to test each defect base one at a time. The Rootgap Knowledge Base will be loaded, tested, and unloaded before calling the Overlap Knowledge Base.

There are three kinds of conflicts to watch for in Knowledge Base development. The first is logic inconsistencies inside a Knowledge Base. This problem would have resulted above if the Rootgap analysis said that the rootgap was adequate and inadequate at the same time. As shown above, this problem is avoidable with careful Knowledge Base design methods. The second type of conflict is less avoidable. This results when two information sources give different results for an identical set of conditions. It is expected at times because experts often disagree. This conflict is not necessarily bad if the user has the background to choose between to conclusions. If not, the designer should eliminate one of the sources, or make sure they do not appear at the same time **[Ghosh and Kalyanaraman 1993]**.

C-4 External Source Referencing

To keep the system fast and efficient, external data for the Fillet Weld KBES is stored in outside files in the same manner as the Knowledge Bases. The external files for the Fillet Weld KBES include text and graphics files. These appear in *Technical Support*, *Why*, and *Help* windows (**Figure 7.2**) when specific events occur during the solution process. **Figure 7.3** shows one such external picture in its *Technical Support* window. This particular illustration is for clarifying *The Expert's* request to the user. Other external sources hold research data, photographs of weld failures, photographs of ship grounding failures, casualty database information, and fracture theory.

Chapter 8. Results

This chapter reviews the performance of the Fillet Weld KBES. I discuss how the system meets most of its design goals, examples showing how these goals have been met, merits of the open-design and design techniques, and areas for improvement.

A. Fillet Weld KBES

With respect to the initial goals, the Fillet Weld KBES performs satisfactorily; it delivers a unique combination of welding information to non-experts in a user-friendly manner. The system includes a information in a variety of formats, such as pictures, weld failure photographs, graphs, data tables, equations, and text. The user receives this information in a useful manner because the systems delivers conclusions related to the initial inquiry. The system also offers suggestions for better welding practices and related conclusions for further exploration, which make it a practical tool for weld improvement.

A-1 Design Approach Results

In the initial stages of design, I used the Top-Down approach and determined the users of the system and the questions to be answered. It helped me design the User Interface *General Analysis* box (**Figure 7.2**). The analysis methods offered in this menu represent the primary concerns of shipping community members regarding weld failure in grounding conditions. I gathered and organized the information required to address these concerns, as is common in the Top-Down approach. For later stages of development, I used the Bottom-Up approach to organize the research data generated by the Tanker Safety Project. All weld topics from this project are intended for implementation in the Fillet Weld KBES as *Index* entries (**Figure 7.2**). Although many of the relationships have not been realized between this newly generated research, these entries provide the tools for future relationships and analysis methods. This is a common result of a Bottom-Up approach.

A-2 User Interface Performance

As part of the user-friendly design, the User Interface includes explanation facilities and justification options. The operation of each of these was described in Chapter 7. At each step in the solution process, the user has a variety of clarification options to guide,

explain, and show what information the Inference Engine needs and why it asks for it in a particular way. Not only does this information help the user answer the questions, it offers references to outside resources, delivers in-depth descriptions of theories used for an analysis, and provides research data so that the user need not consult outside sources. These features make the system easier to use and more informative.

A-3 Open Design Performance

The open design of the Fillet Weld KBES's Knowledge Bases and User Interface has proven very useful. The system is exceptionally easy to expand and change. Because the Tanker Safety Project is constantly generating new information and discovering new relationships between welding topics, a flexible design is important for this particular KBES. Separate and distinct Knowledge Bases reduce consistency problems and expand the number of analysis tools available to the Inference Engine. The appearance of Screens 1 and 2 (**Figures 7.2 and 7.4**) in the User Interface reflects the system's open design and the approaches used to form it. Categories are easily incorporated into both the *Index* and the *General Analysis* menus

A-4 Research and Organizational Assistance

Preliminary tests demonstrate that the Fillet Weld KBES functions as a useful tool for both organization and research . For organization, the system offers flexibility for topics that have complex relationships. The user and the Inference Engine have easy access to all topics. For example, the KBES has been far easier to organize than the Welding Handbook, which covers the same information and serves the same purpose. For the handbook, organizational problems stem from the interdepedence weld topics **[McClintock 1995].** The problem with a handbook is that it is confined to a physical order, which means the user has to sort through sections to collect the relevant data for an analysis. The KBES does this searching and organizing internally. The major drawback to the KBES is that it is a computer program, and its errors may not be as transparent as those occurring in a textual handbook.

As a research tool, the Fillet Weld KBES helps organize new information and find new relationships between data. It also locates research areas that need more work, which is particularly important when research must be applied to real-life occurrences. For the Fillet KBES, the largest disparities I found typically occurred between theoretical calculations and experimental results, and between experimental assumptions and real-life conditions, and between research results and common practice.

For example, tear resistance calculations based on fracture theory do not match those measured in laboratory experiments involving fillet welds exposed to web tension **[Kirkov 1994]**. The former predictions were much higher than the latter. The KBES explains this fact and suggests, for safety, using empirical methods and data for determining tear resistance. It also suggests a test method for this type of information and indicates references with more details on this method.

Another common conflict occurs between research assumptions and real-life conditions. Most research projects encounter this problem because researchers often use simplifications to simulate real-life occurrences in a laboratory. Simplifications are also used when the results need to be explained analytically. For example, when researchers produced an analytical method for calculating a weld's limit load (under web tension), several assumption were made:

- a homogeneous weld material,
- purely plastic deformation within the weld region, and
- non-hardening weld metal behavior during plastic deformation.

[McClintock 1994]

The first assumption was experimentally verified **[Middaugh 1994],** and the supporting data is presented by the KBES if the user requests it. The second assumption was addressed by manipulating the weld's geometry until this condition was true. It is uncertain if this mode of failure can be expected in a real grounding incident. The third assumption simplifies the problem to a reasonable research task. Although this assumption has been satisfied in laboratory tests, there is little information showing how hardening would alter the accuracy of the analytical solution. The KBES can point to these last two caveats so that the user knows the assumptions and possible limitations of the answer. This knowledge is especially important when exact solutions are unavailable and approximations and assumptions are necessary.

B. Merit of the Global KBES Design

Creating the Fillet Weld KBES helped evaluate the Global KBES Design presented in Part II. As discussed in the previous section, the open design proved its merit in the Fillet Weld KBES by allowing easy system expansion, efficient operation, and reduced

system conflicts. For the Global KBES, these abilities are critical, because there is a great deal of grounding related information that could be incorporated into the system, and different users will want to access different portions of it.

B-1 User Interface Design

The User Interface design in the Fillet Weld KBES showed the importance of a clear, concise design. The screens it presents must reflect the contents of the system. The Global KBES design involves multiple User Interfaces for different types of users with different questions. A User Interface for a designer, for example, would include an *Index* box of design related topics and a *General Analysis* box with design methods. Each group of users would be responsible for specifying requirements for their own User Interface because they are most familiar with their own needs.

B-2 External Applications Referencing

The Fillet Weld KBES shows a simple operation of External Applications. The text and graphics files of the Global KBES would operate in a similar manner. Building the Global KBES, however, would require more extensive external resources. To include outside databases, the designer must match formats between the KBES's request and the database's search language. This ability requires additional programming and an in depth understanding of the database's structure. The same procedure is required for mathematical routines and other software. I did not include this type of External Application on the Fillet Weld KBES.

B-3 Knowledge Base Design

The Global KBES would have many more Knowledge Bases than the Fillet Weld KBES. This organization, however, should be similar, because the Fillet Weld KBES works well by keeping each Knowledge Base as discrete files. This organization is one of the Global Design's strengths. The large number of Knowledge Bases in the Global KBES would, however, require grouping Knowledge Bases with similar topics. For example, all weld defect Knowledge Bases could reside in one directory, while Structural Reference information would be placed in another. The User Interface and Knowledge Base rules would be used to locate different files within their respective directories.

B-4 Structural Reference

I did not implement a Structural Reference in the Global KBES. This system could use a Weld Reference which gives default values for fillet welds used in different ship structures. There are some default values in the Fillet Weld KBES, but they are not as organized or as versatile as the Global System's Structural Reference. This is an important component of the Global system, as well as the Fillet Weld KBES. It should be addressed more thoroughly. Another new study has also investigated the use of Knowledge Based Expert Systems for ship hull subdivision and reference **[Sen and Gerigk 1994]**. It supports the use of expert systems for such characterizations.

The Global KBES for Ship Grounding would require a major development effort; the smaller Fillet Weld KBES represents almost one man-year of work. The time required to complete development may be the biggest obstacle in implementing an expert system. Making the Global KBES useful would also require the cooperation of the shipping community, which is another potentially serious obstacle.

Chapter 9. Conclusions

In this thesis, I have presented two investigations of Knowledge Based Expert System applications: the Global KBES for Ship Grounding and the Fillet Weld KBES. The combination of design and implementation provides a thorough look at the strengths and weaknesses of expert systems with respect to ship grounding analysis.

A. The Value of Knowledge Based Expert Systems

Part I comprehensively covers important issues in the expert systems field. These chapters give a short history of expert systems and describe how recent advances in developmental tools and design methods have expanded their use in technical fields. Part I outlines who uses expert systems, when they use them, what these systems will do, and how they work. This information can help an expert system designer determine if a KBES is appropriate for his or her project.

After determining if a topic is suitable for KBES representation, choosing an appropriate development tools critical. This choice will significantly influence the final form of the system and its performance. A commercial software package, which provides the designer with a logic shell and predefined development tools, reduces development time, because the designer does not have to develop the tools or debug the system. Recently, commercial packages have become more dependable, flexible, and suitable for technical tasks. The other alternative is custom designing a system. This option maximizes system flexibility and efficiency because the developer includes only those functions, search procedures, and data sources relevant to the topic. The disadvantage of this method is the time and expertise required to customize a system.

For my project, I chose a commercial logic shell because minimizing development time was critical. Chapter 3 discussed properties of Nexpert Object, the logic shell I used for this project. I chose this package because it offers both rule-based and object-oriented knowledge representation. This capability is useful for classifying research knowledge and defining ship hull components. The combination of forward and backward chaining optimizes search procedures, and the designer has complete control over how the system tests its rules. Other Nexpert Object features, such as the rule and object editors, ease development. The explanation facilities (How, Why, and Current Rule) make the system more understandable to both the designer and users; these functions show how the system

analyzes and interprets its information. This information makes conclusions from the system more believable.

