THE UNIVERSITY OF CALGARY

An Information Structure for Mechanical Design

by

Frank Yangqi Yin

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF MECHANICAL ENGINEERING

CALGARY, ALBERTA

DECEMBER, 1992

ISBN 0-315-83285-1

THE UNIVERSITY OF CALGARY

FACULTY OF GRADUATE STUDIES

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies for acceptance, a thesis entitled " AN INFORMATION STRUCTURE FOR MECHANICAL DESIGN " submitted by Frank Yangqi Yin in partial fulfilment of the requirements for the degree of Master of Science.

Supervisor, Dr. O.R. Fauvel
Department of Mechanical Engineering

Dr. D.H. Norrie
Department of Mechanical Engineering

Dr. P. Gu
Department of Mechanical Engineering

Dr. S. Walker
Faculty of Environmental Design

December 16, 1992

ABSTRACT

This thesis presents a Function-Environment-Embodiment (FEE) design information structure that can support and facilitate the information/knowledge communications during the design process. The FEE structure distinguishes the information involved in the design process into three different domains within which the relationships between the design functions and forms can be established. It is believed that these relationships play a major role in connecting the information from different sources into the design process.

An Object-Oriented Programming (OOP) language is applied to modelling the FEE information structure based on a particular routine design problem. The FEE structure implemented within the OOP paradigm demonstrates the ability of design extendability and code reuse features which can contribute to the flexibility and time efficiency of design process. It is also shown through the design example that it is potentially possible to integrate a wider range of information such as that from manufacturing stage into the design process.

ACKNOWLEDGEMENTS

I wish to express my deepest appreciation and gratitude to my supervisor, Dr. O.R. Fauvel for his help and guidance throughout the course of this project, and in the preparation of this manuscript. His scholarly attitude made this research period a valuable experience.

Special thanks go to Dr. D.H. Norrie for his useful advice during the study period and his time and effort toward the thesis.

I am grateful to Dr. P. Gu for his encouragement during the study period and his time and effort toward the thesis.

Thanks are also due to Dr. S. Walker for his valuable time devoted to be an examination committee member.

Financial assistance received from the Department of Mechanical Engineering is also gratefully appreciated.

To my family,

and my wife, Li Li,

for their support and love.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1. INTRODUCTION

1.1. Background

Engineering design plays a crucial role in the industrial product realization process. It is estimated that typically 70% or more of the life cycle cost of a product is determined during design stage [N.R.C][1]. Therefore, the improvement of the product design process is one of the most important issues in modern industry where it is necessary to gain a competitive edge in world-wide competition. Because an enormous amount of knowledge and information exists in various aspects of the engineering domain such as design, manufacturing, marketing and maintenance, many design activities are conducted with a lack of full awareness of the knowledge which pertains to other life cycle stages such as manufacturing and maintenance. Even though the practice of engineering within a particular domain may be satisfactory, global optimization of product quality and cost cannot be ensured without the appropriate utilization of the knowledge from all other domains of product development at the design stage.

While the integration of available knowledge and information from different life stages is desirable, the realization of this goal remains elusive. Although parts of the problem are being addressed under such headings as "Design for Manufacture", "Design

[1] References are listed at the end of thesis.

for Reliability", and "Concurrent Design", considerable effort still needs to be spent on searching for ways to solve the more general problem.

This research explores the form-function relationships in a particular design process and from this develops an information structure to logically connect information from various domains addressed by the design process. These relationships are then used to guide the implementation of the design information structure using an object-oriented paradigm in order to demonstrate the facilitation of knowledge integration in design.

1.2. Overview of Existing Design Methods

There are many design methods which are intended to address different design stages such as the conceptual, configuration, or detail design stages. Some methods cover a wider scope of the design process but others may be applicable only to a narrower range of problem-solving.

Systematic design methods have been developed to enhance the design of a range of products requiring extensive design. These methods describe the design process as a series of sequential steps; the design process has to follow each of the steps in order to make all design-related decisions.

With the rapid development of computer technology, design is being increasingly

assisted by computer-aided design methods/systems such as CAD systems, knowledge-based design systems, object-oriented design tools, and even intelligent design systems. These methods/systems in general are efficient only in their related domains; expert system design methods tend to apply to a specific domain. But many of these systems are still in the early stage of development and the application areas are also limited. Therefore, there is a need to explore new design methods and tools to take advantage of computer technology.

1.3. Perceived Existing Problems

A traditional, systematic design process (such as that described by Pahl and Beitz [1984]) is typically lengthy and there are potentially numerous iterations around small and large loops. The design process can therefore be slow and lead to a long product development cycle. This sequence can also make it difficult for 'downstream' designers to keep track of the original design functions. Since consideration of additional design factors tends to result in lengthening an already lengthy process, life-cycle optimization of a design is not facilitated by this type of design process.

Even for a routine design process, conventional design methods tend to consider product life-cycle aspects in a static way. Therefore, the design cannot take advantage of new technology developments in manufacturing and consequently product quality and economics cannot benefit from these. With the explosion of design information and

manufacturing techniques and the increasing level of quality standards there are more and more problems encountered in this conventional design process. In addition, conventional design methods are less capable of responding efficiently to changing requirements, either from the customers or from the design infrastructure. Therefore, it is necessary to integrate various types of knowledge into the design process so that many aspects about the product can be considered during the earliest possible stages of design.

1.4. Objectives

In order to examine the problems associated with mechanical design a routine design problem (bolted joint design) is modelled in such a way that the role of design-related information can be determined. The specific objective is to develop an object-oriented model of information interactions involved in designing bolted joints. This model will be useful for the development of concurrent design methods.

1.5. Approach

It is hypothesized that it is possible to map the design knowledge into a design information structure. A Function-Environment-Embodiment (FEE) tripartite structure is proposed as such a design information structure. The FEE structure classifies the knowledge/information into three distinct domains and represents it into such shapes in each domain that a design process can easily make use of it. To demonstrate how the

knowledge/information is represented and used in the design process a pipe joint design problem is presented and then analyzed within the information structure. This design problem provides material content for the development of the information structure and finally serves as a design example to prove the validity of the design information structure being used for this particular design procedure. Object-oriented language is used to model the FEE structure in an object-oriented paradigm so that a more flexible and extendable design support tool can be built.

1.6. Thesis Outline

Following this chapter, a literature survey of existing design methods is conducted in Chapter 2, from which a design problem analysis is presented in Chapter 3. Chapter 4 presents a functional approach to design problem-solving. By analyzing the relationships of design form and function, design information from different engineering domains is believed capable of being logically connected. An analysis is made of the conventional design process to guide the design of an information structure. Following the theoretical analysis a pipe joint design problem is introduced and then analyzed in the Function-Environment-Embodiment design information paradigm in Chapter 5. Then Chapter 6 presents an Object-Oriented Programming (OOP) method, followed by the joint design process implementation within the OOP paradigm. Finally, conclusions are drawn and future directions are discussed in Chapter 7.

CHAPTER 2. DESIGN METHODS REVIEW

2.1. Introduction

This chapter presents a general review of present design methods. It is not intended to be comprehensive but rather to show the variety of ways to approach design problems. It also demonstrates the advancement of design methods with the development of computer technology. The intention is to analyze and reveal the advantages and drawbacks of each design method category, from which the direction of improvement of future design methods is expected to emerge.

2.2. Traditional Design Methods

Design is often considered as a transformation process between a functional description and a physical description of a device or a technical system. Jones [1973] noted seven stages in a general design methodology as:

(1). recognition that the problem exists;

(2). study of parameters and their conversion to recognisable terms;

(3). preparation by assimilating existing knowledge and searching for other data;

(4). analysis with the intention of satisfying a feasible inquiry;

(5). synthesis - manipulation of the analysis to yield available solutions;

(6). evaluation by which a solution is chosen; and

(7). presentation - the method by which the solution is communicated.

In all these stages, the designer has to consider factors such as cost, aesthetics, and ergonomics as well as the prime function or functions required. One or more of these considerations may be paramount depending on the purpose for which the product is intended. Successful design is a compromise between all the factors involved, and a designer must have as wide a knowledge as possible of the factors to arrive at a successful compromise [Cullum].

There are many design methodologies which in principle more or less parallel the above summary. For the design of engineering products or technical systems, most systematic design approaches were originated in European publications. V. Hubka [1982] in the Principles of Engineering Design, elaborated a six-stage design process for technical systems. The general procedural design model provided is intended to serve as a guideline for engineering designers. A similar systematic approach is presented in German Technical Guidelines VDI 2221 [1986] which is focused on the design of technical systems and products. These design methods established a traditional way of designing which is also reflected in the teaching of engineering design.

While the systematic design approaches are still serving as guidelines in many industries, computer applications to the design process have already started to move from pure computational assistance and drawing to design decision-making. New languages

and faster computation can make more powerful design assisting tools possible such as expert systems, intelligent CAD systems, and integrated design systems in some domains. The use of computers opens a new horizon for design engineering. The optimization, computational analysis, and computer-aided drawing are not new topics any more. What are also needed to enhance the design process are those design systems which can integrate as much knowledge as possible from different aspects or life stages of the proposed product into the design process and help make better design decisions.

Based on the following review of the above-mentioned systematic design methods and some computer-based design methods, the advantages and weaknesses of the existing methods can be used to propose a design information system implemented in an object-oriented design environment in which the design process is closely related to the design functions in order to improve the design quality and cost efficiency.

2.3. Systematic Design Methods

As a systematic design method, the Guideline VDI 2221 [1986] deals with the generally valid principles of design independent of a specific branch of industry. It defines those design activities and stages and results which, because of their logical nature and usefulness, provide a general approach in practice. This overall guideline also aims at summarising and ordering the wide variety of design methods which have arisen in the past few years as a result of work carried out in research and practice.

The basis of the systematic approach to design is the application of problem-solving techniques to the process of design. The process of solving problems represents a permanent relationship between goals, planning, execution and control, linked by decisions. The model of the systems approach divides the development of a system into life phases progressing from the abstract to the concrete. The model also contains a strategy for solving problems which is, in principle, applicable to all life phases. To elaborate the life phases and problem-solving process, VDI 2221 defines the life phases of a system as the following, from abstract to concrete: the planning, preliminary study, system development, system production, system installation, system operation and the system replacement. For each of these life phases linked in a linear fashion, the problem-solving steps are sequenced as: problem analysis, problem definition, system synthesis, system analysis, evaluation and decision, and implementation. It is important and customary to make use of repeated cycles in which the different steps are processed several times where each cycle brings the solution closer to the stated requirements.

The strategy of the systematic approach lies in moving from the abstract to the concrete; and from the most important to the less important. The design process, as part of product creation described in the previous paragraph, is subdivided into general working stages making the design approach transparent, rational and independent of a specific branch of industry. The overall approach is divided into seven stages correspondingly producing seven results. Depending on the task, either all of the stages

or some of them need to be complete, with some stages being repeated as necessary. In practice, individual stages are often combined into design phases which assist the overall planning of the design process. Such a combination into phases can differ depending on the industry or company, and also according to the concepts involved. Figure 2.1 shows the systematic design process by the VDI standards.

In order to make clear how this method accomplishes the design process, it is useful to look into the detail of the several stages:

Stage 1 is necessary to clarify and define the requirements of the task requested by the customer or the product planning department. It includes: collecting all the information available and discovering where there are gaps; checking and supplementing external requirements; adding specific company requirements; and defining and structuring the task from the point of view of the designer. The result is a specification which can be established independently of any solution. This list of requirements (specification of behaviours/attributes) is an important working document which should accompany all subsequent stages, and which should be constantly reviewed and kept up-to-date.

Stage 2 consists of determining functions, first the overall function and then the most important subfunctions (main functions) to be fulfilled by the product or system being designed. The classification and combination of these subfunctions into structures

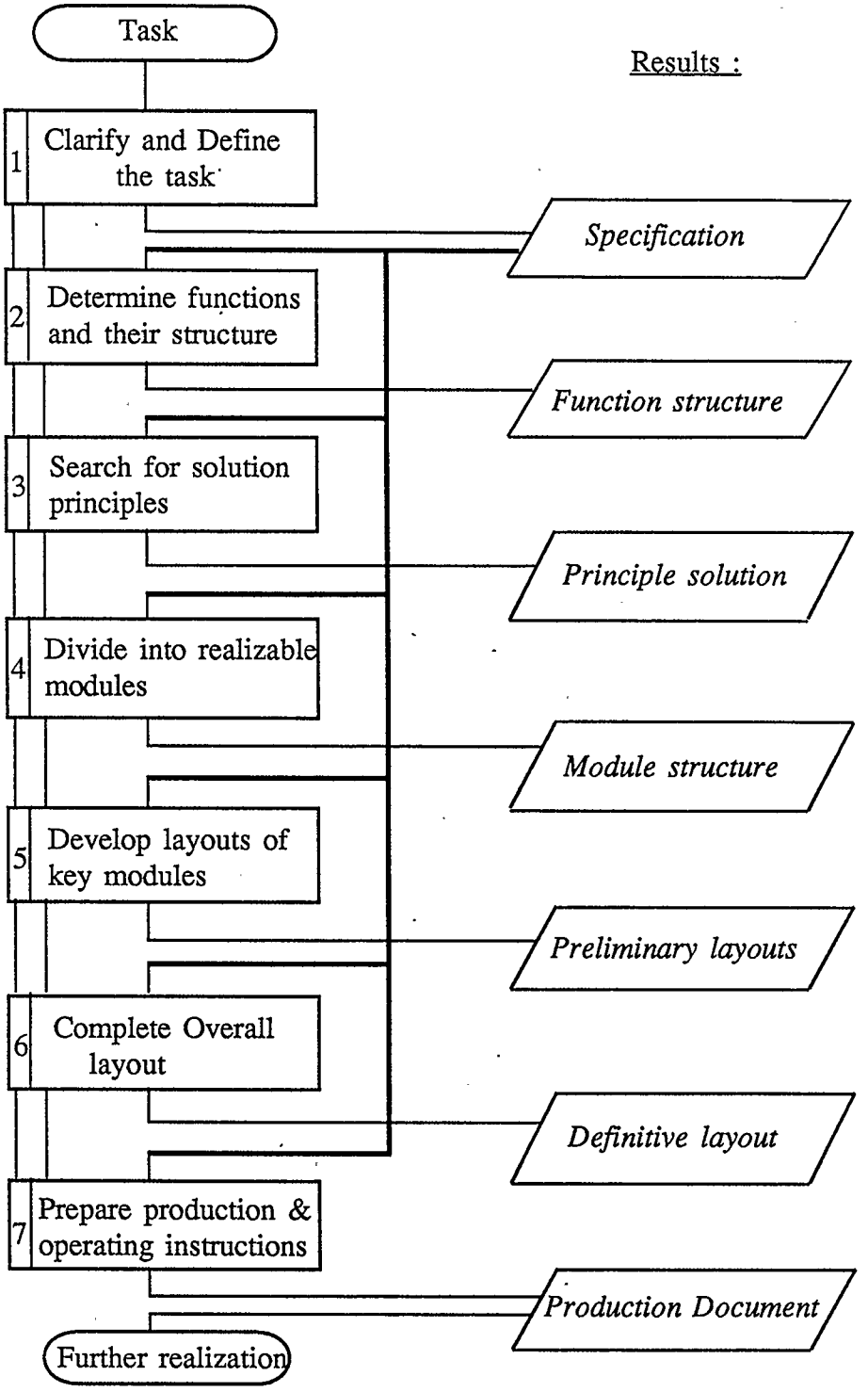

Figure 2.1. Systematic Approach to Design [VDI 2221]

form a basis for the search for solutions for the overall function. The result is one or several function structures.