B. Using a KBES to Study Ship Grounding

There are distinct advantages to both having and developing a KBES for ship grounding. Compared to a standard database, the KBES offers flexibility and knowledge representation. Both of these qualities are important for analyzing ship grounding events. System flexibility allows using the results of multiple grounding investigations, each of which may have its own result format. Grounding analysis is a new field, and researchers have different approaches to its characterization. Knowledge representation is critical because it allows the system to include experience and empirical information which are heavily used in the shipping industry.

Having a KBES for ship grounding is like having an expert around for everyday questions; the system can answer the same questions a human expert could. If the KBES is used often, its value will offset development costs by saving the cost of enlisting human experts. The KBES also collects and saves knowledge and experience that may otherwise be lost over time. For example, the Tanker Safety Project has gathered a great deal of grounding information that supplements its research. When the project is complete, the research results will be written down, but the supporting information may not be included. Preserving this knowledge is also facilitated because a KBES offers more flexible organization than text.

The process of developing a KBES is useful because it organizes bodies of information and reveals knowledge gaps or inconsistencies. I have constructed the Global KBES Design and the Fillet Weld KBES to address particular grounding issues. To accomplish this, I had to collect and organize the necessary information (with the help of experts) in a manner that best addressed the problems outlined in previous sections. As I showed with the Fillet Weld KBES, this process pointed to missing information and possible avenues for future research.

C. The Future of Expert Systems

Exert systems are relatively new in technical fields. Their popularity in these areas has been steadily increasing over the past ten years; my literature survey indicates dozens of recent proposals for new knowledge based expert systems. This expansion is the result of improved development tools and methods. The field of knowledge based systems has matured beyond being experimental computer science. KBES's now offer a practical way to combine research results with applications.

The appearance of new tools for developing expert systems indicates a better understanding of general KBES design. The first successful expert systems were custom designed, and the developers had little previous work for guidance. They had to create their own design tools and development procedures. As more systems developed, the most successful implementations served as guides for subsequent KBES development. In some cases a logic shell from an early system was used to develop a newer system. The developers removed the Knowledge Base from the logic shell, and replace it with a Knowledge Base for the new project.

As developers noticed commonalities between systems and their development, the first design methodologies and commercial packages appeared. These tools allowed more options and faster development times for KBES designers. The Nexpert Object software represents some of these tools, such as:

- the ability to control object and rule representation,
- the inclusion of numeric capabilities,
- easy links to outside data,
- automatic editing facilities, and
- explanation facilities.

New development methods also indicate a better understanding of human knowledge and how to simulate human thinking processes on a computer. These methods include improved knowledge representation, like the combined rule and object representation in Nexpert Object. Search methods combinations, such as forward and backward chaining, also simulate human problem solving processes. I suspect these methods will continue to improve because more systems are being developed, which will provide more feedback.

D. The Ship Grounding KBES Design

The Global KBES Design is adequate for address all principal ship grounding issues. I developed the design using a Top-Down approach (Chapter 4), beginning with identifying potential users and their concerns. I also identified the primary sources of information for this Global KBES, such as casualty data bases, research projects, analysis methods, ship hull descriptions, and international regulations (Chapter 6). To incorporate all of this information into one system, the design had to remain broad and open-ended. Because the system has independent components, the design readily accepts new information and changes without adversely affecting the existing system.

As mentioned early in then paper, the all-encompassing KBES would be too broad to be useful (Chapter 4). To limit the Global KBES and make it practical to particular users, the appropriate User Interface must be developed. These Interfaces would correspond to the members of the shipping community (Chapter 5). Each Interface would only access that information from the Global KBES that directly relates to the user's interests.

Other features play equally important roles. The Ship Structural Reference provides structural dimensions to use as default values when the user cannot supply detailed information. This means that if the user provides general information like ship type and vessel size, the Structural reference can provide information about stiffener spacings and weld sizes. The External Applications allows the system to connect with multiple outside sources. Casualty databases are a good example of this capability. The Global KBES can provide access to U.S. Coast Guard grounding incident reports, without incorporating this information into the Knowledge Base. External Applications keep the system's speed at a maximum.

Part II also covered the logistics required for constructing the Global KBES. Overall, the system must be flexible enough to address all potential user and all types of information and software routines, but specific enough to cover these topics in detail. The KBES should be compatible across many platforms, including personal computers, workstations, and mainframes. The system would be built using C, or another portable language, so that most software applications are accessible. For links to external databases, the designer know the database's query language, so that the KBES can request and receive information in the proper format. Knowledge Bases would be stored in a loose organization of individual topics. By keeping them separate, it is convenient to add more topics and access one set of rules without loading unimportant ones or upsetting the existing system's organization. Links to Knowledge Bases can be generated from user

choices on the User Interface, Inference Engine calls, and Knowledge Base rules. Links to the Structural Reference and External Applications are produced by Inference Engine procedures and Knowledge Base rules.

Because building the complete Global KBES is beyond the scope of this project, I demonstrated the important concepts from the Global KBES Design by developing and operating the Fillet Weld KBES. This implementation shows the strength of the open design and the Global system's organization.

E. The Fillet Weld KBES

I designed and constructed the Fillet Weld KBES and demonstrated that it performs well. The success of its open design supports the design of the Global KBES for Ship Grounding. Using Tanker Safety Project weld information, I created a user-friendly computer expert for fillet welds exposed to grounding loads. The system provides a unique collection of information, including:

- strength analysis,
- design analysis,
- failure mode predictions,
- cost analysis,
- deformation models, and
- defect analysis.

A primary benefit of developing the Fillet Weld KBES is the organization it provides for welding topics with complex relationships. This advantage became apparent when we tried to create an outline for a welding handbook. The Fillet Weld KBES provides a convenient index for topics, and chooses the most convenient topic organization for a given analysis. It can exploit the interdependence of different topics, while hiding this complexity from the user.

Other advantages of this system include those listed at the beginning of this chapter. The explanation facility is an important development and operation tool; it reveals errors in the Knowledge Base, and provides helpful information to system users. References and explanations delivered to the user during operation allow the user to analyze the system's methods and conclusions.

The User Interface is critical for providing easy operation of the system. The *Index* and *General Analysis* menus provide options for both experts and non-experts. Both are hierarchical menu-driven windows, which make the system's organization easier to understand.

F. Future Work

Since I am the principal researcher of KBES for grounding investigation and because the Tanker Safety project is nearing an end, I suspect this project will not continue. I have one more month of full time development work for the Fillet Weld KBES. This system can be completed to a reasonable stage, and added to in the future if desired. If the project were to continue, I suggest the following topics for new research and development.

F-1 The Ship Grounding KBES

The Global Design is unlikely to develop past the design stages because of a lack of interest. The shipping community is understandably hesitant to invest the time and money into a new organizational tool like this system. The timeline of this project depends on the level of sophistication desired for the final version. A product of commercial value takes nearly five years to perfect, field versions can take up to five years, and a complete research model can take at least several years **[Waterman 1986]**. Developing this system would also require an amazing amount of cooperation between members of the international shipping community. This cooperation is also unlikely, because company information is often secret. Parts of the Global system could be developed, however.

I suggest more work on the Structural Reference unit because this component is relevant to almost all shipping issues. A convenient computer driven ship structure reference could be very useful for present and future hull calculations. Also, future exploration into the Global KBES development should include exploring links to external sources and software applications. I did not cover this area thoroughly. Interactions between multiply loaded Knowledge Bases should also be investigated further. I did cover design methods related to this idea, but did not rigorously test the Fillet Weld KBES to determine if the methods work all the time.

For implementation, the designers would need to create a User Interface customized for particular members of the shipping community. This would involve working with

members within each group to determine their concerns regarding ship grounding events. System development should also include .

If the Ship Grounding Expert System is implemented, despite these obstacles, it offers these benefits:

- The Global KBES has the potential to cover the ship grounding concerns of all shipping community members. The global design and the system components are broad and expansive.
- The User Interface accesses only those components that the user needs. It controls the query process, and should be developed by the system's users, or a representative from a user group
- The Structural Reference Unit is a critical component. Ship hull components provide the common link between all forms of grounding analysis. They also provide a common language among shipping community members. Because hull components are the system's common ground, designing this unit with the proper international terminology and regulations is critical.
- The system's ability to link with external software is ideal for ship grounding analysis. Company databases and software tools can be incorporated into the global system, but stored outside the User Interface. This organization increases the system's total resources without reducing the User Interface's efficiency.

F-2 The Fillet Weld KBES

There will be more work on the Fillet Weld KBES. The incomplete sections mentioned in Chapter 7 should be completed in one more month. As research on other aspects of ship grounding finishes, the results of these studies will also be added to the system. This on-going KBES development will help the Tanker Safety Project members organize their information for the Welding Handbook. Future work may also include developing the Welded Structure Reference, similar to the Ship Structural Reference from the Global KBES, to provide default values for weld parameters.

Bibliography

Adeli, H., Expert Systems in Construction and Structural Engineering, Chapman and Hall, New York, NY, 1988.

Agapakis, J.E., and Masubuchi, K., "Expert Systems in Welding Fabrication: an Overview and a Prototype", OMAE, pp.139-146, 1988.

Arockiasamy, M., Expert Systems: Applications for Structural, Transportation, and Environmental Engineering, CRC Press, Boca Raton, FL, 1993.

Atmadja, J. "Computer Simulation of Behaviors of Welded Structures". Progress Report: Joint MIT - Industry Program on Tanker Safety, Department of Ocean Engineering, MIT, January 1994.

Atmadja, Juliana. Graduate student. Department of Ocean Engineering, MIT, 1994.

Bedard, C., and Gowri, K., "Automating Building Design Process with KBES", *Journal of Computing in Civil Engineering*, v 4, no. 2, pp. 69-83, April, 1990.

Bertini, Leonardo, "Material Data Bases for Offshore Platforms, Structural and Reliability Analysis" OMAE, ASME, v II: *Safety and Reliability*, pp. 253-262, 1994.

Bracco, M. Grounding Resistance of Longitudinally Stiffened Single and Double Hulls,M.S. Thesis, Department of Ocean Engineering, MIT, Report No. 29: Joint MIT - Industry Program on Tanker Safety, June 1994,.

Cavanaugh, P.F., and Billatos, S. "An Expert System for Selection of Plastic Parts", *Japan/US Symposium on Flexible Automation*, ASME, v 2, pp. 1473-1477, 1992.

Edwards, John S., Building Knowledge-Based Systems - towards a methodology, Pitman Publishing, London,1991.

Eyres, D.J., Ship Construction, (3rd edition), Heinemann Professional Publishing, London, 1988.

Fukuda, A., "Development of WELDA: An Advisor Expert System for Welding", Knowledge Based Expert Systems for Engineering: Classification, Education, and Control, D. Sriram and R.A. Adey (editors), Computational Mechanics Publications, Great Britain, 1987.

Ghosh, D.K., and Kalyanaraman, V., "KBES for Design of Steel Structural Elements", *Journal of Computing in Civil Engineering*, v 7, no. 1, pp. 23-35, January, 1993.

Gonzalez, Francisco Fernandez. Modeling of Ship-Obstacle Interaction and Hull Structural Definition for Damage Calculations. Report No. 32: Joint MIT - Industry Program on Tanker Safety, Department of Ocean Engineering, MIT, June 1994.

Gonzalez, Francisco Fernandez. Professor UPMadrid (Spain), MIT visiting Professor of Naval Architecture and Ship Construction, Department of Ocean Engineering, MIT, 1993-1994.

Joint Industry Project on Grounding Protection of Oil Tankers, Addendum, March 1992.

Kirkov, K., Tearing Resistance for Fillet Welds in Ships Exposed to Grounding, A Full-Scale Test and Cost Implications, M.S. Thesis, Department of Ocean Engineering, MIT, Report No. 30: Joint MIT - Industry Program on Tanker Safety, June 1994.

Koga, S. Visiting Scientist of Kawasaki Heavy Industries "Present Status of VLCC Bottom Hull Welding Procedures in Japan". unpublished. Welding Systems Laboratory, Department of Ocean Engineering, MIT, January 26, 1993.

Liang, Li. Graduate student. Department of Ocean Engineering, MIT, 1994/95.

Maher, M.L., Expert Systems for Civil Engineers, Techniques and Applications, ASCE, New York, NY, 1987.

Malaureille, P., "MULTIDIAG: A Parameterizable Expert System for Diagnosis of Mechanical Failure", Knowledge Based Expert Systems for Engineering: Classification, Education, and Control, D. Sriram and R.A. Adey (editors), Computational Mechanics Publications, Great Britain, 1987.

Masubuchi, K. Kawasaki Professor of Materials Science and Ocean Engineering. Department of Ocean Engineering, MIT, 1993-1995.

McClintock, F. Fully Plastic Mechanics for Welded T-Joints. Report No.26: Joint MIT - Industry Program on Tanker Safety, Department of Ocean Engineering, MIT, January 1994.

McClintock, F. Professor Emeritus of Mechanical Engineering, Senior Lecturer, MIT. 1994-1995.

McDonald, H.A., Required Strength and Tear Resistance for Fillet Welds in Ships Exposed to Grounding or Collision Loads. M.S. Thesis, Department of Ocean Engineering, MIT, Report No. 14: Joint MIT - Industry Program on Tanker Safety, May 1993.

Middaugh, R. "Vicker's Microhardness Measurements of Fillet Welds Made from a Low-Alloy Shipping Steel". unpublished, in care of Professor K. Masubuchi, Department of Ocean Engineering, MIT, 1994.

Nakamura, T., A Computer Aided System for Space Welding, M.S. Thesis, Department of Civil and Environmental Engineering, MIT, January, 1994.

NEXPERT Object Fundamentals, Macintosh Version 1.1 (system manual), Neuron Data Inc., Palo Alto, CA, 1987.

NEXPERT Object User's Guide, Version 3.0 (system manual), Neuron Data Inc., Palo Alto, CA, 1994.

Peers, S.M.C., Tang, M.X., and Dharmavasan, S., "Knowledge Based Approach to Inspection Planning for Offshore Structures", OMAE, v II: *Safety and Reliability*, ASME, pp. 263-269, 1994.

Poudret, J., Huther, M., Jean, P., and Vauhan, H., "Grounding of a Membrane Tanker; Correlation Between Damage Predictions and Observations", *Extreme Loads Response Symposium*, SNAME, New York, NY, pp. 125-131, October 19-20, 1981.

President and Fellows of Harvard College, "Smart Wave (A): The Wave-Soldering Expert System", #9-187-062, Harvard Business School Case Services, Boston, MA, 1986.

President and Fellows of Harvard College, "Smart Wave (B): Implementing an Expert System at Digital Equipment Corporation", #9-187-063, Harvard Business School Case Services, Boston, MA, 1986.

Price, C.J., Knowledge Engineering Toolkits, Ellis Horwood Books, New York, NY, 1990.

Reddy, D.V., Arockiasamy, M., Badve, A.P., Connor, J., and Sriram, D., "A Knowledge Based Approach for he Design and Analysis of Offshore Structures", *7th International Conference on Offshore Mechanics and Arctic Engineering*, Houston, Texas, February 7-12, 1988.

Sen, P. and M.K. Gerigk. "Some Aspects of a Knowledge Based Expert System for Preliminary Ship Subdivision Design for Safety" Department of Marine Technology, Newcastle University, NE1 7RU, 1994.

Siddall, James N., Expert System for Engineers, Marcel Dekker Inc., New York, NY, 1990.

Sinmao, M. Special report on the U.S. Coast Guard's Casualty Data Base. in care of Professor Thomasz Wierzbicki, Department of Ocean Engineering, MIT, Joint MIT - Industry Program on Tanker Safety. 1994.

Sriram, D., and Adey, R.A., (editors), Knowledge Based Expert Systems for Engineering: Classification, Education, and Control, Computational Mechanics Publications, Great Britain, 1987.

Taggart, R. (editor), Ship Design and Construction, SNAME, New York, NY, 1980.

Taylor, D.A., Merchant Ship Construction, (2nd edition), Butterworths, Boston, MA, 1985.

Thornbrugh, A.L., "Organizing Multiple Expert Systems: A Blackboard Based Executive Application", Knowledge Based Expert Systems for Engineering: Classification, Education, and Control, D. Sriram and R.A. Adey (editors), Computational Mechanics Publications, Great Britain, 1987.

Wang, X. Peeling Type Fracture of Ship Hull Plate Due to Grounding. MIT, Report No.13: Joint MIT - Industry Program on Tanker Safety, unpublished, May, 1993.

Waterman, D.A., A Guide to Expert Systems, Addison-Wesley Publishing Co., Reading, MA, 1986.

Wierzbicki, T., Rady, E., Peer, D., and Shin. JAG., Damage Estimates in High Energy Grounding of Ships, Report No. 1: Joint MIT - Industry Program on Tanker Safety, Department of Ocean Engineering, MIT, June 1990.

Appendix 1

The following is a list of rules and objects I created for the Fillet Weld KBES. The format is from the Nexpert Object version 3.0 logic shell **[Neuron Data 1994]**. Prinicipal headings refer to the Knowledge Bases. Classes and Objects are listed first, with their associated properties. The list of metaslots, indicated by the @META prompt, each has its own Prompt, Why, and Comment notes, which are defined by the developer to create a user friendly environment. The rules (@RULE), their repesective conditions (@LHS = left hand side), hypothesis (@HYPO), and associated tasks (@RHS = right hand side) are listed at the end of each setion. The rules also may have their own Why, Comment, and Prompt text.