In Stage 3 a search is made for solution principles for all sub-functions, or initially for the most important sub-functions of the function structure. In the case of mechanical systems, such embodiment features include, for example, the geometry, the motion and the type of material. The solution principles discovered for sub-functions must subsequently be combined in accordance with the overall function structure. In so doing, further subfunctions may become apparent, and these in turn make possible the realization of the certain effects or solution principles. The result of this stage is a principle solution which represents the best combination of physical effects and preliminary embodiment features to fulfil the function structure.

In Stage 4, the principle solution is divided into realizable modules, before starting the complex and time-consuming process of defining these modules in more concrete terms. This results in a module structure which, in contrast to the function structure or principle solution, provides a preliminary indication of the breakdown of the solution into the realizable groups and elements which, together with their links (interfaces), are essential for its implementation.

Stage 5 consists of developing the layouts of the key modules. The level of refinement of the geometry, materials and other details should only be pursued as far as

to allow the optimum design to be selected. The result of this stage is a set of preliminary layouts for the key modules, which can be represented as scale drawings, etc.

In Stage 6, the preliminary layouts of the modules are completed by the addition of further detailed information about the assemblies and components previously not included, and by the combination of all assemblies and components. It is often possible to define those modules not included in Stage 5 by selecting standard or commercially available items. This stage produces a definitive layout containing all the essential configuration information for the realization of the product. The main forms of representation of the design are scale layout drawings, preliminary parts lists, instrumentation flow charts, etc.

In Stage 7 all the final production and operating instructions for which the design department is responsible are prepared. This stage thus overlaps with the proceeding one. The result of this stage is a set of product documents, in the forms of detail and assembly drawings; parts lists, and production, assembly, testing, transport and operating instructions.

It is important to note that the stages do not necessarily follow rigidly one after the other. They are often carried out iteratively, returning to preceding ones, thus achieving a step-by-step optimization.

While this systematic design approach has wide enough coverage of the design process, it inevitably prolongs the design process of the product design. Because the method is in a step-by-step fashion, the design knowledge is not uniformly integrated all along the way from the initial stage to the final stage. The necessity of design knowledge integration is hampered by the lack of consistency in information types. At the same time, the long design cycle has very slow responses to the new design information such as new materials, tools, methods, etc. The worst case is when the product design reaches the later stage and the designers are justifiably reluctant to go back all over again to make corrections unless the change is critical. This will finally incur the loss of quality and cost preference. It is noted that the iterations are among each of those stages, which needs a lot of special effort to prevent the information explosion. The elimination of some unreasonable solutions at earlier stages is not always easily achieved.

2.4. Computational Approaches

During the design process, a designer transforms an abstract functional description of a device into a physical description that satisfies the functional requirements. In this sense, design is a transformation from a functional domain to a physical domain.

Many researchers tackled the mechanical design problems from the above point of view. S. Finger [1989] raised a transformational approach to mechanical design using

a bond graph grammar concept. The long term goal of the research is to create a transformational strategy in which the design specifications for a mechanical system can be transformed into a description of a collection of mechanical components. To realize this goal it requires formal representations for the behavioral and physical specifications of mechanical system as well as formal representations for the behaviours and the physical characteristics of mechanical components. Because the interactions of the components are important in the synthesis strategy the representation of the behaviours of mechanical components must be linked to the representation of their physical characteristics, which is actually the modelling of the relationship between form and function of the components. Finally, a strategy is needed to transform an abstract description of the desired behaviours of a device into a description that corresponds to a collection of available physical components. The result is a component database which contains graphs and form-behaviour relations for a limited class of design, like bevel, spur, and worm gears. But the work is preliminary and must be expanded in many directions before it can prove the approach is valid for a larger class of mechanical design.

S. Srinivasan, et al [1989] used a partitioning and constraint-guided search method to transform the functional requirements into more specific forms of a component in the preliminary design. Specifically, a generic design template has been created as a tool to structure information to facilitate the problem-solving in three different example domains. Information in each implementation has been partitioned as hierarchical levels

of abstraction related through constraints. Function identifies the top level design goals to reduce the search involved for feasible solutions. Goal-directed level search, driven by the design application and top-down refinement, reduces the number of possible alternatives. The commonalities extent in the domains have to be represented as design goals at three levels of abstraction in the design template. Similar frame-based knowledge representations with inheritance hierarchies and mixed reasoning have been developed for the KEE[1]-based implementations in each domain. Distinctions among the domains have been modeled as low level slots in the frame hierarchy.

The generic implementation template for preliminary design has *n* levels of abstraction to represent an arbitrary number of the design. However, in this research the method only used three levels of abstraction. Applications are in the preliminary design of mechanical springs, composite laminates, and expert systems. The implementation of templates in different domains have frame-based representation with inheritance and class hierarchies. Constraints are used to generate feasible alternatives to each design decision wherein each of the alternatives becomes a candidate sub-goal to be evaluated with reference to the goal state represented by a particular design decision. At the top-most level, the function for which the design is required is used in identifying the top level design goals. This considerably reduces the search involved in terms of computational effort since all that is involved is obtaining a match of the application slot value of different alternatives with the current functional specification

[1] KEE is a knowledge-based system.

and all matches become feasible solutions to the design goals.

A computational approach to conceptual design used in the domain of fastener design by K. Ulrich, et al [1987], hypothesized that most new designs are derived from knowledge of existing designs. They identified a special case of this process and called it novel combination. By describing a fully implemented program which designs novel mechanical fasteners, they explained how knowledge of existing devices can be represented and used. This work is important for establishing a functional understanding of computational design. The objective of this work is to use a computational approach to obtain a better understanding of engineering idea generation. The basis of the method is the relationship among the functions and the structure of the design. A functional description consists of a set of functional attributes which might be characterized by a truth table, a performance curve or a set of verbal phrases. The structure of a device is its physical form. A structural description consists of a set of structural attributes, which might be communicated through a drawing, a model, or a physical implementation of the device itself.

It is noticed that there is a fixed framework for the form of the solution in applying novel combination. The difficulty of the design task can be measured by the degree of rigidity of the design framework. So if a design problem does not have a relatively easy framework, this method will be very difficult.

From the above research, it is generally recognized that the design process is a transformation process from the functional domain to the physical domain. However, this transformation process is not well characterized nor understood for mechanical systems. The difficulty arises because mechanical designs are often composed of highly integrated and tightly coupled components where the interactions among the components are essential to the behaviour and cost of the design. So converting a single behavioral requirement to a single component is often both impractical and infeasible. Each component may contribute to several required behaviours, and a single required system behaviour may involve many components. In fact, most mechanical components perform not only the desired behaviour, but also many additional unintended behaviours.

Although the above research demonstrated different methods to perform the bridging of functions to forms of mechanical systems or components, there is still a need to explore a more general representation of this process for wider application domains. A specific method is only useful in a range of well formulated problems, if a design problem is not easy to be formulated, then the use of the above design methods will be limited. On the other hand, functions, as inherent design basis, should be analyzed in depth. More elaborate functional relations can reveal in-depth knowledge about the intrinsic structure of the design. Hence, the design process will be made more transparent by a clear layout of functional structure. Similar to the analysis of design functions, the related physical domain should also be characterized into a suitable format

in order to accomplish the transformation process.

2.5. Expert Systems Approaches

Expert systems, often referred to as knowledge-based systems as by Rynchener [1988], provide a new tool for the design process by allowing a designer to express design knowledge in a form that computers can understand and inference for decisions thereafter. The human can read and improve the expert knowledge, while computers can aid in exercising and applying it, achieving at least partially the results that human experts do. Often expert systems serve as partners on a complex design project. The expert systems used in design processes mainly concentrate on the following topics:

(1). synthesis, the creation and development of alternative designs;

(2). representation of nature of design expertise, as tools that can enhance the designer's decision-making;

(3). integration of existing tools into intelligent, cooperative frameworks; and

(4). the use of graphics interface with built-in knowledge about the design being configured.

The application examples below reveal to some extent the way expert systems are used in helping the design processes.

M.A. Wright [1988] used an expert system on the selection of axial fans. The

user interface is in menu-driven form and it requires a small number of simple inputs in order to arrive at an optimized selection from a database of commercially available fans. The menu options permit the designer to accomplish the design in several ways, ranging from prediction of gross characteristics of a fan to be designed specifically for an application, to the selection of the best commercially available fans for the situation. Acceptable ranges of fan performance for selection and acceptable scaling procedures and limitations are drawn from the experience of experts in fan selection.

M.W. Long [1988] discussed an expert system to aid and evaluate the design of the mechanically fastened joints and make fastener selection recommendations. The system uses a backward chaining mechanism provided by the shell to ask questions concerning the component parts of a mechanically fastened joint from the user. Based on the answers and the rules contained in the knowledge base the system makes recommendations and provides the updated changes in the current joint design criteria. It can also provide the designer with the knowledge of reliability, maintainability, producibility and costing in order for the design to be complete and acceptable.

N. Ramchandran [1988] presented an expert system approach in the design of mechanical components. Basically there are two parts in this system: (1). MEET, and (2). DPMED. MEET follows the principle: Design = Refinement + Constraint Propagation. The primary task of the MEET system is to refine a mechanical design specification into component modules such as gear-pairs, belt-pulley, shaft-bearing, etc.

The constraint propagation distributes design choices made in one particular part of the design through the entire design.

There are many expert systems being used in different design problems. As stated above, expert systems usually are useful in a narrow domain of design, but as design problem gets more complex, wider knowledge scope should be considered, then an integration of these knowledge from different aspects of the design stages is necessary. At the same time, a suitable way for the representation of knowledge is also needed. Only the rule-based style of knowledge is not enough to deal with huge amounts of knowledge. There is a trend that the object-oriented programming will be the future promise to cope with different kinds of design problems.

2.6. Object-Oriented Approaches

Object-Oriented Design methods have already been widely used in many areas. The concept of this method is based on the four properties: hierarchy, inheritance, encapsulation and polymorphism. The first two properties are mostly used in the establishment of the design concept while the others help to enhance the programming efficiency in implementation. To model a design system in a certain domain, the designer has to find out the interrelationships among the functional requirements and the final form of the designing product, and also among the parameters, attributes, and so on. Accordingly the hierarchical and inheritance relations among the design attributes

can be established. With the object-oriented programming tool, the designer can solve a problem more closely to the natural human thinking process. Object-oriented programming systems have inherent appeal to designers since these systems can allow them to look at the design problem as a collection of objects with their attributes. If a design problem can be organized as some set of objects linked with information which are inter-related through a series of rules and constraints, then this problem is well suited to object-oriented programming application.

S. Akagi, et al [1988] described an expert system in their paper which is developed for engineering design based on object-oriented knowledge representation concept. The design process is understood as determining design variables and their relationships which compose a design model. The design model is represented as a network in the computer system using the object-oriented knowledge representation. The system built with the above concept provides the following capabilities:

(1). flexible model building and easy modification;

(2). effective diagnosis of the design process;

(3). supporting method for redesign;

(4). a hybrid function with numerical computations and graphics; and

(5). applicability for various design problems.

This system focused on a ship design process. The functions of the system which is encoded in Lisp are combined with Fortran programs for graphics and large

scale numerical computations. In the design process, the individual design knowledge elements are modularized and represented as objects. To determine the values of the design variables, a well-known message-passing procedure is introduced. For example, when the value of a design variable is to be determined, a message to request its determination should be passed to the corresponding object in which the algorithm for calculating its value is described. If the values of the other design variables are needed in the calculation, some similar messages are passed also to the objects corresponding to them. On the other hand, when the value of some other variables must be corrected in this procedure, the values of the other related design variables are corrected automatically by the message-passing function tracing in the reverse way of the above message-passing. All the design variables or parameters are grouped into classes which have hierarchies with each other. The knowledge element is usually acquired from a design handbook or the designer's expertise.

T. Yokoyama [1990] presented an object-oriented and constraint-based knowledge representation system for design object modelling. In that paper an object model represented as a set of objects is not a mere data structure but an active entity which works to solve design problems. Knowledge representation provided in the system, based on the object-oriented paradigm, makes it possible to describe constraints in a declarative form. A class hierarchy is represented with "*is-a*" links and "*includes*" links. The problem-solving mechanism of the system is based on constraint satisfaction techniques. Constraints are declared statically and can be added to objects dynamically.

An object has a function to keep its state satisfying given constraints. The system was implemented using ESP language on a PSI machine.

The present Object-Oriented Design mainly applies to relatively simple problem-solving by representing either the design knowledge or the design processes. For more complex design problems such as large software system designs, information management system designs, etc. the object-oriented programming language will demonstrate its more advantages. In Chapter 6 a detailed discussion will be made on the use of this object-oriented programming application.

2.7. Concurrent Design

Concurrent design can be considered as part of concurrent engineering which is generally recognized as a practice of incorporating various life cycle considerations into the early stage of design. The life cycle considerations include not only the product's primary functions but also its cost, manufacturability, reliability, serviceability, disposability and so on. Concurrent design seeks to combine the concerns of marketing, production, field service and performance-oriented design into one integrated procedure. The concept of concurrent design is that the design stages should take more of these x-abilities into account in order to improve product quality and reduce the cost and development time.

One approach to concurrent design is called Strategic Approach to Product Design (SAPD) suggested by James L. Nevins [1989]. This method is based on the integration of design for assembly and design for fabrication. Although each of these methods is not new, once they are integrated there are many goals to be satisfied at the same time. Therefore, there are too many interactions and trade-offs in the integrated process when each decision is made. To determine which of the many conflicting goals and rules to apply, SAPD provides the opportunity to formulate the strategy and deal with those trade-offs at the best time during the design process.