A. Limitload (Weld Strength) Knowledge Base

```
(@VERSION=   030)
(@PROPERTY=          double_sided      @TYPE=Boolean;)
(@PROPERTY=          homogeneous       @TYPE=Boolean;)
(@PROPERTY=          leglength         @TYPE=Float;)
(@PROPERTY=          leglength_1       @TYPE=Float;)
(@PROPERTY=          leglength_2       @TYPE=Float;)
(@PROPERTY=          limitload         @TYPE=Float;)
(@PROPERTY=          nonhardening      @TYPE=Boolean;)
(@PROPERTY=          offered   @TYPE=String;)
(@PROPERTY=          plastic_only      @TYPE=Boolean;)
(@PROPERTY=          shear_strength    @TYPE=Float;)
(@PROPERTY=          tensile_strength  @TYPE=Float;)
(@PROPERTY=          thickness         @TYPE=Float;)
(@PROPERTY=          transverse_tensile          @TYPE=Boolean;)
(@PROPERTY=          weld_length       @TYPE=Float;)

(@CLASS=      weld_type
        (@PROPERTIES=
                weld_length

(@OBJECT=     applied_load
        (@PROPERTIES=
                transverse_tensile
                Value    @TYPE=Float;

(@OBJECT=     butt_weld
        (@CLASSES=
                weld_type)
        (@PROPERTIES=
                weld_length

(@OBJECT=     calc_approx_web_shear_strength
        (@PROPERTIES=
```

```
                Value    @TYPE=Boolean;

(@OBJECT=       calc_approx_web_shear_strength_needed
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       calc_approx_weld_shear_strength
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       calc_approx_weld_shear_strength_needed
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       calc_ideal_web_limitload
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       calc_ideal_web_limitload_needed
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       calc_ideal_weld_limitload
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       calc_ideal_weld_limitload_needed
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       calc_nonideal_web_limitload
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       calc_nonideal_weld_limitload
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       calc_web_limitload
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       calc_web_limitload_needed
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       calc_web_shear_strength
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       calc_web_shear_strength_needed
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       calc_weld_limitload
        (@PROPERTIES=
                Value    @TYPE=Boolean;
```

```
(@OBJECT=     calc_weld_limitload_needed
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=     calc_weld_shear_strength
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=     calc_weld_shear_strength_needed
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=     calc_weld_tensile_strength
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=     fillet_weld
        (@CLASSES=
                weld_type)
        (@PROPERTIES=
                double_sided
                weld_length

(@OBJECT=     ideal_fillet_weld
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=     likely
        (@PROPERTIES=
                Value    @TYPE=Float;

(@OBJECT=     major_deformation
        (@PROPERTIES=
                Value    @TYPE=Integer;

(@OBJECT=     make_weld_leglengths_equal
        (@PROPERTIES=
                Value    @TYPE=String;

(@OBJECT=     make_weld_legs_equal_lengths
        (@PROPERTIES=
                Value    @TYPE=String;

(@OBJECT=     make_weld_lengths_equal
        (@PROPERTIES=
                Value    @TYPE=String;

(@OBJECT=     massive_deformation
        (@PROPERTIES=
                Value    @TYPE=Float;

(@OBJECT=     massive_joint_deformation_likely
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=     massive_joint_deformation_possible
        (@PROPERTIES=
                Value    @TYPE=Boolean;
```

```
(@OBJECT=       massive_joint_deformation_unlikely
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       metal_deformation
        (@PROPERTIES=
                plastic_only

(@OBJECT=       need_approx_web_shear_strength
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       peeling_failure
        (@PROPERTIES=
                Value    @TYPE=Integer;

(@OBJECT=       possible
        (@PROPERTIES=
                Value    @TYPE=Float;

(@OBJECT=       suggest
        (@PROPERTIES=
                Value    @TYPE=String;

(@OBJECT=       suggestion
        (@PROPERTIES=
                offered
                Value    @TYPE=String;

(@OBJECT=       suggestions_offered
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       suggest_increase_weld_penetration
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       suggest_increase_weld_size
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       suggest_joint_is_adequate
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       suggest_joint_is_good
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       suggest_joint_is_inadequate
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=       suggest_joint_may_be_inadequate
        (@PROPERTIES=
                Value    @TYPE=Boolean;
```

```
(@OBJECT=      suggest_make_weld_leglengths_equal
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=      suggest_make_weld_lengths_equal
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=      unlikely
        (@PROPERTIES=
                Value    @TYPE=Float;

(@OBJECT=      web
        (@PROPERTIES=
                limitload
                shear_strength
                tensile_strength
                thickness

(@OBJECT=      web_fails_first
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=      web_failure_likely
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=      web_shear_strength_unknown
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=      weld
        (@PROPERTIES=
                leglength_1
                leglength_2
                limitload
                shear_strength
                tensile_strength

(@OBJECT=      weld_fails_first
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=      weld_failure_likely
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=      weld_metal
        (@PROPERTIES=
                homogeneous
                nonhardening

(@OBJECT=      weld_peeling_failure_likely
        (@PROPERTIES=
                Value    @TYPE=Boolean;

(@OBJECT=      weld_peeling_failure_possible
        (@PROPERTIES=
```

Value @TYPE=Boolean;

(@OBJECT= weld_shear_strength_unknown
(@PROPERTIES=
Value @TYPE=Boolean;

(@META= transverse_tensile
@HELP="The figure shown indicates a transverse tensile load for a double-sided fillet weld. The analysis for determining weld peeling failure only applies when this loading condition is TRUE.";

(@META= applied_load.transverse_tensile
@PROMPT="Is the load applied to the weldment a tensile load, transverse to the joint?";
@COMMENTS="To determine whether weld peeling failure is likely, the following analysis requires transverse and tensile joint loads.";
@WHY="To determine whether weld peeling failure is likely, the following analysis requires transverse and tensile joint loads.";
@HELP="This value must be true for a limitload analysis";

(@META= applied_load.Value
@PROMPT="How large is the load applied to the weld (ksi)?";
@COMMENTS="In order to determine if weld peeling failure is likely, the applied load must be known or estimated. For grounding conditions, load estimates may be available in the future. These can be added to the system as default values.";
@WHY="In order to determine if weld peeling failure is likely, the applied load must be known or estimated.";

(@META= fillet_weld.double_sided
@PROMPT="Is the weld a double-sided fillet weld?";
@COMMENTS="The analysis to calculate the weld limit load applies to a double sided fillet weld.";
@WHY="The analysis to calculate the weld limit load applies to a double sided fillet weld. Using this solution for a single sided fillet weld introduces a currently unknown error factor. ";

(@META= metal_deformation.plastic_only
@PROMPT="Does the weld metal exibit plastic deformation only?";
@COMMENTS="The present solution for the weld limit load assumes that there is negligible elastic deformation, or primarily plastic deformation within the weld region.";
@WHY="The present solution for the weld limit load assumes that there is negligible elastic deformation, or primarily plastic deformation within the weld region.";

(@META= suggest_make_weld_leglengths_equal.Value
@COMMENTS="Welds should have equal size leg lengths to maximize strength, critical displacement, and tear resistance when exposed to transverse tensile loads.
*[Kirkov, K. \"Tearing Resistance for Fillet Welds in Ships Exposed to Grounding, A Full-Scale Test and Cost Analysis\", M.S. Thesis, Department of Ocean Engineering, MIT, June 1994, Report #29, Joint MIT-Industry Program for Tanker Safety.]";
@WHY="Welds with different leg length sizes have proven to be more brittle and less tear resistant when exposed to transverse tensile loads than welds with even leg lengths.
*[Kirkov, K. \"Tearing Resistance for Fillet Welds in Ships Exposed to Grounding, A Full-Scale Test and Cost Analysis\", M.S. Thesis, Department of Ocean Engineering, MIT, June 1994, Report #29, Joint MIT-Industry Program for Tanker Safety.]";

(@META= web.limitload
@PROMPT="Please indicate a value for the web limit load of the joint (ksi).";
@COMMENTS="The web limitload is required in order to establish whether weld peeling failure is likely. If this value is unknown, please indicate \"NOTKNOWN\".";
@WHY="The web limitload is required in order to establish whether weld peeling failure is likely. If this value is unknown, please indicate \"NOTKNOWN\".";

(@META= web.shear_strength
@PROMPT="What is the shear strength of the web metal (ksi)?";
@COMMENTS="To determine the limit load, the shear strength of the metal must be known. If this value is not available, please select \"NOTKNOWN\".";
@WHY="To determine the limit load, the shear strength of the metal must be known. If this value is not available, please select \"NOTKNOWN\".";

(@META= web.tensile_strength
@PROMPT="What is the tensile strength of the web metal (ksi)?";
@COMMENTS="To approximate the shear strength of the web metal, the tensile strength must be known.";
@WHY="To approximate the shear strength of the web metal, the tensile strength must be known.";

(@META= web.thickness
@PROMPT="What is the web thickness for the weldment (mm)?";
@COMMENTS="To calculate the limit load of the web, the web thickness must be known.";
@WHY="To calculate the limit load of the web, the web thickness must be known.";

(@META= weld.leglength_1
@PROMPT="What is the length of fillet weld's first leg (mm)?";
@COMMENTS="To determine the limit load of the fillet weld, the lengths of both legs of the fillet must be known.";
@WHY="To determine the limit load of the fillet weld, the lengths of both legs of the fillet must be known.";

(@META= weld.leglength_2
@PROMPT="What is the length of the fillet weld's second leg (mm)?";
@COMMENTS="To determine the limit load for the fillet weld, the size of both legs must be known.";
@WHY="To determine the limit load for the fillet weld, the size of both legs must be known. Leglength_1 and leglength_2 are arbitrarily chosen. You may provide these values in any order.";

(@META= weld.limitload
@PROMPT="Please indicate a value for the weld limit load of the joint (ksi).";
@COMMENTS="The weld limit load must be known to determine whether weld peeling failure is likely. If this value is not known, please select \"NOTKNOWN\".";
@WHY="The weld limit load must be known to determine whether weld peeling failure is likely. If this value is not known, please select \"NOTKNOWN\".";

(@META= weld.shear_strength
@PROMPT="What is the shear strength of the weld metal (ksi)?";
@COMMENTS="To calculate the limit load of the weld, its size must be known. If this value is not available, please select \"NOTKNOWN\".";
@WHY="To calculate the limit load of the weld, its size must be known. If this value is not available, please select \"NOTKNOWN\".";