Concurrent design can be realized through small systems to a certain degree. M.B. Waldron, et al [1988] proposed an expert system which is divided basically into two parts: building-tree and travelling-tree. Through the former a designer can create or modify an existing tree structure. Building-tree allows the designer to add rules, condition options and constraints for any node. The travelling-tree function allows a user to travel a previously created tree in top-down fashion. Hence the designer can check on the spot whether all the knowledge and solutions have been entered correctly. This system also aims at building up an integrated environment which contains the knowledge of the different aspects of the product life cycle such as manufacturability, maintainability, reliability, etc. This to some extent can be considered as a practice towards the concurrent design.

In addition to the above concept of concurrent design, there are many computer

programs which are considered as contributions to a certain aspect of concurrent design. Their roles can be generally classified as several groups as suggested by Ishii [1990]:

(1). CAD environment that allows continuous access of the design;

(2). simulation of the process and sensitivity analysis;

(3). design assessment programs;

(4). on-line advisory during design; and

(5). concurrent design of product and process.

Each of these groups is useful to the design process in a specific area, but an integrated system which can include several aspects of life cycle considerations is still yet to come.

K. Ishii [1990] also proposed an approach to concurrent design: Compatibility Analysis. The idea is to simultaneously evaluate a candidate design from multiple viewpoints. A "round table" design model is presented in which multiple reviewers with various expertise study a candidate design. Each gives their own comment about the compatibility between the design and the life cycle value for which he/she is responsible. Each expert uses an adjective qualifier: excellent, very good, good, poor, bad and very bad. They do not have to be adjectives since the qualifiers are eventually mapped into a [0,1] measure. The key here is that some compatibility issues are absolute design rules, i.e., definitely not permitted, or absolutely good, while some others are not so extreme. The qualifier "poor" indicates that the compatibility is bad and the design is undesirable, but if other constraints dominate, then the expert will accept the design.

To realize a concurrent design system, an enormous amount of work has to be made as to the knowledge representation, organization, communication and management, etc. At present, some small systems have been implemented such as those computer programs mentioned above, but the development of a full-scale concurrent design system has not been seen to emerge. With the further advance of the computer technology and human understanding about the scenario of design process, this kind of system may be implemented in the future.

2.8. Summary

This chapter has reviewed a variety of currently available and potential design methods. From this review, it is noticed that each design method has its own advantages and drawbacks. Besides, a lot of problems in terms of information integration, knowledge representation, and computer implementation have been revealed. In the following chapters some prominent problems will be analyzed and accordingly, some specific approaches are elaborated to support the problem-solving of the design process.

CHAPTER 3. STATEMENTS OF DESIGN PROBLEMS

3.1. Introduction

This chapter presents some perceived design problems referred to in the previous chapter and explores ways of approaching them. From this discussion it is shown that some of the problems related to the design process are related to time and cost requirements as well as lack of flexibility. These problems can be improved in a concurrent design environment where the product can be developed and evaluated with the presence of designers and manufacturing experts.

3.2. Design Problems and Discussions

3.2.1. Description of existing design procedure

As discussed in the previous chapter, most traditional design processes demonstrate a sequential nature of problem-solving. The design process progresses by pursuing the specifications set in the beginning of the design process. Design decisions are made in a step-by-step procedure. Within each step and among several steps there often exist iterations in order to make an optimum decision to satisfy the goals of each step. The

design process as a whole is often an isolated process because the manufacturing considerations can only be made after the design is finalized. There are many difficulties with this design process. Some of these problems may include the difficulties of controlling cost, reducing development time, and maintaining the functional requirements of the original design. In the following discussions some reasons behind these problems are explained and the ways to approach them are explored.

3.2.2. Description of cost evaluation process

In this competitive world product quality and cost are increasingly vital to the survival of an industry or a company. As cited earlier, up to 70% of product life cycle cost is defined during the conceptual design stage and furthermore 85% of the cost can be determined by the time that the design process is completed. From Figure 3.1, it is noticed that the product life cycle cost approximates to an exponential curve with respect to the time within the life cycle. In the early design stage the cost commitment increases very quickly, which means that the conceptual design phase is very important compared with the rest of the realization of the design. It is also shown that the rate of cost commitment change decreases with respect to the life stage growth. This means that as the development and production stages advance it is getting more difficult to make any change in the original design because for example, the manufacturing facilities have already committed to the first design. Because the early phase of the design process bears most responsibility in deciding the product cost it has the greatest potential to make

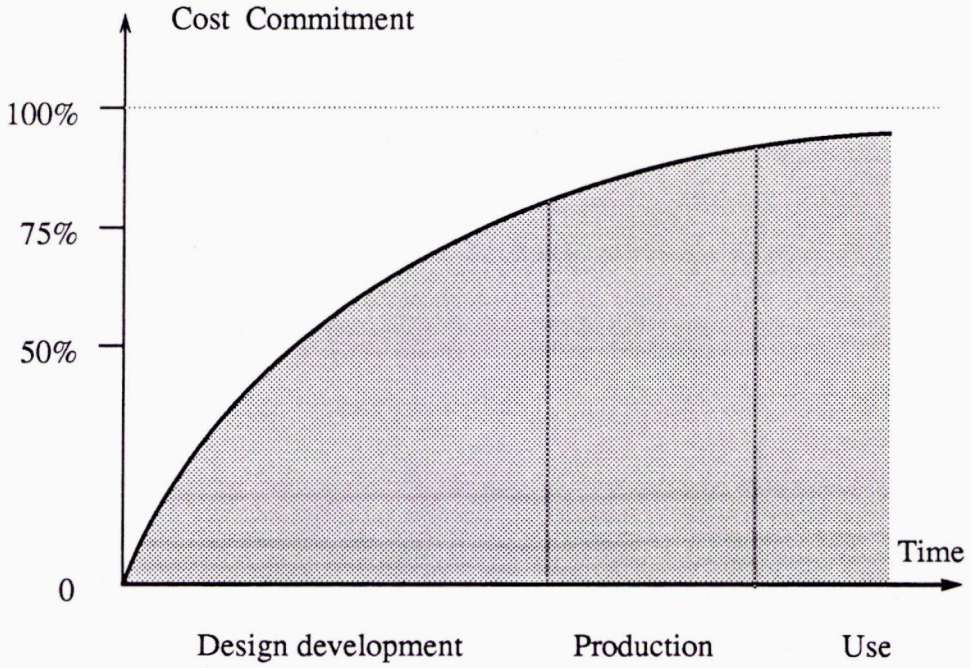

Figure 3.1. Product Life Cycle Cost Commitment [N.R.C.]

savings. Therefore the cost considerations should be elevated to the same importance as the product performance in the early design process to ensure competitiveness.

While the early design stage is important in determining the product cost, the structure of the traditional design process does not facilitate the consideration of cost. Instead, the cost can only be evaluated after the detail design is made, or even after the production plan is made by manufacturing experts. Therefore if the cost is then found to exceed the goal, it may be too late to make major changes in the design. From this it is seen that the traditional sequential design process can not effectively control the product cost at early design stages.

3.2.3. Product development time

Rawling [1991] states that Japanese industry is 12 months faster than its U.S. counterpart in producing the same product. This indicates that U.S. industry will have a great loss of market opportunities since the competitive advantage is seriously reduced.

The product development time is largely dependent on the speed of the design process and of the determination of the manufacture. The systematic design method is comprehensive in providing guidance to solving new design problems but it involves many steps before the completion of a design. There may be many iterations within the design process in order to make decisions. For the design process alone, accessing of design knowledge and information from different remote sources also needs much time because they have not been integrated into the design process. Besides, the linear nature of this method makes it necessary to increase the length of the design process in order to solve a more complex problem.

In addition, the traditional product development process is based on the structure that the design department is separated from the manufacturing department. Only after the design is finalized, can the manufacturing experts get involved. If the manufacturing stage needs changes on the product, the design has to be redone. This incurs an additional time requirements. The same situation may happen between the design and other life stages such as maintenance, marketing, etc. If these departments find that the

design does not meet their requirements, the design also has to be modified.

3.2.4. Design flexibility

Another aspect which often affects the above time and cost problems is the design flexibility. The design flexibility is the ability of a design process to quickly respond to and easily adapt to any changes affecting the design. The design process has to be adaptive to requirement changes and previously-made decisions should also be easily modified without incurring much time delay. For example, if a manufacturing facility has changed or there are more efficient tools available during the progress of the design development, the information base involved in the design process should be up-dated. Therefore, the design process should quickly respond to these changes and modify design decisions accordingly. Also, the decisions made previously should be easily changeable. If the design process does not have the flexibility to respond to the requirement changes, designers are usually reluctant to change the previously-made decisions.

3.3. Summary

In the above discussion the problems related to the traditional design process have been elaborated and the expected characteristics of the design process have been made clear. In order to improve the design process so that the related problems can be approached, the following context will suggest some possible methods.

3.4. Closure

It has been suggested that the above design problems can be improved through pursuing concurrent design approaches which are discussed by Bedworth [1991], DeLorge [1992], Nevins [1989]. To support the concurrent design concept an enhanced information communication tool should be available to access the knowledge from participating engineering domains. It is noticed that the different domains involved in the concurrent design process usually have different languages. The only common language among all these parties are the design functions. The functional descriptions of design intention represent an uniform language which provides a common understanding of the product. Therefore, in the following chapters, a function-based design approach will be presented.

CHAPTER 4.

FUNCTION-CENTRED INFORMATION STRUCTURE

4.1 Introduction

This chapter presents a function-based approach to design to deal effectively with some of the design problems described in Chapter 3. The basis of the approach lies in enhancement of communication among different knowledge domains and represents a concurrent approach to design. The implementation of the design approach includes the recognition of information domains which are labelled Function, Environment, and Embodiment.

To facilitate the communications among different knowledge domains in a concurrent design approach which was believed to effectively deal with the design problems recognized in the previous chapter, this chapter presents a function-based design approach within a FEE design information structure.

In the following a type of routine design is presented to indicate that the design process can be thought of as an information transformation process: analysis of function is followed by generation of forms. Then the design form-function relationships are elaborated to show that the knowledge from different domains can be connected; the

design functions provide a common basis for transmitting design information among the different domains. Because the design process can be overwhelmed by a wide variety of knowledge and information either from existing sources or from emerging new technologies, a Function-Environment-Embodiment information structure is suggested to allow classification of design related information and knowledge into three separate domains. In this way it is suggested that communication and information management can be improved. Features and characteristics of the FEE structure are also discussed in the later part of this chapter.

4.2. Design Process

Traditional design processes typically employ a series of procedures to solve a design problem. The most significant stages in the series of procedures can be thought of as problem definition, solution search/evaluation and design implementation. Problem definition transforms the design needs along with some necessary constraints into a set of functional requirements, based on the design environment. The solution search stage includes all the activities starting from conceptual design (in which functional analysis plays a major role) to the transformation of the functional requirements into physical forms using available knowledge and existing facilities such as standard components, tools, and materials. Design implementation consists of integrating all forms of design embodiments based on functional requirements and finalizing the detailed documentation of the design. It is realized that there are three kinds of information involved in the three

major design stages: the information about the existing knowledge in the design environment, the information about the design functions and their relationships with design forms, and the information about the design embodiment which is the final form of the design.

It is noticed that the connections between design functions and design forms thread through the transformation process from design needs to a final embodiment. Therefore it is useful to explore the relationships between the design forms and functions so that the progress of the design process can be explicitly represented by them.

4.3. Design Form-Functions

According to Suh [1990], the essential activity in the design process is to transform the functional requirements (FR's) in the functional domain into the design parameters (DP's) in the physical domain. To approach the design from the functional perspective, the objectives of the design must be determined in terms of functional requirements. Then to satisfy these functional requirements, physical embodiment characterized in terms of design parameters must be created. In this case the design process is to relate those FR's of the functional domain to the DP's of the physical domain. This design process can be simply illustrated as a transformation process in Figure 4.1, where the DP's in the physical domain are chosen to satisfy the FR's in the functional domain. The purpose of the design process is therefore to bridge the two

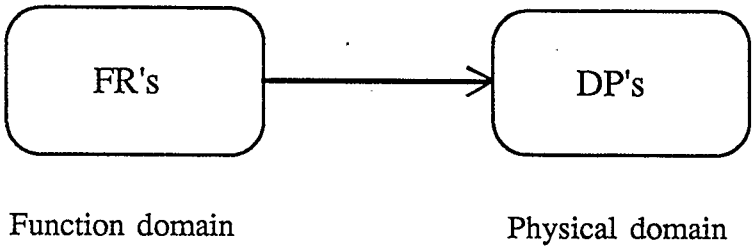

Figure 4.1. Design process as a transformation from one domain to another.

FR's: Functional Requirements; DP's: Design Parameters.
Design process is to transform the FR's defined in the function domain to DP's in the physical domain.

domains and transform the FR's in the functional domain into DP's in the physical domain.

The design process usually begins with the establishment of FR's in the functional domain to satisfy a given set of needs and ends with the creation of a physical entity which satisfies those FR's. This physical entity is usually represented by a set of design parameters (DP's) as suggested above. By defining the design objectives in functional terms the essential purpose of the design can be captured. The design definitions which are not in truly functional terms tend to pre-define the solution domains and therefore limit the possibilities of producing several independent solutions which may consequently result in losing the opportunity to generate better alternative solutions. To express the design objectives in functional terms the vocabulary for the "functions" should indicate 'what'(in terms of behaviour) is expected instead of 'how' it is achieved. For example, in designing a pipe joint, the functional definition should express the requirements for 'containment', 'reliability', 'cost', etc., instead of a 'boltedJoint', or a 'weldedJoint'. A specific approach defined in the latter case may not be an optimum solution to achieve 'what' is expected but because it is assumed in the design definition it becomes implemented.

Because the design process is manipulated in terms of the form and function of all the design entities, the characteristics of the form and function should be explicitly demonstrated in detail so that the design process is clearly seen as the transformation

process from the design functions to forms. Besides, the forms and functions of the design entities are the design information carriers when they are viewed from the information scenario. Therefore, if the design process is considered an information processing then the primary effort should be devoted to exploring the relationships and characteristics of the information used to describe form and function.

In describing the design process it was stated that the first step is to define the problem to be solved in terms of FR's; that is, the designer should establish the functions from the needs that the final product or process must satisfy. This is clearly one of the most critical stages in the design process. This definitional step requires insight into the problem and a knowledge base encompassing issues related to the problem. This suggests that the design recognize the true needs of a design and the possibility of realizing them. If the definition of FR's does not represent the needs, the design process will create an unacceptable solution either by not meeting the needs or by exceeding the needs with extra cost.

4.3.1. Definitions of Form and Function

In the above discussion, the word Function or 'functional requirement' means the desired behaviour of the proposed product. Functions are the designers' characterization of the perceived needs for a product whether it is a device, process, software, system, or organization. In other words, functions of a product or device are the ultimate goals and

satisfy the perceived needs by transforming certain energy, material, and information from one state to another state. The overall design purpose is usually expressed as overall functions expected from the product or system. Following the definition of the overall functions a series of design activities can be performed in terms of establishing function structure and decomposing it into sub-levels which will eventually be satisfied by some physical forms.

Form or the 'design parameter' of a design includes all those physical attributes that all together generate the desired behaviour. The word 'physical' is used in some places to differentiate form from function. Design forms in this context may include all available physical sources which may contribute to the completion of the design. The design form can range from all existing hardware such as parts, materials, equipment to design knowledge such as methods, regulations and standards. Even though the design forms sometimes refer to the 'physical' hardware entities such as parts, materials, the representation of the forms referring to these entities is still in a software format such as data and rules. Design activities are therefore supported by the existence of all these forms.

4.3.2. Determination of FR's in a Design Process

There are two considerations in defining the functions as suggested by [Suh]. Firstly, functions should be defined as independent requirements that completely

characterize the design objective for the specific needs. In other words, functions must be independent of each other, and thus each function can be stated without considering other functions. Secondly, the number of functions should be kept to minimum. If some functions are dependent on other functions then they are considered redundant. This situation should be avoided because it will complicate the design process without adding any benefits.

To successfully accomplish a design need by a function-to-form transformation process, it is clearly important to define the functions appropriately. The choice of the FR's depends on the way in which the designer hopes to satisfy a set of needs. For example, to design a joint which will be used in a pressurized dry air system, the FR's may not need to include 'corrosion resistance' which otherwise must be considered in a corrosive chemical piping system. To define an appropriate set of functions the information about the required product attributes must be present to help the decision-making. This information refers to the forms already existing in the design environment. The definition of FR's is an important step in the design process because the final design is unlikely to be better than the set of FR's that it was created to satisfy.

Moreover, when the goal is to create design solutions that have not previously been in existence, FR's must be defined in a solution-neutral environment, i.e., the functional space. They should be defined without any preconceived notion of a physical solution in mind. To define FR's without preconceived notions about what may work

best, the description of the FR's may be include attributes such as: simplicity, efficiency, light weight, and reliability, etc. If there are any functional constraints, they should also be defined purely in the functional domain.

In many cases the establishment of an acceptable set of FR's may require an iterative process. The iteration may involve the entire life cycle of product development (including the complete sequence of design-manufacture-testing-use-disposal) which is time consuming and costly - both financially and in terms of lost opportunities. Just like the design process itself, the most desirable iteration cycle, next to "no iteration", is the reiteration at the conceptual stage of the design process itself. Once the conceptual design is completed the expected performance of the resultant design product can be compared with the original perceived needs of the product. If they differ then an improved set of FR's can be established without incurring the cost of making and testing the hardware and/or software.

4.3.3. Hierarchies of FR's and DP's

The design decision-making is based on the form-function relationships. In order to perform the decision-making process, the characteristics of the form-functions have to be to searched. There are following characteristics for both the functions and forms, as suggested by Suh [1990]:

(1). FR's and DP's have hierarchies, and they can be decomposed;

(2). FR's at the *i*th level cannot be decomposed into the next level of the FR hierarchy without first going over to the physical domain and developing a solution that satisfies the *i*th level of FR's with all the corresponding DP's. That is, the designer has to travel back and forth between the functional and physical domain in developing the FR and DP hierarchies.

It is critical to identify the most important FR's at each level of the functional hierarchy by eliminating secondary factors from consideration. Otherwise, the information explosion cannot be handled. Therefore, the design procedure should allow consideration of all functions at every level simultaneously and it should make use of the hierarchical nature of the FR's and DP's. For example, in a joint design problem each level of functional hierarchy has several possible solutions which may all satisfy the upper level function. Decisions must be made as to which function should be decomposed first to continue the decomposition process.

As shown in Figure 4.2 (a), the 'PipeJoining' as a top level function can be satisfied by a number of alternate physical functions such as Welding, Coupling, FlangeJoining, and AdhesiveJoining. The 'FlangeJoining' function at the second level of the hierarchy can consequently be decomposed as in Figure 4.2 (b) because it is important to consider other factors in satisfying the overall 'Pipe Joining' function. Those other factors may include the functional attributes such as 'permanency', 'simplicity', etc. Other functions at the same level may be decomposed later if more solutions need to be

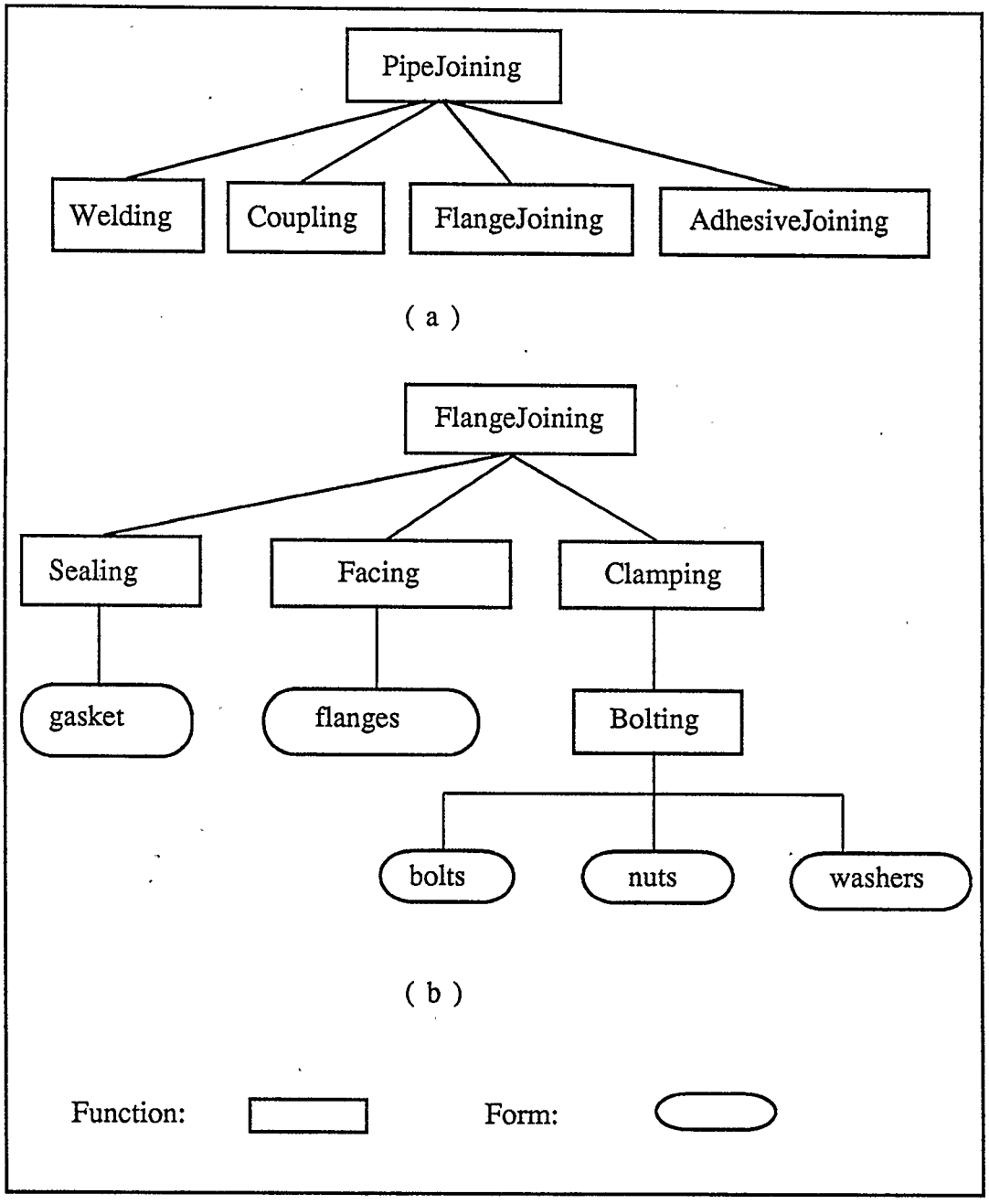

Figure 4.2. A Joint Function Hierarchy

A function can be decomposed into a hierarchical structure. Form can also be decomposed into a similar structure. Function hierarchy at lower level cannot be decomposed further until a form is used to satisfy the function.

found to compare with each other. There exist different relations between any two neighbouring levels in the function hierarchy. These relations are observed as either a 'is-a' relation or a 'part-of' relation. Between the top level and second level, as well as the third level and fourth level, the relations between the upper function and the subfunction/forms are 'is-a' relations, which means that each subfunction/form 'is-a' kind of implementation of the upper function. Similarly, the 'part-of' relations existing between second and third level as well as the fourth and fifth level indicate that each subfunction/form is a 'part-of' the implementation of the upper function. These relations will be used in the implementation of the function structure to connect different pieces of design information.

The function hierarchy reveals the disposition of design knowledge in different levels. Function at some level can be connected with even detailed information about forms such as material characteristics, heat treatments, stresses, and so on. Therefore, the information in the design process can be integrated through function-oriented representation of the different aspects of knowledge involved in the design. The cost, space, skills, accuracy, etc. should all be considered by structuring the functional relations.

4.3.4. Information Integration through Form-Function Relationships

Form and function relationships are multi-relationships. This is particularly true in the mechanical design process. For example, a PipeJoining function can be

accomplished by several joining methods such as weldedJoint, boltedJoint, or adhesiveJoint as shown in Figure 4.3 (a). That is, a function can be satisfied by many forms. Which form should be used depends on the evaluation of all the other functions. In a similar way, a form can satisfy many functions at the same time, though some of the functions are useful, others may bring undesired side effects. Figure 4.3 (b) shows some functions provided by a BoltedJoint which is a form of connection. While the non-permanency and sealing functions are usually desired the strengthReduction caused by bolt holes implied with this kind of connection may affect the part's strength adversely. By considering the many-to-many relationships in matching the functions with the forms, more information about the characteristics of the form-functions can be integrated.

For a certain perceived need, an acceptable set of FR's is not necessarily unique. There may be many other sets of FR's which can all represent the same perceived need. Corresponding to a set of FR's there can be many design solutions all of which may satisfy the same set of FR's. It is the time constraints that often limit the multiple solution search, which consequently may prevent a better solution from being considered. When the original set of FR's is changed either from the re-definition or from the customer, a new solution should be found.

Based on the functional analysis depicted in Figure 4.2, it is seen that several joining methods are available to satisfy the upper level function 'PipeJoining' and the transformation process from function to form may not yield unique result. In other

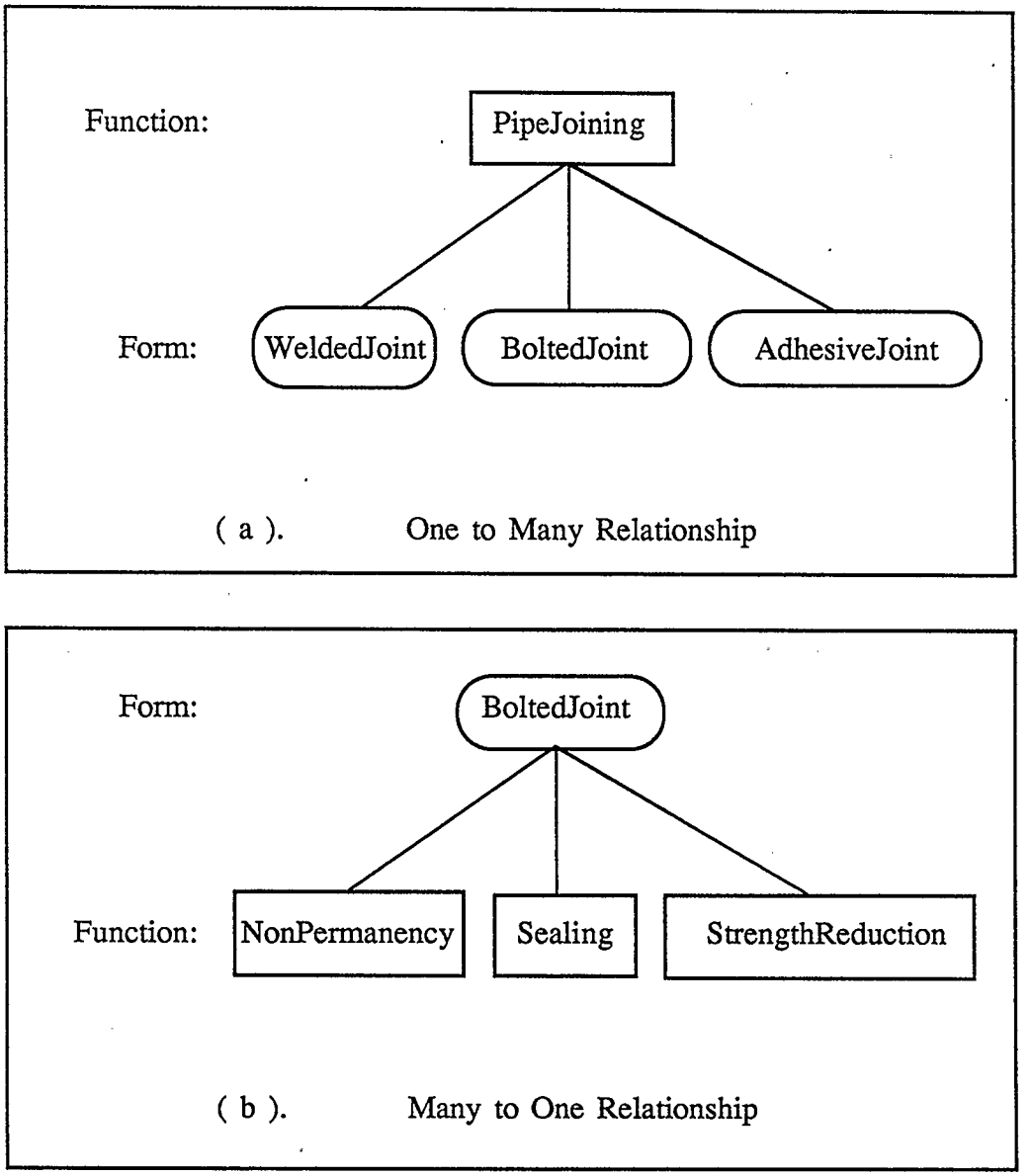

Figure 4.3. Multi-Relations between Form and Function

words, more than one design solution may result from the generation of the DP's that satisfy the PipeJoining function. Evaluation of these solutions is necessary to choose a best alternative considering the design constraints such as cost, time, and company production capability.