(@META= weld.tensile_strength
@PROMPT="What is the tensile strength of the weld metal (ksi)?";
@COMMENTS="To approximate the weld shear strength, the tensile strength of the weld metal must be known.";

```
        @WHY="To approximate the weld shear strength, the tensile strength of the weld metal must be
known.";

(@META=         weld_metal.homogeneous
        @PROMPT="Is the weld metal homogeneous?";
        @COMMENTS="The present analysis assumes that the weld metal is mostly homogeneous in the
immediate weld region. This does not include the weld HAZ region. *[reference: Gina Middaugh]";
        @WHY="The present analysis assumes that the weld metal is mostly homogeneous in the
immediate weld region. This does not include the weld HAZ region. *[reference: Gina Middaugh]";

(@META=         weld_metal.nonhardening
        @PROMPT="Does the weld metal exibit non-hardening behavior during plastic deformation?";
        @COMMENTS="The present analysis assume that the mechanical properties of the weld remain
constant during plastic deformation. The assumes a non-hardening metal.";
        @WHY="The present analysis assume that the mechanical properties of the weld remain constant
during plastic deformation. The assumes a non-hardening metal.";


(@RULE=         R_calc_approx_web_shear_strength
        (@LHS=
                (Yes    (calc_web_shear_strength_needed)))
        (@HYPO=         calc_approx_web_shear_strength_needed)
        (@RHS=
                (Reset  (web.shear_strength))
                (Assign (0.75*web.tensile_strength)         (web.shear_strength))

(@RULE=         R_calc_approx_weld_shear_strength
        (@LHS=
                (Yes    (calc_weld_shear_strength_needed)))
        (@HYPO=         calc_approx_weld_shear_strength_needed)
        (@RHS=
                (Reset  (weld.shear_strength))
                (Assign (0.75*weld.tensile_strength)        (weld.shear_strength))

(@RULE=         R_calc_ideal_web_limitload
        (@LHS=
                (Yes    (calc_web_limitload_needed))
                (Show   ("weld dimensions")         (@KEEP=FALSE;@WAIT=TRUE;@RECT=50,50;))
                (=      (weld.leglength_1)          (weld.leglength_2))
                (Yes    (ideal_fillet_weld))
                (<>     (web.shear_strength)        (NOTKNOWN)))
        (@HYPO=         calc_ideal_web_limitload_needed)
        (@RHS=
                (Reset  (web.limitload))
                (Assign (2*(web.shear_strength*web.thickness))      (web.limitload))

(@RULE=         R_calc_ideal_weld_limitload
        (@LHS=
                (Yes    (calc_weld_limitload_needed))
                (=      (weld.leglength_1)          (weld.leglength_2))
                (Yes    (ideal_fillet_weld))
                (<>     (weld.shear_strength)       (NOTKNOWN)))
        (@HYPO=         calc_ideal_weld_limitload_needed)
        (@RHS=
                (Reset  (weld.limitload))
                (Assign (2*(weld.shear_strength*weld.leglength_1))  (weld.limitload))

(@RULE=         R_calc_web_limitload
```

```
        @COMMENTS="if the user does not know the limit load equations exist to calculate them";
        @WHY="By comparing limitloads for both the weld and web of a joint, the system predicts which
fails first. Values are required for both components for this analysis.";
        (@LHS=
                (=      (web.limitload)   (NOTKNOWN)))
        (@HYPO=         calc_web_limitload_needed)

(@RULE=         R_calc_web_shear_strength
        (@LHS=
                (=      (web.shear_strength)      (NOTKNOWN)))
        (@HYPO=         calc_web_shear_strength_needed)

(@RULE=         R_calc_weld_limitload
        (@LHS=
                (=      (weld.limitload)   (NOTKNOWN)))
        (@HYPO=         calc_weld_limitload_needed)

(@RULE=         R_calc_weld_shear_strength
        (@LHS=
                (=      (weld.shear_strength)      (NOTKNOWN)))
        (@HYPO=         calc_weld_shear_strength_needed)

(@RULE=         R_calc_weld_tensile_strength
        (@LHS=
                (=      (weld.shear_strength)      (NOTKNOWN)))
        (@HYPO=         weld_shear_strength_unknown)
        (@RHS=
                (Reset  (weld.shear_strength))
                (Assign (0.75*weld.tensile_strength)        (weld.shear_strength))

(@RULE=         R_ideal_fillet_weld
        @COMMENTS="Limit load analysis for a weld under transverse tensile loads assumes a double
sided fillet weld that is homogeneous within the weld metal, that deforms plastically, and that does not
harden significantly during deformation. ";
        @WHY="Limit load analysis for a weld under transverse tensile loads assumes a double sided fillet
weld that is homogeneous within the weld metal, that deforms plastically, and that does not harden
significantly during deformation. ";
        (@LHS=
                (Show   ("ideal fillet")    (@KEEP=FALSE;@WAIT=TRUE;))
                (Yes    (fillet_weld.double_sided))
                (Yes    (metal_deformation.plastic_only))
                (Yes    (weld_metal.homogeneous))
                (Yes    (weld_metal.nonhardening)))
        (@HYPO=         ideal_fillet_weld)

(@RULE=         R_massive_joint_deformation_likely
        (@LHS=
                (Yes    (web_failure_likely)))
        (@HYPO=         massive_joint_deformation_likely)
        (@RHS=
                (Show   ("massive deformation")   (@KEEP=FALSE;@WAIT=TRUE;)))

(@RULE=         R_massive_joint_deformation_possible
        (@LHS=
                (Yes    (web_fails_first)))
        (@HYPO=         massive_joint_deformation_possible)

(@RULE=         R_massive_joint_deformation_unlikely
```

```
        (@LHS=
                (Yes    (weld_failure_likely)))
        (@HYPO=         massive_joint_deformation_unlikely)

(@RULE=         R_suggestions_offered
        @COMMENTS="Weld leg lengths should be equal for maximum strength and tear resistance
when exposed to transverse tensile loads.               *[Kirkov, K. \"Tearing Resistance for Fillet Welds
in SHips Exposed to Grounding, A Full-Scale Test and Cost Analysis\", M.S. Thesis, Department of
Ocean Engineering, MIT, June 1994, Report#29, Joint MIT-Industry Program on Tanker Safety.";
        @WHY="Weld leg lengths should be equal for maximum strength and tear resistance when
exposed to transverse tensile loads.";
        (@LHS=
                (<>     (weld.leglength_1)      (weld.leglength_2)))
        (@HYPO=         suggest_make_weld_leglengths_equal)
        (@RHS=
                (Show   ("Kirk's Data")   (@KEEP=FALSE;@WAIT=TRUE;@RECT=50,50;)))

(@RULE=         R_suggest_increase_weld_penetration
        @COMMENTS="An increased weld penetration has been hypothesized to increase weld strength,
weld limit load, tearing resistance, and critical displacement.
*[reference: McClintock, Masubuchi, and Wang, 1983]";
        @WHY="An increased weld penetration has been hypothesized to increase weld strength, weld
limit load, tearing resistance, and critical displacement.
*[reference: McClintock, Masubuchi, and Wang, 1983]";
        (@LHS=
                (Yes    (weld_failure_likely)))
        (@HYPO=         suggest_increase_weld_penetration)

(@RULE=         R_suggest_increase_weld_size
        @COMMENTS="Larger welds exhibit better strengths, tear resistance, and critical displacement
when subjected to transvers tensile loads.      *[reference: Kirkov, K. \"Tearing Resistance for Fillet Welds
in Ships Exposed to Grounding, A Full-Scale Test and Cost Analysis\", M.S. Thesis, Department of
Ocean Engineering, MIT, June 1994, Report#29, Joint MIT-Industry Program on Tanker Safety.]";
        @WHY="Larger welds exhibit better strengths, tear resistance, and critical displacement when
subjected to transvers tensile loads.      *[reference: Kirkov, K. \"Tearing Resistance for Fillet Welds in
Ships Exposed to Grounding, A Full-Scale Test and Cost Analysis\", M.S. Thesis, Department of Ocean
Engineering, MIT, June 1994, Report#29, Joint MIT-Industry Program on Tanker Safety.]";
        (@LHS=
                (Yes    (weld_failure_likely)))
        (@HYPO=         suggest_increase_weld_size)
        (@RHS=
                (Show   ("Kirk's Data")   (@KEEP=FALSE;@WAIT=TRUE;)))

(@RULE=         R_suggest_joint_is_good
        (@LHS=
                (Yes    (web_fails_first)))
        (@HYPO=         suggest_joint_is_adequate)

(@RULE=         R_suggest_joint_is_inadequate
        (@LHS=
                (Yes    (weld_fails_first)))
        (@HYPO=         suggest_joint_may_be_inadequate)

(@RULE=         R_web_fails_first
        (@LHS=
                (>      (weld.limitload-web.limitload)      (0)))
        (@HYPO=         web_fails_first)
```

```
(@RULE=     R_web_failure_likely
        (@LHS=
                (Yes    (web_fails_first))
                (>      (applied_load-web.limitload)        (0))
                (Yes    (applied_load.transverse_tensile)))
        (@HYPO=     web_failure_likely))

(@RULE=     R_web_shear_strength_unknown
        (@LHS=
                (=      (web.shear_strength)        (NOTKNOWN)))
        (@HYPO=     web_shear_strength_unknown)
        (@RHS=
                (Reset  (web.shear_strength))
                (Assign (0.75*web.tensile_strength)         (web.shear_strength)))

(@RULE=     R_weld_fails_first
        @WHY="To determine if weld failure is likely, both web and weld limit loads must be known.
The smaller limit load falis first.                         *   If these values are unknown, please select
\"NOTKNOWN\"";
        (@LHS=
                (>      (web.limitload-weld.limitload)      (0)))
        (@HYPO=     weld_fails_first)

(@RULE=     R_weld_failure_likely
        (@LHS=
                (Yes    (weld_fails_first))
                (>      (applied_load-weld.limitload)       (0))
                (Yes    (applied_load.transverse_tensile)))
        (@HYPO=     weld_failure_likely)

(@RULE=     R_weld_peeling_failure_likely
        @COMMENTS="Recent tanker grounding incidents, like the Exxon Valdez experienced weld
peeling failure. This mode of failure can result in major hull damage.";
        @WHY="Recent tanker grounding incidents, like the Exxon Valdez, have experienced weld peeling
failures in joints along the bottom hull. This mode of failure can result in major hull damage.
The following analysis compares the limit load of a joint's web and weld to determine whether or not weld
peeling is possible under tensile web loading. For this analysis, the following conditions must be satisfied:
1. Double-sided fillet weld.                                2. Tensile web loads
applied transverse to weld.                 3. Homogeneous weld metal, not including weld HAZ.
4. Plastic deformation only, in weld metal.                 5. Non-hardening weld metal
during deformation.";
        (@LHS=
                (Show   ("fillet load")     (@KEEP=FALSE;@WAIT=TRUE;@RECT=50,50;))
                (Yes    (applied_load.transverse_tensile))
                (Yes    (weld_fails_first))
                (>      (applied_load)      (weld.limitload)))
        (@HYPO=     weld_peeling_failure_likely)
        (@RHS=
                (Show   ("peeling failure")         (@KEEP=FALSE;@WAIT=TRUE;))

(@RULE=     R_weld_peeling_failure_possible
        (@LHS=
                (Yes    (weld_fails_first)))
        (@HYPO=     weld_peeling_failure_possible)
```

B. Rootgap Knowledge Base

```
(@VERSION=  030)
(@PROPERTY=         correction_required         @TYPE=Boolean;)
(@PROPERTY=         fatigue_life        @TYPE=String;)
(@PROPERTY=         increase_weld_buildup       @TYPE=Boolean;)
(@PROPERTY=         large       @TYPE=Boolean;)
(@PROPERTY=         no_rootgap_changes_required         @TYPE=Boolean;)
(@PROPERTY=         reduce_rootgap      @TYPE=Boolean;)
(@PROPERTY=         reduce_stress_concentration         @TYPE=Boolean;)
(@PROPERTY=         rootgap_2_to_3mm        @TYPE=Boolean;)


(@CLASS=        hypotheses

(@OBJECT=       fatigue_life_reduced
        (@CLASSES=
                hypotheses)
        (@PROPERTIES=
                Value   @TYPE=Boolean;

(@OBJECT=       rootgap_adequate
        (@CLASSES=
                hypotheses)
        (@PROPERTIES=
                Value   @TYPE=Boolean;

(@OBJECT=       rootgap_large
        (@CLASSES=
                hypotheses)
        (@PROPERTIES=
                Value   @TYPE=Boolean;

(@OBJECT=       rootgap_small
        (@CLASSES=
                hypotheses)
        (@PROPERTIES=
                Value   @TYPE=Boolean;

(@OBJECT=       suggested_action
        (@PROPERTIES=
                correction_required
                increase_weld_buildup
                no_rootgap_changes_required
                reduce_rootgap
                reduce_stress_concentration
                rootgap_2_to_3mm

(@OBJECT=       weld
```

```
        (@SUBOBJECTS=
                weld_buildup
                rootgap
                residual_stress
                weld_type

(@OBJECT=       weld_defective
        (@PROPERTIES=
                Value   @TYPE=Boolean;

(@OBJECT=       weld_strength_reduced
        (@CLASSES=
                hypotheses)
        (@PROPERTIES=
                Value   @TYPE=Boolean;

(@OBJECT=       weld_strength_severely_reduced
        (@CLASSES=
                hypotheses)
        (@PROPERTIES=
                Value   @TYPE=Boolean;

(@RULE=         RGs3
        (@LHS=
                (Yes    (rootgap_small)))
        (@HYPO=         fatigue_life_reduced)


(@RULE=         RGL2
        (@LHS=
                (Yes    (rootgap_large))
                (<=     (rootgap.height)  (5))
                (=      (weld_buildup.status)       ("performed")))
        (@HYPO=         rootgap_adequate)


(@RULE=         RG
        (@LHS=
                (Yes    (weld_type.fillet_weld))
                (>=     (rootgap.height)  (2))
                (<=     (rootgap.height)  (3)))
        (@HYPO=         rootgap_adequate)
        (@RHS=
                (Assign (TRUE)          (suggested_action.no_rootgap_changes_required)))

(@RULE=         RGL1
        (@LHS=
                (Yes    (weld_type.fillet_weld))
                (>      (rootgap.height)  (3)))
        (@HYPO=         rootgap_large))

(@RULE=         RGs1
        (@LHS=
                (Yes    (weld_type.fillet_weld))
                (<      (rootgap.height)  (2)))
        (@HYPO=         rootgap_small)
        (@RHS=
                (Assign (TRUE)          (residual_stress.large))
```

```
                (Assign (TRUE)      (suggested_action.rootgap_2_to_3mm))

(@RULE=         RGs2
        (@LHS=
                (Yes    (rootgap_small)))
        (@HYPO=         stress_concentration_severe)
        (@RHS=
                (Assign (TRUE)          (suggested_action.reduce_stress_concentration))

(@RULE=         R12
        (@LHS=
                (Yes    (weld_strength_reduced)))
        (@HYPO=         weld_defective))

(@RULE=         WS2
        (@LHS=
                (Yes    (weld_strength_severely_reduced)))
        (@HYPO=         weld_defective))

(@RULE=         WS1
        (@LHS=
                (Yes    (stress_concentration_severe)))
        (@HYPO=         weld_defective))

(@RULE=         RGL3
        (@LHS=
                (Yes    (rootgap_large))
                (=      (weld_buildup.status)       ("none"))
                (<=     (rootgap.height)  (5)))
        (@HYPO=         weld_strength_reduced)
        (@RHS=
                (Assign (TRUE)          (suggested_action.increase_weld_buildup))
                (Assign (TRUE)          (suggested_action.rootgap_2_to_3mm)))

(@RULE=         RGL4
        (@LHS=
                (Yes    (rootgap_large))
                (>      (rootgap.height)  (5)))
        (@HYPO=         weld_strength_severely_reduced)
        (@RHS=
                (Assign (TRUE)          (suggested_action.reduce_rootgap))
                (Assign (TRUE)          (suggested_action.correction_required))
```

C. Undercut Knowledge Base

```
(@VERSION=   030)
(@PROPERTY=          area      @TYPE=String;)
(@PROPERTY=          at_tip    @TYPE=String;)
(@PROPERTY=          brittle   @TYPE=String;)
(@PROPERTY=          brittle_fracture     @TYPE=String;)
(@PROPERTY=          cross_section_area          @TYPE=String;)
(@PROPERTY=          defective           @TYPE=Boolean;)
(@PROPERTY=          design_load         @TYPE=String;)
(@PROPERTY=          energy_absorption           @TYPE=String;)
(@PROPERTY=          exists    @TYPE=Boolean;)
(@PROPERTY=          failure   @TYPE=String;)
(@PROPERTY=          fatigue_at_tip      @TYPE=String;)
(@PROPERTY=          fillet    @TYPE=Boolean;)
(@PROPERTY=          fillet_weld         @TYPE=Boolean;)
(@PROPERTY=          increase_weld_fill          @TYPE=Boolean;)
(@PROPERTY=          load      @TYPE=String;)
(@PROPERTY=          present   @TYPE=Boolean;)
(@PROPERTY=          probability         @TYPE=String;)
(@PROPERTY=          property            @TYPE=String;)
(@PROPERTY=          requirement         @TYPE=String;)
(@PROPERTY=          shape     @TYPE=String;)
(@PROPERTY=          shorter_weld_arc    @TYPE=Boolean;)
(@PROPERTY=          size      @TYPE=Float;)
(@PROPERTY=          strength            @TYPE=String;)
(@PROPERTY=          stressconc          @TYPE=String;)
(@PROPERTY=          type      @TYPE=String;)
(@PROPERTY=          undercut_standards_adequate          @TYPE=Boolean;)
(@PROPERTY=          weld_arc            @TYPE=String;)
(@PROPERTY=          weld_fill           @TYPE=String;)

(@OBJECT=      AWS
        (@PROPERTIES=
                design_load
                undercut_standards_adequate)

(@OBJECT=      AWS_design_load
        (@PROPERTIES=
                Value     @TYPE=String;)

(@OBJECT=      AWS_undercut
        (@PROPERTIES=
                requirement)

(@OBJECT=      AWS_undercut_rule_adequate
        (@PROPERTIES=
                Value     @TYPE=Boolean;)
```

```
(@OBJECT=      AWS_undercut_standards_adequate
        (@PROPERTIES=
                Value    @TYPE=Boolean;)

(@OBJECT=      brittle_fracture
        (@PROPERTIES=
                probability
                Value    @TYPE=String;)

(@OBJECT=      brittle_fracture_highly_likely
        (@PROPERTIES=
                Value    @TYPE=Boolean;)

(@OBJECT=      brittle_fracture_likely
        (@PROPERTIES=
                Value    @TYPE=Boolean;)

(@OBJECT=      critical_loading_condition
        (@PROPERTIES=
                Value    @TYPE=Boolean;)

(@OBJECT=      cross_section
        (@PROPERTIES=
                area)

(@OBJECT=      failure
        (@PROPERTIES=
                probability)

(@OBJECT=      loading
        (@PROPERTIES=
                type)

(@OBJECT=      local_yielding_at_crack_tip_likely
        (@PROPERTIES=
                Value    @TYPE=Boolean;)

(@OBJECT=      material
        (@PROPERTIES=
                property)

(@OBJECT=      must_fix
        (@PROPERTIES=
                Value    @TYPE=Boolean;)

(@OBJECT=      probability
        (@PROPERTIES=
                brittle_fracture
                fatigue_at_tip)

(@OBJECT=      stress_concentration
        (@PROPERTIES=
                at_tip)

(@OBJECT=      stress_concentration_at_tip
        (@PROPERTIES=
                Value    @TYPE=Boolean;)
```

```
(@OBJECT=       suggest
        (@PROPERTIES=
                weld_arc
                weld_fill)

(@OBJECT=       suggested_action
        (@PROPERTIES=
                increase_weld_fill
                shorter_weld_arc)

(@OBJECT=       undercut
        (@PROPERTIES=
                exists
                present
                requirement
                shape
                size
                Value   @TYPE=String;)

(@OBJECT=       undercut_insignificant
        (@PROPERTIES=
                Value   @TYPE=Boolean;)

(@OBJECT=       undercut_not_critical .
        (@PROPERTIES=
                Value   @TYPE=Boolean;)

(@OBJECT=       undercut_possibly_critical
        (@PROPERTIES=
                Value   @TYPE=Boolean;)

(@OBJECT=       undercut_possibly_severe
        (@PROPERTIES=
                Value   @TYPE=Boolean;)

(@OBJECT=       undercut_possibly_significant
        (@PROPERTIES=
                Value   @TYPE=Boolean;)

(@OBJECT=       undercut_severe
        (@PROPERTIES=
                Value   @TYPE=Boolean;)

(@OBJECT=       undercut_tip
        (@PROPERTIES=
                stressconc)

(@OBJECT=       wedge
        (@PROPERTIES=
                Value   @TYPE=Float;)

(@OBJECT=       weld
        (@PROPERTIES=
                cross_section_area
                energy_absorption
                failure
                load
                size
```

```
                strength)

(@OBJECT=       weld_defective
        (@PROPERTIES=
                Value     @TYPE=Boolean;)

(@OBJECT=       weld_failure
        (@PROPERTIES=
                probability)

(@OBJECT=       weld_fracture
        (@PROPERTIES=
                brittle
                Value     @TYPE=String;)

(@OBJECT=       weld_material
        (@PROPERTIES=
                property)

(@OBJECT=       weld_possibly_defective
        (@PROPERTIES=
                Value     @TYPE=Boolean;)

(@OBJECT=       weld_severely_defective
        (@PROPERTIES=
                Value     @TYPE=Boolean;)

(@OBJECT=       weld_severly_defective
        (@PROPERTIES=
                Value     @TYPE=Boolean;)

(@OBJECT=       weld_state
        (@PROPERTIES=
                defective)

(@OBJECT=       weld_type
        (@PROPERTIES=
                fillet
                fillet_weld)

(@OBJECT=       weld_yielding
        (@PROPERTIES=
                at_tip)

(@OBJECT=       weldleg
        (@PROPERTIES=
                size)

(@OBJECT=       weldthroat
        (@PROPERTIES=
                area
                size)

(@RULE=         UC2
        (@LHS=
                (Yes      (weld_type.fillet_weld))
                (Yes      (undercut.present))
                (<=       (weldleg.size)      (12.7))
```

```
            (<=     (undercut.size)    (1.6)))
        (@HYPO=     AWS.undercut_standards_adequate)
        (@RHS=
            (Assign ("< 1.6mm")     (AWS_undercut.requirement))
            (Assign ("met")  (AWS.design_load))
            (Assign ("adequate")     (weld.strength)))

(@RULE=     UC6
        (@LHS=
            (Yes    (undercut_severe))
            (=      (weld_material.property)    ("brittle")))
        (@HYPO=     brittle_fracture_likely))

(@RULE=     UC9
        (@LHS=
            (Yes    (undercut_severe))
            (=      (loading.type)     ("transverse")))
        (@HYPO=     critical_loading_condition)
        (@RHS=
            (Assign ("increased")      (weld_failure.probability)))

(@RULE=     UC7
        (@LHS=
            (Yes    (undercut_severe))
            (=      (weld_material.property)    ("ductile")))
        (@HYPO=     local_yielding_at_crack_tip_likely))

(@RULE=     UC10
        (@LHS=
            (Yes    (undercut_severe))
            (=      (undercut.shape)  ("sharp"))
            (=      (weld.load)       ("severe")))
        (@HYPO=     stress_concentration_at_tip)
        (@RHS=
            (Assign ("increased")      (weld_failure.probability)))

(@RULE=     UC3
        (@LHS=
            (Yes    (weld_type.fillet_weld))
            (Yes    (undercut.present))
            (<=     (weldleg.size)     (12.7))
            (<=     (undercut.size)    (1.66)))
        (@HYPO=     undercut_insignificant)
        (@RHS=
            (Assign ("not reduced")    (weld.strength))
            (Assign ("not reduced")    (weld.energy_absorption)))

(@RULE=     UC8
        (@LHS=
            (Yes    (weld_type.fillet_weld))
            (Yes    (undercut.present))
            (<=     (weldleg.size)     (12.7))
            (>      (undercut.size)    (1.6)))
        (@HYPO=     undercut_possibly_severe)
        (@RHS=
            (Assign ("reduced")        (weldthroat.area)))

(@RULE=     UC4
```

```
        (@LHS=
                (Yes    (weld_type.fillet_weld))
                (Yes    (undercut.present))
                (>      (weldleg.size)    (12.7))
                (<      (weld.size)       (19))
                (<=     (undercut.size)   (1.6)))
        (@HYPO=         undercut_possibly_severe)
        (@RHS=
                (Assign ("reduced")       (weld.strength))
                (Assign ("reduced")       (weld.energy_absorption)))

(@RULE=         UC5
        (@LHS=
                (Yes    (weld_type.fillet_weld))
                (Yes    (undercut.present))
                (>=     (weldleg.size)    (19))
                (>=     (undercut.size)   (1.6)))
        (@HYPO=         undercut_severe)
        (@RHS=
                (Assign ("not met")       (AWS.design_load))
                (Assign ("reduced")       (weld.strength))
                (Assign ("reduced")       (weld.energy_absorption))
                (Assign ("increased")     (probability.brittle_fracture))
                (Assign ("increased")     (probability.fatigue_at_tip))
                (Assign ("increased")     (stress_concentration.at_tip)))

(@RULE=         UC1
        (@LHS=
                (Yes    (undercut_possibly_severe)))
        (@HYPO=         weld_possibly_defective))

(@RULE=         UCsug3
        (@LHS=
                (Yes    (brittle_fracture_likely)))
        (@HYPO=         weld_severely_defective)
        (@RHS=
                (Assign (TRUE)            (suggested_action.increase_weld_fill))
                (Assign (TRUE)            (suggested_action.shorter_weld_arc)))

(@RULE=         UCsug2
        (@LHS=
                (Yes    (local_yielding_at_crack_tip_likely)))
        (@HYPO=         weld_severely_defective)
        (@RHS=
                (Assign (TRUE)            (suggested_action.increase_weld_fill))
                (Assign (TRUE)            (suggested_action.shorter_weld_arc)))

(@RULE=         UCsug1
        (@LHS=
                (Yes    (undercut_severe)))
        (@HYPO=         weld_severely_defective)
        (@RHS=
                (Assign (TRUE)            (suggested_action.shorter_weld_arc))
                (Assign (TRUE)            (suggested_action.increase_weld_fill))
```