To satisfy a particular design function, the associated subfunctions should be searched. These subfunctions may extend beyond the immediate design domain or extend to fields such as design theory, material, manufacturing technology, maintenance, and existing products. The function hierarchy can in this way connect the knowledge from these different aspects. This makes possible the integration of several aspects of design information. For example, Figure 4.4 shows a function-form hierarchy of the 'InstallCost'. It is noticed that the different costs at second level are determined by the forms at the lower level which in this case may be some names for the ToolType. To further find the parameters for the forms such as ToolType and SkillLevel, connections need to be made to the existing data about this kind of knowledge. Therefore, the design information on the cost is integrated from the function domain to the form domain.

4.4. Design Information Domains

From the form-function hierarchies discussed above it is noted that the design functions and the forms which are used to satisfy the functions are not explicitly organized in a predetermined way so that whenever there is a need to decompose a

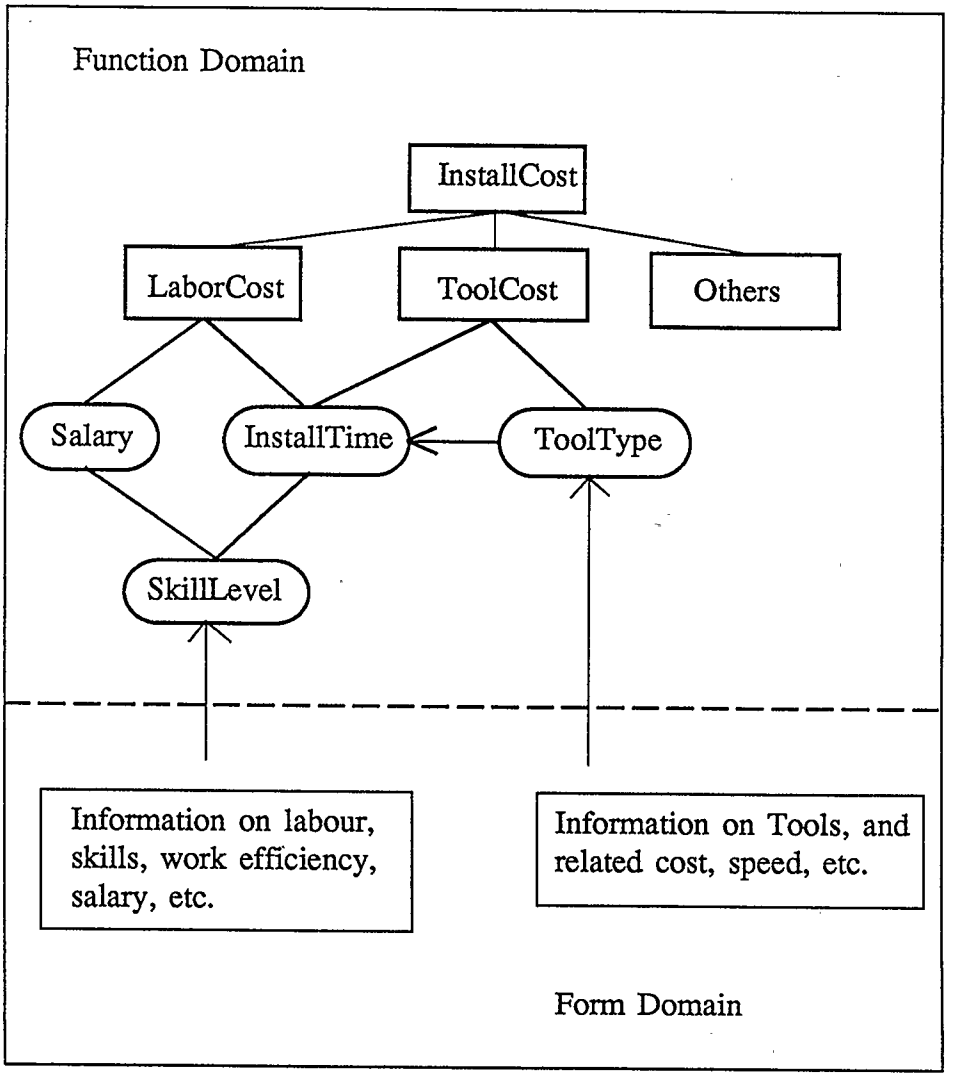

Figure 4.4. Cost Analysis in Function Domain Accesses the Information from the Form Domain

function or to look for forms to satisfy some functions the search route is created to go to a certain knowledge base and find the related information. Also, the design results coming from the function-form matching process are not specifically recorded. Therefore, it is necessary to have a certain mechanism to organize and manage the design information and results.

A Function-Environment-Embodiment design information structure is designed to classify the design information into three distinct domains. The three information domains as shown in Figure 4.5 are named as Environment domain, Function domain and Embodiment domain respectively. The Environment domain represents the real world environment, in which all the design information such as existing parts, materials and tools reside. Design problems are initiated in this domain from some perceived needs within some part of the environment. The Function domain contains the statement of desired/achieved characteristics and relationships of design functions and forms which are necessary to guide the solution search and decision-making in transforming the functions to forms. The last domain is the Embodiment domain in which the design results in various formats are stored. Three domains in the tripartite structure interact with each other during the design process to retrieve and record the design information. Design information organized in this paradigm will prove to be easily accessed and more useful to the design decision-making.

To further clarify the FEE structure, the Environment domain is where the design

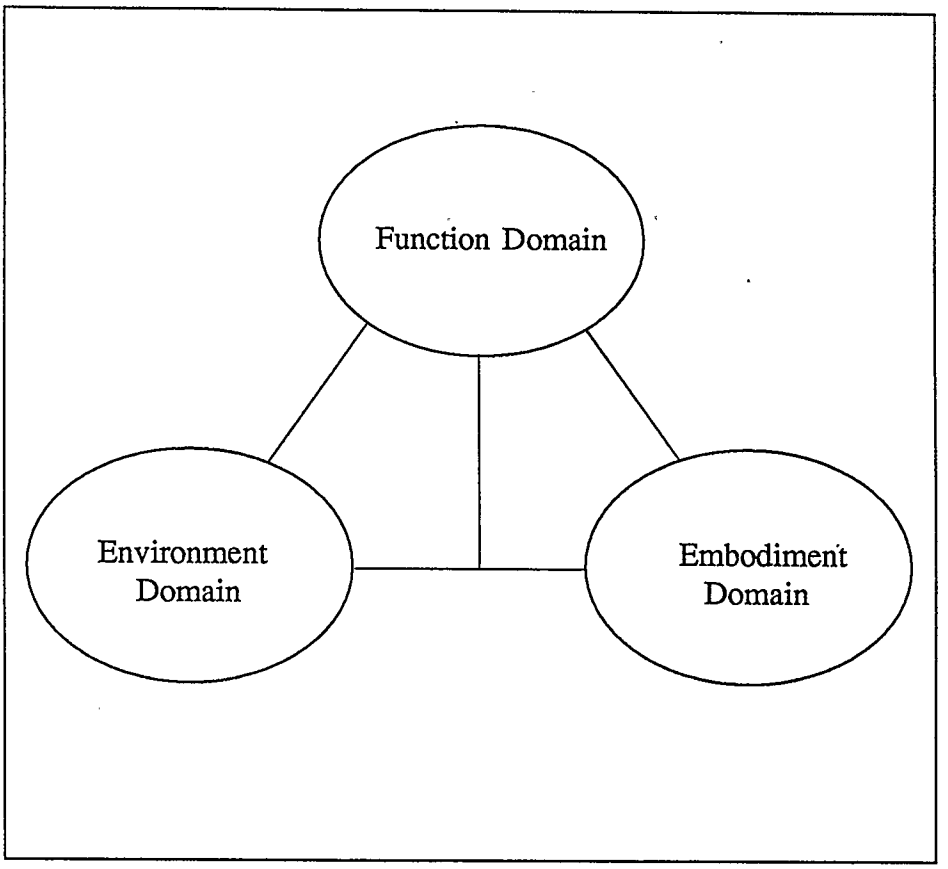

Figure 4.5. FEE Design Information Tripartite Strucutre.

Three Information Domains interact with each other during the design process.

problems come out of and where the design definitions are made based on the information contained in this domain. A perceived need is usually recognized from the environment and therefore a design problem emerges. This domain as observed from above discussion includes all the tasks in the initial stage of the traditional design process such as task clarification, and specification. These two activities are accomplished based on the existing knowledge and information. The Function domain is used to record all aspects of information about the relationships of design functions. After the design task is defined from the Environment domain the Function domain is structured to organize the various functional aspects. The Embodiment domain is where the various design documentations reside. Based on this general description an in-depth discussion on how each domain contributes to the FEE information structure will be presented.

4.4.1. Environment Domain

It is important to recognize that the starting point of design resides in knowledge about 'that which already exists' and how it is seen to work [Fauvel]. Clearly the modelling of the 'real world' is an enormous task but it will be recognized that for any given problem area only the localized portions of this domain will enter into play. On the other hand there may well exist a large number of aspects of the local environment which impinge on the design process. For example, the local legal environment may be particularly relevant and exist in the form of design code and health and safety considerations or product liability requirements. Other important aspects might include

ergonomics consideration with regard to potential user/operator/maintainers, etc.; availability of potential components and/or manufacturing facilities, etc. The body of knowledge about material and system behaviours as coded in the form of engineering science represents a particularly but not exclusively important segment of this domain.

The most important characteristic describing the content of the Environment domain is the fact that it already exists and to that extent is incapable of being reshaped within the context of the particular design of the project. Thus if the project is aimed at modifying a machine which already exists then if there exist aspects of the machine which must remain as is, these would represent constraints which would be present in the Environment domain.

4.4.2. Function Domain

Whereas the Environment domain exists prior to the undertaking of a design project, the early portions of a design process are aimed at defining the desired outcome of the process in terms which do not preclude any potentially accessible approach. By expressing the desired results in a solution independent form and placing this within the Function domain it is possible to address questions of 'what' is to be achieved rather than 'how' it is to be achieved. The Function domain is intended to span all relevant behavioral attributes of the proposed design. In addition to the prime functional requirements of the proposed design there are usually seen to be adjunct requirements

such as manufacturability, assemblability, conformance with legal and related requirements, and may range to maintainability, reliability, or even recyclability and disposability. These desired attributes may exist in the form of targets or constraints and tend to be defined early in the design process even though in somewhat general or abstract form.

As the design evolves so does the information in the Function domain. The direction of this evolution is from abstraction to concrete or from general to specific. An abstraction hierarchy is presented by Rasmussen [1986] which reflects this evolution. At the highest level is the Function Purpose by which means the objective of the design is defined. This is broken down at the second level into 'Abstract Function' at which level causal structure is elaborated as information, energy, and mass flows. 'General Function' can be identified in terms of control regimes, identifiable standard functions which can be associated with families of hardware (e.g. pump, heat exchanger, transducer, etc.). 'Physical Function' at the next lower level are those functions associated with more specifically identifiable pieces of hardware (e.g. positively displacement pump, air-to-air counterflow heat exchanger, pressure transducer, etc.). As this functional purpose is developed, it is apparent that it is increasingly simple to map the functional element into some configuration of the hardware which would be expected to perform the desired tasks.

The nature of mechanical design, however, is such that ancillary behaviours or

attributes will figure largely in the success or failure of the total system. Under this heading may appear attributes which have traditionally been regarded as secondary to the prime function -- attributes such as manufacturability, reliability, serviceability, and so on. Increasingly it is being recognized that these attributes must be considered as early as possible in the design process therefore these attributes and behaviours are also part of the function space albeit as consequential attributes as distinct from the protogenic ones typically expressed as the Functional Purpose.

4.4.3. Embodiment Domain

The Embodiment domain can be classified as including any information which serves to describe the physical configuration of the object or the system of the objects which is in the process of being designed. It is suggested that within this domain the information will form a hierarchy which is parallel to that outlined above in the function domain hierarchy. Referring to the systematic design method, it will be seen that the design results from different stages are the progressive embodiment of the design. These results can be recorded into the Embodiment domain to keep track of all the design information/knowledge. Since the design embodiment evolves with the design progress from abstract to specifically detailed, the knowledge in the Embodiment domain also forms a hierarchical relationship. Initially the Embodiment object might merely be represented as 'that which will function in the required fashion'. As the Function domain is elaborated and made more specific the Embodiment will evolve through such

forms as schematic diagrams, conceptual configurations, general arrangement drawings, part specifications, manufacturing drawings, etc.

4.4.4. Information Contents in the Three Domains

From the discussion of each of the three information domains it can be recognized that each domain holds different forms of design information. Figure 4.6 shows the variety of design information in each domain. it should be noted that the information in the Embodiment domain grows as the design progresses. At the beginning of the design the information may be sparse and vague but at the end of the design all aspects about the design will be saved as shown in the different categories.

4.5. Characteristics of FEE Structure

The FEE information structure can serve as a design framework. It allows placement and organization of the different information needed in a design process. Integration of the information is therefore facilitated. Furthermore, the role of the information can be recorded for possible subsequent re-evaluation.

By way of contrast with the problem-solving process as outlined in Figure 2.1, the design process can now be examined within the framework of the suggested FEE information structure.

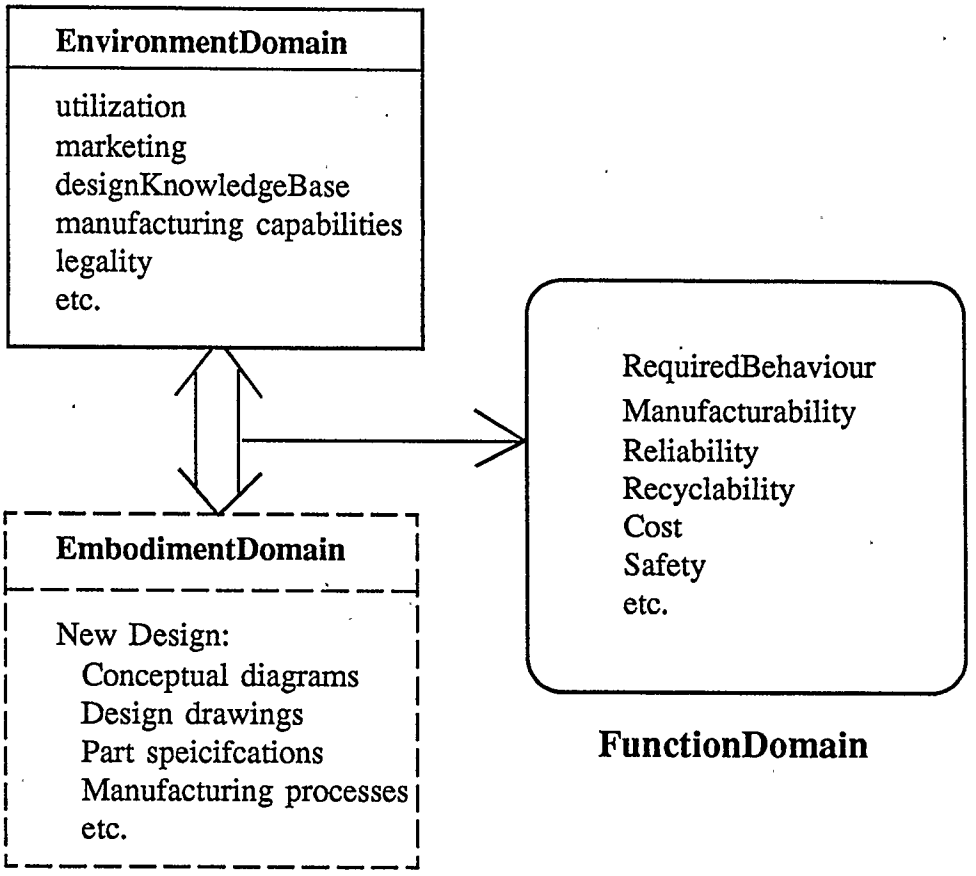

Figure 4.6. Information Contents in the FEE Framework (after Fauvel [1992]).