D. Structural Reference Knowledge Base

```
(@VERSION=  030)
(@PROPERTY=     average_speed   @TYPE=Integer;)
(@PROPERTY=     block_coefficient       @TYPE=Float;)
(@PROPERTY=     bottom_location @TYPE=String;)
(@PROPERTY=     bow_type        @TYPE=String;)
(@PROPERTY=     bracketed       @TYPE=Boolean;)
(@PROPERTY=     bulkhead_number         @TYPE=Integer;)
(@PROPERTY=     bulkhead_spacing        @TYPE=Float;)
(@PROPERTY=     continuous      @TYPE=Boolean;)
(@PROPERTY=     corrugation     @TYPE=Boolean;)
(@PROPERTY=     distance_from_AP        @TYPE=Float;)
(@PROPERTY=     distance_from_FP        @TYPE=Float;)
(@PROPERTY=     draft   @TYPE=Float;)
(@PROPERTY=     DWT   @TYPE=Integer;)
(@PROPERTY=     electrode_type  @TYPE=String;)
(@PROPERTY=     elongation      @TYPE=Float;)
(@PROPERTY=     flange_length   @TYPE=Float;)
(@PROPERTY=     flange_width    @TYPE=Float;)
(@PROPERTY=     flat_margin     @TYPE=Boolean;)
(@PROPERTY=     floor_height    @TYPE=Float;)
(@PROPERTY=     floor_plate_thickness   @TYPE=Float;)
(@PROPERTY=     floor_spacing   @TYPE=Float;)
(@PROPERTY=     framing_type    @TYPE=String;)
(@PROPERTY=     geometry_type   @TYPE=String;)
(@PROPERTY=     height  @TYPE=Float;)
(@PROPERTY=     height_above_outer_bottom       @TYPE=Float;)
(@PROPERTY=     hull_type       @TYPE=String;)
(@PROPERTY=     intercostal     @TYPE=Boolean;)
(@PROPERTY=     keel_height     @TYPE=Float;)
(@PROPERTY=     keel_plate_thickness    @TYPE=Float;)
(@PROPERTY=     keel_width      @TYPE=Float;)
(@PROPERTY=     LBP   @TYPE=Float;)
(@PROPERTY=     loading_conditions      @TYPE=String;)
(@PROPERTY=     LWL   @TYPE=Float;)
(@PROPERTY=     plate_thickness @TYPE=Float;)
(@PROPERTY=     ship_type       @TYPE=String;)
(@PROPERTY=     sloped_margin   @TYPE=Boolean;)
(@PROPERTY=     solid_floor     @TYPE=Boolean;)
(@PROPERTY=     spacing @TYPE=Float;)
(@PROPERTY=     steel_type      @TYPE=String;)
(@PROPERTY=     stern_type      @TYPE=String;)
(@PROPERTY=     stiffener_height @TYPE=Float;)
(@PROPERTY=     stiffener_spacing @TYPE=Float;)
(@PROPERTY=     structural_importance   @TYPE=String;)
(@PROPERTY=     tensile_strength @TYPE=Float;)
(@PROPERTY=     thickness       @TYPE=Float;)
(@PROPERTY=     volume @TYPE=Float;)
```

```
(@PROPERTY=          watertight          @TYPE=Boolean;)
(@PROPERTY=          weight    @TYPE=Integer;)
(@PROPERTY=          weld_angle          @TYPE=Integer;)
(@PROPERTY=          weld_condition      @TYPE=String;)
(@PROPERTY=          weld_width          @TYPE=Float;)
(@PROPERTY=          weldability         @TYPE=String;)
(@PROPERTY=          weldleg_length1     @TYPE=Float;)
(@PROPERTY=          weldleg_length2     @TYPE=Float;)
(@PROPERTY=          weldlength          @TYPE=Float;)
(@PROPERTY=          width     @TYPE=Float;)
(@PROPERTY=          yield_strength      @TYPE=Float;)
(@PROPERTY=          Youngs_modulus              @TYPE=Float;)


(@CLASS=       bottom_hull_structure
        (@SUBCLASSES=
                stiffeners
                weldments
                keel
                vertical_flooring
                bottom_plating
                hull_materials)
        (@PROPERTIES=
                bottom_location
                framing_type
                hull_type
                ship_type)

(@CLASS=       bottom_plating
        (@PROPERTIES=
                bottom_location
                framing_type
                hull_type
                loading_conditions
                plate_thickness
                ship_type
                structural_importance)

(@CLASS=       bulkheads
        (@PROPERTIES=
                bulkhead_number
                bulkhead_spacing
                corrugation
                distance_from_AP
                distance_from_FP
                framing_type
                hull_type
                ship_type
                watertight)

(@CLASS=       general_hull_dimensions
        (@PROPERTIES=
                block_coefficient
                draft
                DWT
                framing_type
                hull_type
                LBP
```

```
                LWL
                ship_type
                width)

(@CLASS=        general_ship_properties
        (@SUBCLASSES=
                standard_operating_conditions
                main_hull_structures
                general_hull_dimensions)
        (@PROPERTIES=
                framing_type
                hull_type
                ship_type)

(@CLASS=        hull_materials
        (@PROPERTIES=
                bottom_location
                elongation
                framing_type
                hull_type
                ship_type
                steel_type
                tensile_strength
                weldability
                yield_strength
                Youngs_modulus)

(@CLASS=        hull_sections
        (@PROPERTIES=
                distance_from_AP
                distance_from_FP
                framing_type
                hull_type
                ship_type)

(@CLASS=        keel
        (@PROPERTIES=
                bottom_location
                framing_type
                hull_type
                keel_height
                keel_plate_thickness
                keel_width
                ship_type)

(@CLASS=        main_hull_structures
        (@SUBCLASSES=
                bulkheads
                hull_sections)
        (@PROPERTIES=
                distance_from_AP
                distance_from_FP
                framing_type
                hull_type
                ship_type)

(@CLASS=        standard_operating_conditions
        (@PROPERTIES=
```

```
                average_speed
                framing_type
                hull_type
                ship_type)

(@CLASS=        stiffeners
        (@PROPERTIES=
                bottom_location
                framing_type
                hull_type
                ship_type
                stiffener_height
                stiffener_spacing
                structural_importance
                volume)

(@CLASS=        vertical_flooring
        (@PROPERTIES=
                bottom_location
                bracketed
                continuous
                floor_height
                floor_plate_thickness
                floor_spacing
                framing_type
                hull_type
                intercostal
                ship_type
                solid_floor
                watertight)

(@CLASS=        weldments
        (@PROPERTIES=
                bottom_location
                electrode_type
                weld_condition
                weldlength)


(@OBJECT=       amidship
        (@CLASSES=
                hull_sections)
        (@PROPERTIES=
                distance_from_AP
                distance_from_FP
                framing_type
                hull_type
                ship_type)

(@OBJECT=       bottom_shell_material
        (@CLASSES=
                hull_materials)
        (@PROPERTIES=
                bottom_location
                elongation
                framing_type
                hull_type
                ship_type
```

```
                steel_type
                tensile_strength
                weldability
                yield_strength
                Youngs_modulus)

(@OBJECT=       bow
        (@CLASSES=
                hull_sections)
        (@PROPERTIES=
                bow_type
                distance_from_AP
                distance_from_FP
                framing_type
                hull_type
                ship_type))

(@OBJECT=       butt_weld
        (@CLASSES=
                weldments)
        (@PROPERTIES=
                bottom_location
                electrode_type
                weld_condition
                weld_width
                weldlength))

(@OBJECT=       duct_keel
        (@CLASSES=
                keel)
        (@PROPERTIES=
                bottom_location
                framing_type
                hull_type
                keel_height
                keel_plate_thickness
                keel_width
                plate_thickness
                ship_type)

(@OBJECT=       fillet_weld
        (@CLASSES=
                weldments)
        (@PROPERTIES=
                bottom_location
                electrode_type
                structural_importance
                weld_angle
                weld_condition
                weldleg_length1
                weldleg_length2
                weldlength)

(@OBJECT=       flange
        (@SUBOBJECTS=
                flange)
        (@PROPERTIES=
                flange_length
```

```
                flange_width
                geometry_type)

(@OBJECT=       flat_plate_keel
        (@CLASSES=
                keel)
        (@PROPERTIES=
                bottom_location
                framing_type
                hull_type
                keel_height
                keel_plate_thickness
                keel_width
                plate_thickness
                ship_type)

(@OBJECT=       inner_bottom_plating
        (@CLASSES=
                bottom_plating)
        (@PROPERTIES=
                bottom_location
                framing_type
                height_above_outer_bottom
                hull_type
                loading_conditions
                plate_thickness
                ship_type
                structural_importance)

(@OBJECT=       logitudinal_girders
        (@CLASSES=
                vertical_flooring)
        (@PROPERTIES=
                bottom_location
                bracketed
                continuous
                floor_height
                floor_plate_thickness
                floor_spacing
                framing_type
                hull_type
                intercostal
                ship_type
                solid_floor
                watertight)

(@OBJECT=       longitudinal_bulkheads
        (@CLASSES=
                bulkheads)
        (@PROPERTIES=
                bulkhead_number
                bulkhead_spacing
                corrugation
                distance_from_AP
                distance_from_FP
                framing_type
                hull_type
                ship_type
```

```
                watertight)

(@OBJECT=       longitudinal_girders
        (@CLASSES=
                stiffeners)
        (@SUBOBJECTS=
                flange)
        (@PROPERTIES=
                bottom_location
                framing_type
                hull_type
                ship_type
                stiffener_height
                stiffener_spacing
                structural_importance
                volume)

(@OBJECT=       machinery_space
        (@CLASSES=
                hull_sections)
        (@PROPERTIES=
                distance_from_AP
                distance_from_FP
                framing_type
                hull_type
                ship_type
                weight)

(@OBJECT=       outer_shell_plating
        (@CLASSES=
                bottom_plating)
        (@PROPERTIES=
                bottom_location
                framing_type
                hull_type
                loading_conditions
                plate_thickness
                ship_type
                structural_importance)

(@OBJECT=       side_shell_material
        (@CLASSES=
                hull_materials)
        (@PROPERTIES=
                bottom_location
                elongation
                framing_type
                hull_type
                ship_type
                steel_type
                tensile_strength
                weldability
                yield_strength
                Youngs_modulus)

(@OBJECT=       stern
        (@CLASSES=
                hull_sections)
```

```
        (@PROPERTIES=
                distance_from_AP
                distance_from_FP
                framing_type
                hull_type
                ship_type
                stern_type))

(@OBJECT=       transverse_bulkheads
        (@CLASSES=
                bulkheads)
        (@PROPERTIES=
                bulkhead_number
                bulkhead_spacing
                corrugation
                distance_from_AP
                distance_from_FP
                framing_type
                hull_type
                ship_type
                watertight)

(@OBJECT=       transverse_flooring
        (@CLASSES=
                vertical_flooring)
        (@PROPERTIES=
                bottom_location
                bracketed
                continuous
                floor_height
                floor_plate_thickness
                floor_spacing
                framing_type
                hull_type
                intercostal
                ship_type
                solid_floor
                watertight)

(@OBJECT=       transverse_stiffeners
        (@CLASSES=
                stiffeners)
        (@PROPERTIES=
                bottom_location
                framing_type
                hull_type
                ship_type
                stiffener_height
                stiffener_spacing
                structural_importance
                volume
```

E. Weld Structural Reference Knowledge Base

```
(@VERSION=   030)
(@PROPERTY=          angle    @TYPE=Integer;)
(@PROPERTY=          butt_weld          @TYPE=Boolean;)
(@PROPERTY=          concavity          @TYPE=String;)
(@PROPERTY=          convexity          @TYPE=String;)
(@PROPERTY=          cruciform_weld     @TYPE=Boolean;)
(@PROPERTY=          current  @TYPE=Integer;)
(@PROPERTY=          distance @TYPE=Integer;)
(@PROPERTY=          exists   @TYPE=Boolean;)
(@PROPERTY=          fillet_weld        @TYPE=Boolean;)
(@PROPERTY=          GMAW               @TYPE=Boolean;)
(@PROPERTY=          height   @TYPE=Float;)
(@PROPERTY=          lapjoint_weld      @TYPE=Boolean;)
(@PROPERTY=          length   @TYPE=Float;)
(@PROPERTY=          MAW      @TYPE=Boolean;)
(@PROPERTY=          number   @TYPE=Integer;)
(@PROPERTY=          penetration        @TYPE=String;)
(@PROPERTY=          SAW      @TYPE=Boolean;)
(@PROPERTY=          SMAW     @TYPE=Boolean;)
(@PROPERTY=          spacing  @TYPE=Integer;)
(@PROPERTY=          speed    @TYPE=Integer;)
(@PROPERTY=          voltage  @TYPE=Integer;)
(@PROPERTY=          weld_length        @TYPE=Float;)
(@PROPERTY=          weld_shape         @TYPE=String;)
(@PROPERTY=          weld_technique     @TYPE=String;)
(@PROPERTY=          width    @TYPE=Float;)
(@PROPERTY=          yield_strength     @TYPE=Integer;)
(@PROPERTY=          Youngs_modulus          @TYPE=Integer;)


(@CLASS=       butt_dimensions)

(@CLASS=       continuity)

(@CLASS=       cruciform_dimensions)

(@CLASS=       fillet_dimensions)

(@CLASS=       lapjoint_dimensions)

(@CLASS=       method_parameters
        (@PROPERTIES=
                current
                distance
                speed
                voltage)

(@CLASS=       method_type
        (@PROPERTIES=
                GMAW
                MAW
```

```
                SAW
                SMAW)

(@CLASS=        pass_number)

(@CLASS=        weld_defects)

(@CLASS=        weld_dimensions
        (@SUBCLASSES=
                butt_dimensions
                cruciform_dimensions
                fillet_dimensions
                lapjoint_dimensions)

(@CLASS=        weld_method
        (@SUBCLASSES=
                continuity
                pass_number
                method_type
                method_parameters)

(@CLASS=        weld_type
        (@PROPERTIES=
                butt_weld
                cruciform_weld
                fillet_weld
                lapjoint_weld)


(@OBJECT=       centerline_notch
        (@CLASSES=
                weld_defects)

(@OBJECT=       concavity
        (@CLASSES=
                weld_defects)

(@OBJECT=       continuous
        (@CLASSES=
                continuity)
        (@PROPERTIES=
                exists)

(@OBJECT=       convexity
        (@CLASSES=
                weld_defects)

(@OBJECT=       intermitant
        (@CLASSES=
                continuity)
        (@PROPERTIES=
                exists
                spacing)

(@OBJECT=       leg1
        (@CLASSES=
                fillet_dimensions)
        (@PROPERTIES=
```

```
                angle
                length
                penetration)

(@OBJECT=       leg2
        (@CLASSES=
                fillet_dimensions)
        (@PROPERTIES=
                angle
                length
                penetration)

(@OBJECT=       multiple_pass
        (@CLASSES=
                pass_number)
        (@PROPERTIES=
                exists
                number)

(@OBJECT=       overlap
        (@CLASSES=
                weld_defects)

(@OBJECT=       rootgap
        (@CLASSES=
                fillet_dimensions)
        (@PROPERTIES=
                height
                penetration)

(@OBJECT=       rootgap_too_large
        (@CLASSES=
                weld_defects)

(@OBJECT=       single_pass
        (@CLASSES=
                pass_number)
        (@PROPERTIES=
                exists)

(@OBJECT=       undercut
        (@CLASSES=
                weld_defects)

(@OBJECT=       weld_shape
        (@CLASSES=
                fillet_dimensions)
        (@PROPERTIES=
                concavity
                convexity)

(@OBJECT=       weld_throat
        (@CLASSES=
                fillet_dimensions)
        (@PROPERTIES=
                penetration
                width
```