To begin with, the perception of some needs of a particular aspect of the Environment is seen to give rise to an alternative view of the workings of that Environment: this view is expressed in graphic form at the most abstract level of the Function domain as shown in Figure 4.7 (a). From this functional purpose will follow an abstract Embodiment as shown in Figure 4.7 (b). If at this stage it is possible to adduce the nature of the interactions between the Embodiment and appropriate portions of the Environment then these interactions will be represented within the Function domain as illustrated in Figure 4.7 (c). As the design process advances it is possible to move from abstract to concrete level in a fashion which has previously been described.

At the abstract (early) stages of the design it is generally difficult to establish more than the most vague connections between the Embodiment, the Environment, and the consequential attributes of the design. As the design proceeds (as successive iterations at increasingly specific levels of the information domain hierarchies) it is increasingly easy to ascertain consequential attributes of the design. For example, an assessment of the manufacturability of a product would require that proposed embodiment be examined in the light of the manufacturing options resident in the Environment. The more detailed the Embodiment, the more accurate an assessment of the manufacturability can be determined.

So far in this section the information structure has been seen as an adjunct to the design process of the type shown in Figure 2.1. As such the following benefits may be

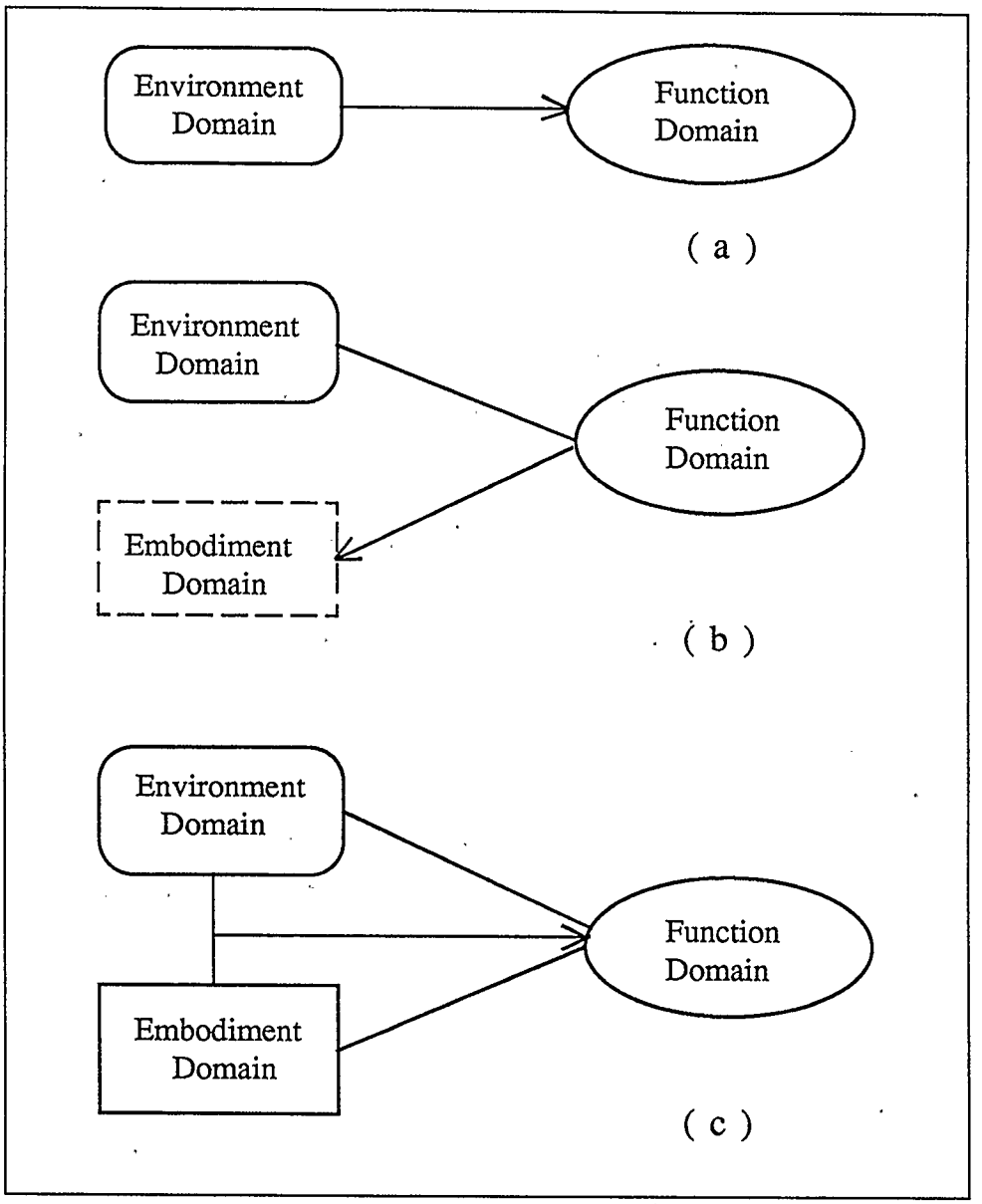

Figure 4.7. Design Progress in the FEE Information Domains. Function domain serves as medium to interact with the Environment domain and the Embodiment domain.

realized:

(1). the role of all information used in the course of the design is explicitly represented;

(2). the relationship between the functional purpose and the remaining attributes can be portrayed in an on-going fashion thus illustrating the involvement of the multiple interests in the design;

(3). a change of any factor used in the course of the design can be followed using the logic trail to determine potential consequences.

4.6. Summary

From the discussion in this chapter, it is clear that the design information should be structured in a way that it can be accessed by the design process easily and effectively. A suggested information structure is the FEE tripartite structure in which the design form-function relationships play a major role in connecting the different information domains. The important advantages of this FEE paradigm is that it provides a design information structure which can connect design information from different sources so that the design process can be expedited. In the following chapter, a joint design-related problem is presented in this FEE structure to demonstrate the working principle of the FEE information structure.

CHAPTER 5. JOINT COMPONENT SELECTION PROCESS

5.1. Introduction

This chapter presents the conventional joint component selection procedures during a pipe joint design process; this is followed by the mapping of the procedure into a FEE information structure. Because the joint is typically designed from existing components the design process is mainly devoted to the component selection from different sources. It is noted that even though the joint design problem may be considered to be a well-defined design problem, much related information is not formulated in a way that the design process can make use of it efficiently. Therefore the designer still has to go through every step of calculation to determine each parameter of all components. In order to choose a component the designer must often go through many standards and data tables which are usually separated from each other and not related in a meaningful way.

After the conventional joint design procedure is presented, the knowledge and information involved in the design will be projected into the three information domains of a FEE design information structure described in the previous chapter. By distinguishing the characteristics of the design information in different domains the connections between them are expected to be made. Therefore the integration of different types of design knowledge and information can be achieved.

5.2. Joint Component Selection Process

Pipe joint design is as widely used as are pipe applications. Present piping design practice usually involves two aspects: one is the selection process of component parts; the other is the installation operation. There exist a variety of industry standards and codes which provide design knowledge of every aspect in a separate and non-related manner. In order to choose a set of components which will satisfy the design conditions the designer usually has to search among many standards in order to make decisions.

By way of example a proposed pipe joint is to be accomplished by using a flanged joint for which the reasons will be seen when the design functions are analyzed in the later discussion. Therefore, in the following context the flanged joint component selection process as usually practised in the industry is presented in detail. A sectional view of a bolted flange joint diagram is shown in Figure 5.1 in which the major components of this type of joint are shown to consist of flanges, gasket, and bolts/nuts. The following component selection process will focus on these three component groups.

5.2.1. Flange Selection

Flange selection is the first step in the selection stage of the joint design process. For large diameter carbon steel flanges there are several class designations to represent the characteristics of the flanges. The class designation may imply the flange properties

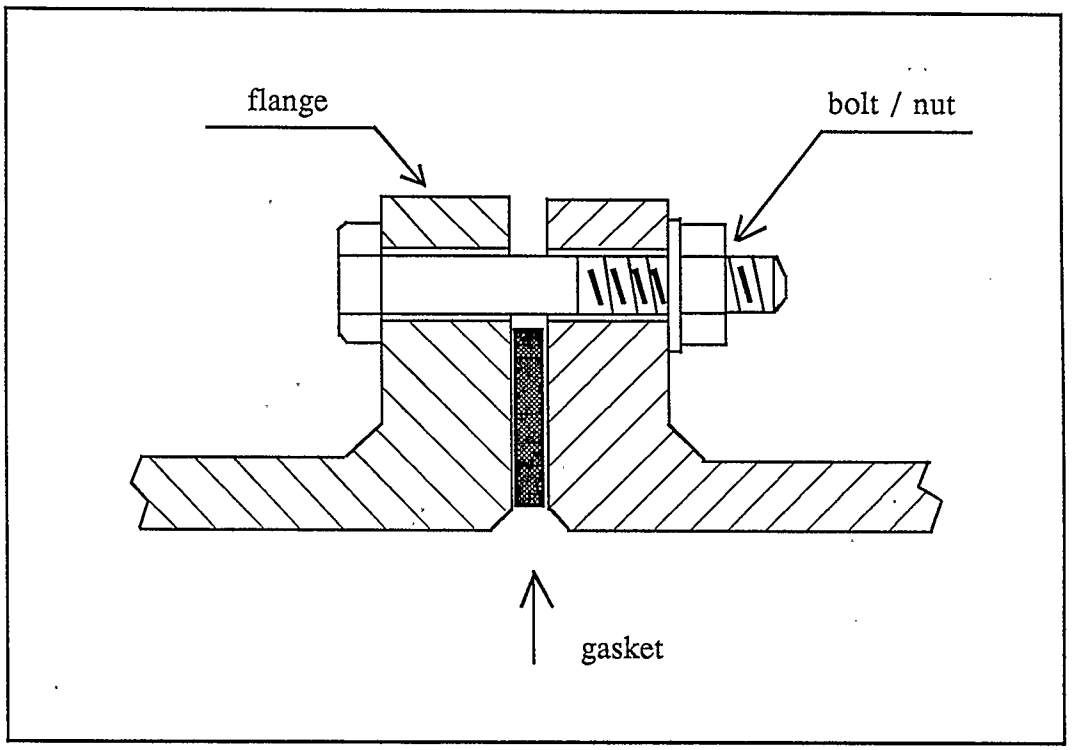

Figure 5.1. Sectional View of Components of a Bolted Flange Joint.

Joint component selection focuses on the three major components which are considered as the flange, gasket, and fasteners.

such as material heat treatment, strength, and hardness. For each specific class, there are a variety of flanges in different sizes. A flange is made to fit with the pipes to be connected and therefore the nominal size of a flange is defined by the Nominal Pipe Size (NPS) which describes the pipe diameter. According to API 605 Standard [1981] the flange for an application is decided by the working pressure and temperature as well as the NPS. Therefore, the selection process will use these parameters as criteria to decide which flange will be considered suitable to the working requirements.

From Table[1] 5.1, it is noticed that the flange class designation (or rating) can be found based on the given expected working conditions such as temperature and pressure. Any value lying between the listed values in the table should be rounded upward to the next higher value in order to achieve a conservative design. For example, if the design temperature is 325 degrees Fahrenheit and design pressure is 430 psi, then the search for class rating should be based on the next higher values of 400 and 635 to decide that the flange to be used should be within the class 300.

According to API 605 Standard [1981], the flanges are categorized into six class designations, which are classes 75, 150, 300, 450, 600, 900, respectively. These classes represent an increasing order of the material properties in terms of strength and related characteristics. Under each of these classes there exists a series of flanges with their dimensional parameters. Flange information under all these classes is listed in Tables 5.2

[1] Tables are listed in Appendix One.

to 5.7 respectively. All the information should be available for the design process whenever it is needed. Because the flange material defined by this standard is carbon steel ASTM A105, normalized or quenched and tempered, the flange material selection is assumed to comply with this. In actual industry application, the first step in deciding a flange may be the material selection, depending on the nature of application area and the working conditions.

Once the class rating is decided a certain type of flange is selected. To find out which specific flange is appropriate for use, the nominal pipe size (NPS) is used. This leads to Table 5.2 which lists all the flanges in different sizes under Class 300. From this table it is clear that for any NPS there exists one specific flange with its configuration and all dimensions; this selection will affect the later stages of design in all aspects. In order to conform with the Standards used in this context, it is assumed that the flange NPS's are ranging from 26 to 40 inches with 2 inch intervals.

5.2.2. Gasket Selection

The process of gasket selection follows the flange selection. Gasket selection is based on two major considerations: material and configuration. Industrial standards often have separate data tables for these characteristics. Configuration directly affects the installation, while the material properties define two very important parameters expressed as the 'm' factor and the 'y' factor which represent different aspects of the strength level

of the gasket.

Gasket configuration can be decided fairly easily because gaskets are usually classified into different classes in the same way that flanges are classified. Therefore the selection of a gasket also starts with a definition of the class designation. For a particular application, gasket and flange cooperate and withstand the same working conditions and therefore the gasket class rating takes the same value as the flange class rating. After the class designation is defined the NPS is used to determine which gasket should be used from a series of gaskets under the same class designation. Table 5.8 and Table 5.9 list all the spiral-wound gaskets designed for API Standard 605 flanges. It is logical that the gaskets are also classified into different classes (class 75 gaskets are not present in the table so whenever the class 75 flanges are used for an application the gaskets in class 150 can safely be used). It is clear from this procedure that a specific gasket will be defined with its all dimensions which will be used in the later stage force calculation and installation considerations.

In order to make the above decision the gasket material has to be defined before the data table can be used. Gaskets are usually available in a variety of different materials. There is a wide range of gasket materials to choose from very soft to very hard depending on the application. There are many rules and considerations to guide the decision-making in selecting a material. In this context, we suppose that there exists a certain process which can determine the gasket material using related knowledge. This

process may be an expert system, a consultation process from a gasket specialist, or a method of retrieving information from many databases. The material should be selected according to the type of industry, working conditions, and environmental concerns. In this design we temporarily assume that the material has already been defined as spiral-wound stainless steel, which is a hard material, depending on the general knowledge that for high temperature and high pressure the gasket material should be strong enough to withstand the required load. Therefore the process can proceed with ease. Once the material is decided the above-mentioned 'm' factor and 'y' factor will be decided from Table 5.10, which was extracted from ASME Code a [1989]. It is noticed that for different materials the 'm' and 'y' factors have very different values. This will affect the later force calculations and consequently some further decisions on the fastener selection.

At this point it is realized that the design process is a series of decision-makings based on thorough understanding about every perspective of each stage. A designer may not have all the in-depth knowledge about every aspect of the design process such as material selection but needs to know where specific information or knowledge can be accessed. If the design system can integrate the necessary information from different sources then a designer can make decisions much more efficiently rather than spend much time to find out how and from where to obtain the information.

5.2.3. Bolts Selection

It is a design pattern which may not be well recognized that the design process proceeds by first defining the major components followed by the minor components in terms of their physical correspondence or dependence. Compared with the flange and gasket in the joint, bolts are physically smaller and installed onto them. Therefore, they are easier to be accommodated to the other components. For this reason the bolt selection is performed after the other two components are defined.

Two kinds of forces play major roles in the bolt selection process. One is the force to satisfy the operating conditions and the other is the force to effect the gasket sealing requirements. The major one of these two will be used in deciding the strength level of the bolts. The definitions of the two forces (W_{m1} and W_{m2}) in the following text are taken from Pipe Joint, Part 1 [IMechE, (1986a)].

5.2.3.1. Minimum Bolt Load to Satisfy Operating Conditions (W_{m1})

To satisfy the operating conditions the minimum bolt load, W_{m1}, has to be sufficient to resist the hydraulic end force exerted by the maximum allowable working pressure as well as to maintain the gasket compression load sufficient to assure a tight and lasting seal. The gasket factor is a function of the gasket material and configuration. Table 5.10 shows the gasket factors for different materials. To proceed with the

following calculations, the gasket dimension and material should be decided in advance.

$$W_{m1} = H + H_p$$

$$= (\pi / 4) G^2P + 2b\pi GmP$$

where H --- Hydraulic end force;

H_p -- Total gasket contact surface compression load;

G --- Diameter at location of gasket load reaction;

Calculated as:

where $b_0 \leq 6.3$ mm, G = mean diameter of gasket contact face;

where $b_0 > 6.3$ mm, G = outside diameter of gasket contact face less 2b.

b_0 --- Basic gasket seating width;

b ---- Effective gasket width [ASME 1989 a];

where $b_0 \leq 6.3$ mm, $b = b_0$.

where $b_0 > 6.3$ mm, $b = (\text{sqrt } b_0) / 2.0$.

m ---- Gasket m factor;

P ---- Design pressure.

The values of b and b_0 can be obtained from ASME Code a [1989]. The value of G can be calculated by the above definition after the gasket dimension is obtained.

5.2.3.2. Minimum Load to Effect Gasket Sealing Purpose (W_{m2})

Before a leak-free joint can be obtained it is necessary to seat the gasket properly by applying a minimum initial stress (under atmospheric temperature condition without the presence of initial pressure). This is the stress required to deform the gasket into the irregularities of the flange surfaces and it is governed by compressibility of the gasket material. The minimum initial bolt load required for this purpose, W_{m2}, is:

$$W_{m2} = \pi b G y$$

Where b --- Effective gasket width;

G --- Diameter at location of gasket load reaction;

y --- Minimum design gasket seating stress (y factor).

5.2.3.3. Minimum Total Cross-selectional Area of Bolts

The total cross-selectional area of bolts, Am, required for both the operating condition and gasket seating is the greater of the value for A_{m1} and A_{m2}:

$$A_{m1} = W_{m1} / S_b$$

$$A_{m2} = W_{m2} / S_a$$

$$A_m = \text{Max} (A_{m1}, A_{m2})$$

Where: S_b --- Allowable bolt stress at design temperature;

S_a --- Allowable bolt stress at ambient temperature;

A_m --- Minimum required total cross-sectional area.

In the above calculations, W_{m1} is for the operating condition so that S_b is the stress at design temperature. While the W_{m2} is for gasket seating at the environment condition, the stress at ambient temperature S_a is used. Bolt selection is made such that the actual total cross-sectional area of bolts is not less than the minimum required area A_m.

Before using the values of S_a and S_b which are shown in Table 5.11 extracted from ASME Code b [1989], the bolt material has to be decided. This is a complex problem since many behaviours of the bolt have to be considered. For example, if the joint is to be used in a high temperature, high pressure environment, and it contains corrosive fluid, the bolt material should be able to withstand this working condition and have high performance properties such as corrosion resistance, and higher tensile strength, etc. To satisfy all the requirements which can be significant to the application, in-depth knowledge about the material properties is needed. Therefore it is necessary for the designer to consult a specialist in material technology in order to make a good choice if the designer does not have enough knowledge on this aspect. Table 5.11 shows that several high alloy steel bolt materials have different strengths at different temperatures. From the table the values S_a and S_b will be obtained if the bolt material is decided.

This is to suggest that this design process needs to be integrated with other information or knowledge necessary for the design decision-making. If knowledge of material behaviours are available through the design information system, the designer can more easily make decisions regarding the design problem. The advantage of integrating the specialist knowledge at this stage in the design process is to ensure the quality of decision-making without unduly lengthening the design process.

5.2.3.4. Bolt Selection

Given the minimum total bolt cross-sectional area, the cross-sectional area of each bolt can be determined:

$$A = A_m / N$$

where: A --- cross-sectional area of a bolt;

N --- Number of bolts defined by flange for the joint.

Hence its diameter D can be calculated:

$$D = \text{sqrt} (4A / \pi)$$

where: D --- Minimum bolt diameter to satisfy the load requirements.

To select a specific bolt from bolt standards, the bolt diameter D is used as the nominal diameter of the bolt. Table 5.12 lists the available bolt nominal diameters. Bolt diameter increases with a certain interval; in selecting bolts, if the diameter calculated

from stress requirement is not listed in the product data table, the next larger diameter bolt should be used.

All the dimensions of the bolt selected from the standards should be checked against flange hole dimensions to ensure the potential for proper installation. If any discrepancy appears it is necessary to go back the previous stage and modify some parameters then an alternative value can be sought by repeating part of the process again. In this case the bolt material can be selected from a wide range of stress levels while the dimensions are constrained by the flange configuration. So an alternative bolt material may be used in order to effect a bolt diameter change to satisfy the dimensional constraints. Usually the iteration within the design process is a local one so that the iteration time can be kept to a minimum. Only when difficulties are encountered within the local area is an extended part of design process involved to generate alternative solutions. In this case if the bolt selection alone cannot satisfy the dimensional conflict an alternate flange may be considered.

5.3. Some Installation Considerations

Once the components of the joint are decided, the other related problems should be considered. These problems can be from different life stages such as manufacturing and maintenance stages depending on the nature of the design problem. In this case the assemblability is one aspect of the design attributes to be evaluated in order to ensure the

designed joint is potentially assemblable at this design stage.

5.3.1. Assemblability Check

Because the components are physically incorporated together to achieve the required design functions, the dimensions of each component must conform with its neighbouring ones. For this design example the major concern regarding dimensional compatibility is the flange hole size and the bolt diameter. The bolt diameter cannot be larger than the bolt hole size on the flange. At the same time, it also cannot be so small as to allow rotational movement of the flange. The length of the bolt is also a concern for installation. Both of these parameters should be checked against the values defined by the flange.

5.3.2. Tool Availability Check

Tool availability can be checked when the parts and the installation space are defined. Other conditions may also affect the tool selection in terms of installation speed, cost, etc. For the bolt installation, the head shape of the bolt may vary from square to dodecagonal and therefore the tool grip should be able to accommodate the different head shapes. On the other hand, the installation space may limit the applicability of tools. If the wrench grip is not adaptable to different shapes of bolts and the space available prevents the minimum turning angle then the wrench may not be used. Therefore the

design process should also consider situations in this aspect so that the installation does not encounter difficulties or incur much cost due to the need for special tools and processes.

5.4. Design Environment Considerations

As suggested in the previous chapter, the design Environment represents a certain type of design knowledge and information. For the joint design process presented above, the knowledge and information involved in the Environment domain are as follows:

(1). design conditions;

(2). design process; and

(3). existing designs and data.

The design conditions refer to the environmental conditions and the working requirements. These requirements are raised according to a certain needs and a certain working conditions. They may be:

(a). the media to be sealed;

(b). the operating temperature ranges of the media;

(c). the operating pressures of the media;

(d). the manufacturing and operating infrastructure;

(e). the safety requirements and cost constraints; and

(f). the life expectancy.

These design conditions set the expected behaviours for the joint to be designed. From

these requirements it is noticed that there is no indication as to what type of joint is suggested for the design implementation. These requirements therefore are stated in functional terms which will be useful in the Function domain to decide what Form of implementation will be suitable.

The information about this design process is usually contained in design handbooks and codes. When this knowledge is not implemented into computers the designer has to follow the design steps as described in the books and therefore the design speed cannot be enhanced. On the other hand, as new design methods and tools are always emerging it is difficult for a designer to keep track of all the new developments if this knowledge has still not been implemented in accessible databases.

The information about the existing designs is abundant. All the existing joint designs of different types constitute the product category, and all the components constitute the component category. But these existing designs are not represented as ready-to-use knowledge which can be used efficiently by a design process. Instead, the design process still has to go through the calculations to find out if the components will work together to produce the desired effects. In addition to these, piles of industry design codes provide an enormous amount of data describing the rules and regulations with regard to the design and use of these products and components. Even though the products already exist, the present design process still tends to design similar products from scratch because the information about existing designs is not organized and represented into some

form which can be used. Therefore the design process involves making repeated efforts which should be avoided if possible.

Figure 5.2 shows a typical selection process during the joint design. The different forms of the existing design information are being accessed by the design process. The first group of data needed after the design requirements and conditions are defined is the existing joint types. This kind of information is saved in the Environment domain. It includes the physical functions such as Welding, FlangeJoining, or Bonding in order to satisfy the design conditions. General knowledge about the functions of these different types of joining should be available to design the Function domain and to decide which type is appropriate. This knowledge usually can be formed as rules to guide the top level matching between the requirements and the general solution. In this case suppose the Flange Joint is selected. Once it comes to the parts selection level such as flange selection, the data which can be used are mostly numerical values which do not explicitly express the functional aspects. Difficulties are encountered here because the data has to be processed into meaningful terms to represent the components before it can be used. Similar difficulties are also encountered during the GasketSelection stage and BoltSelection stage. To access these data is a time-consuming process in which a lot of calculations have to be made to transform the numerical data such as dimensions, 'm' and 'y' factors, or stresses into some meaningful terms such as force, weight, etc.

The existing information on the other hand also places some constraints on the

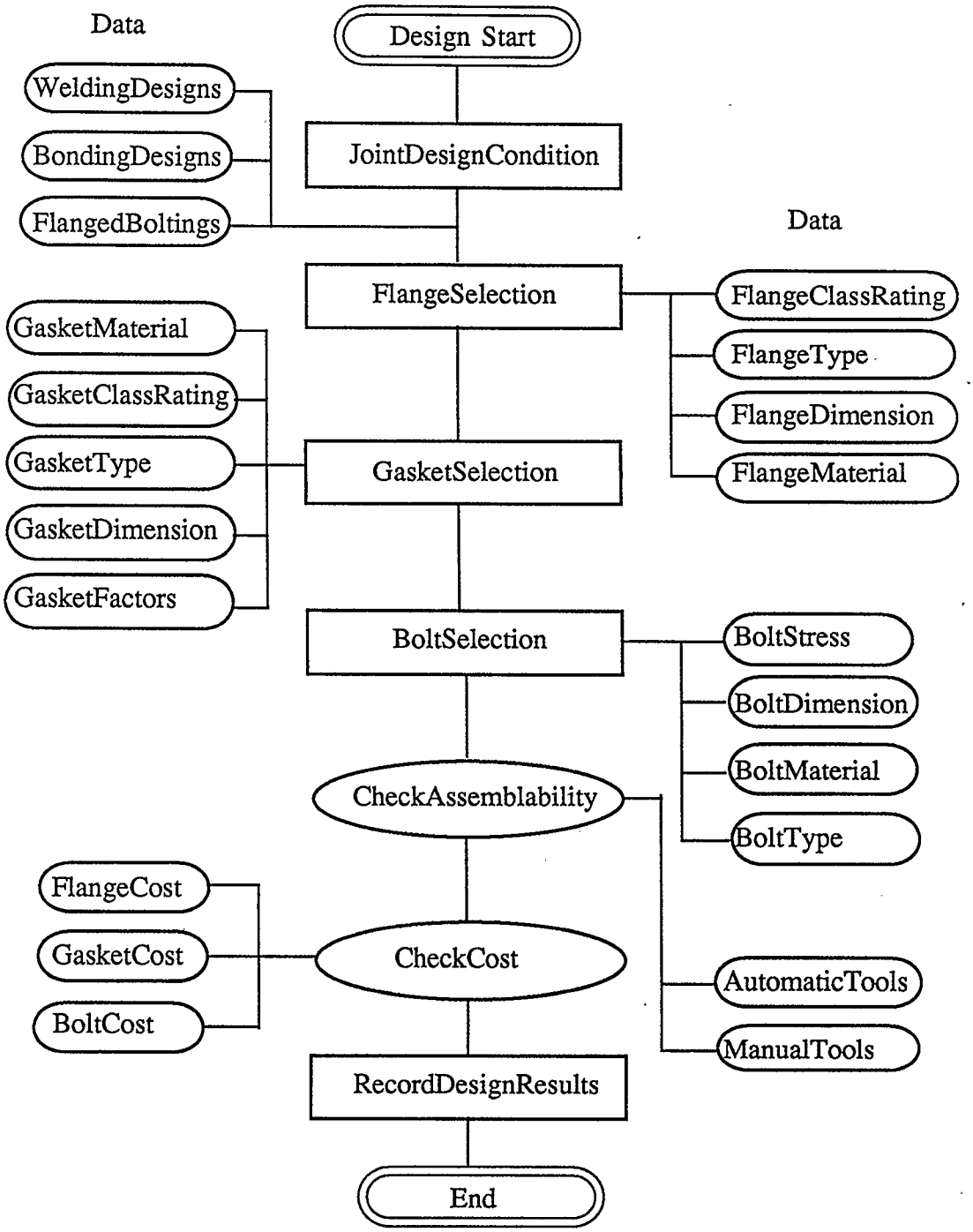

Figure 5.2. Design Process as Data Accessing

design process, such as material availability, tool availability, etc. For example, when the joint is implemented as a bolted flange joint, the related concerns will arise such as the flange characteristics and gasket material availability. The design process needs to access all these data to help the design proceed.

To deal with this design problem in the design information structure, the above information about the design environment has been positioned into the Environment domain. When this information is accessed from the Function domain during the design process, it can be related to the corresponding functions.

5.5. Function Domain Analysis

The Environment domain has been used to present the forms of information involved in a conventional process for the joint design. From there it is noticed that the information about design is scattered around and is mostly obtained from data tables carried in the various design standards. Therefore it is difficult to integrate the information into the design process in a meaningful way because the data from the data tables are not related explicitly to any functional characteristics. For example, the 'maximum clamping force of a bolt' is not expressed in any data table. Instead, what can be seen are merely sizes and stress levels at different temperatures. The whole design process becomes a data 'functionization' process. This process will be explained in the next chapter.

The use of design function analysis is an attempt to relate the components' configurations to their functional characteristics so that the design process can easily use the existing designs by matching the design functions set from the design process with the functional representations of the physical components listed in the data tables.

The Function domain is designed to accommodate all the design functions so that they can be used to guide the search of existing components. On the other hand, the components' descriptions should be characterized into a higher level of abstraction in the form of functional representation. These two parts together can achieve a successful functional matching from the design requirements to design implementation. The knowledge about the functional representation and decomposition of the two aspects should be part of the Function domain.

For example, from the design function point of view the ultimate goal of the design is to prevent leakage, in other words, to provide containment. This functional requirement has to be satisfied by the physical implementation regardless of the form of the implementation. Accompanying the primary function, secondary functions may require the joint to be easily disassembled for piping relocations and also to withstand some high temperature and pressure. To satisfy these requirements the FlangedJoint becomes a favourable choice as shown in Figure 5.3. The information about the FlangedJoint must include these features so that the decision can be made upon matching the required functions with the functions that the FlangedJoint can provide.

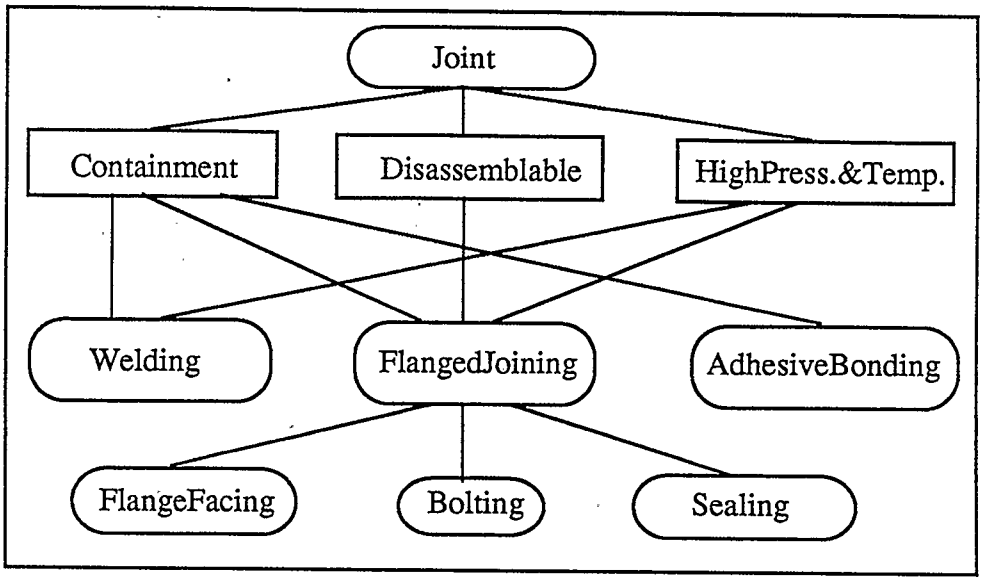

Figure 5.3. Design Function Matching

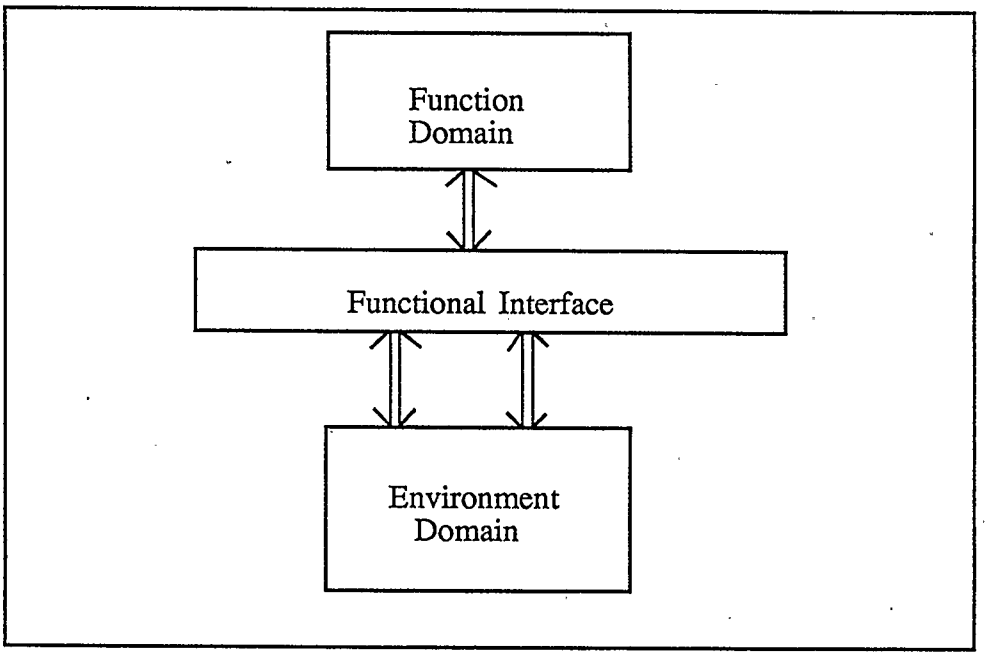

Figure 5.4. Functional Interface Integration

As discussed in Chapter 4, a function can be decomposed to several lower level sub-functions, and this decomposition can be continued until some sub-function has to be instantiated by a physical form. Design matching is complete when all the functions are satisfied in this way. In order to accomplish the design process based on the function matching, Figure 5.4 suggests that a functional interface has to be built to represent the functions of the physical forms known as various existing components. By establishing this functional interface between the Function domain and Environment domain, the design process will be able to easily integrate the knowledge from these two domains.

In a similar way, knowledge on other aspects such as assemblability and cost evaluation can also be formed into functional hierarchies in the Function domain while their counterpart physical aspects in Environment domain can be abstracted into some functional terms in order to be bridged with the Function domain. Figure 5.5 and Figure 5.6 show the hierarchies of the assemblability problem and the cost evaluation problem respectively.

The collection of the different function hierarchies finally contribute to the contents of the Function domain. Through the mapping from the Function domain to Embodiment domain the design solution can be sought.

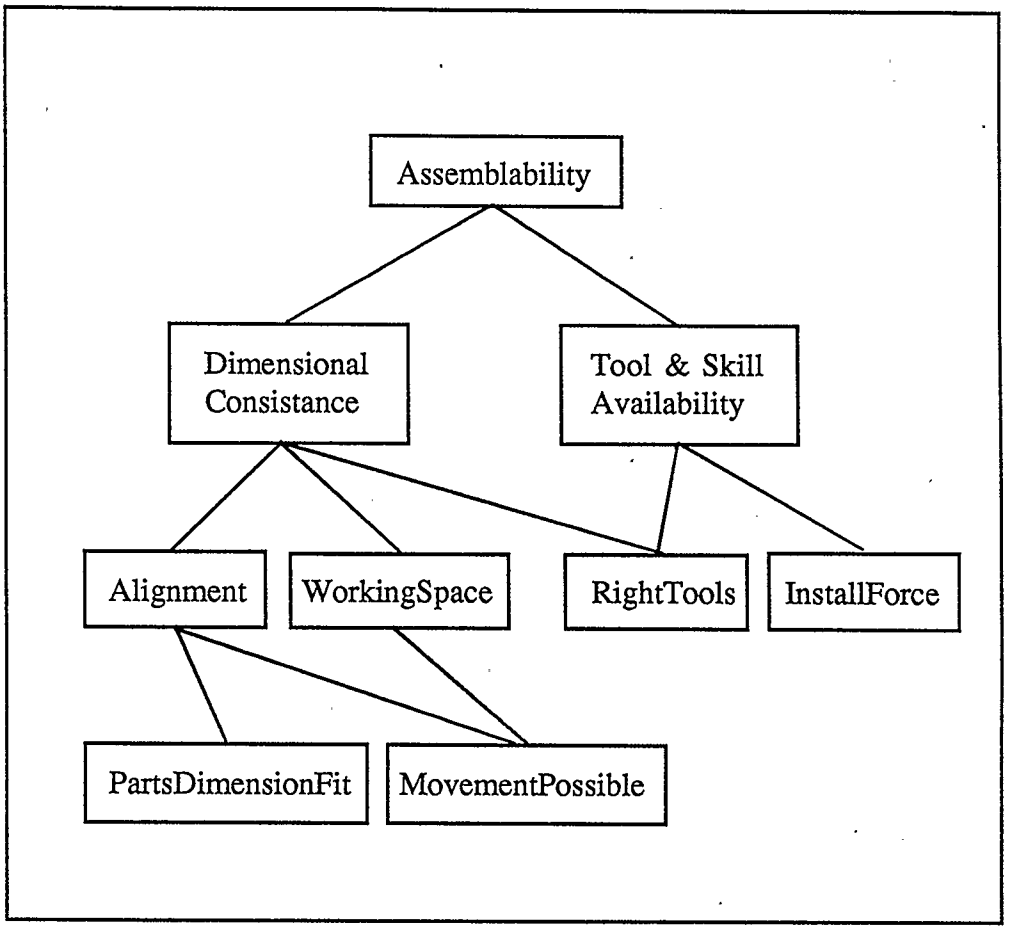

Figure 5.5. Assemblability hierachical relationship

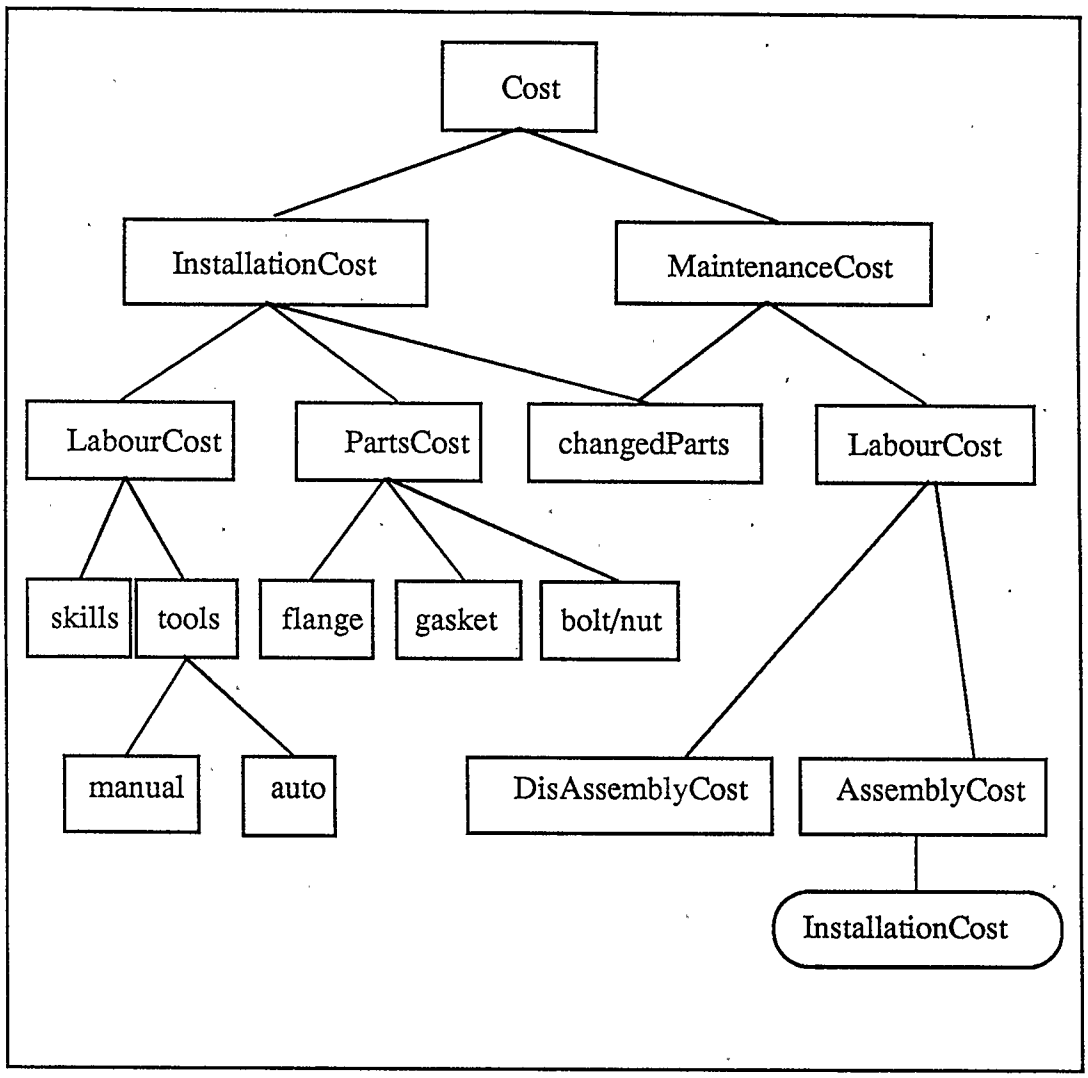

Figure 5.6. Cost Evaluation Hierarchy

5.6. Design Embodiment

When the design results are formed from the interactions of the other two domains, they are logically translated into the Embodiment domain. The design results can be in many forms: the parts, assembly drawings, the system configurations, and the physical specifications. For the joint design concerned in this context the design results are the descriptions of the selected components.

In addition, other related concerns are also presented as part of the final design results. For example, after the components have been decided, the cost model accesses the cost values from each part and the installation process and calculates the total cost of the product. This cost is saved in this domain and can be used to compare with the design cost constraints. The knowledge stored in this domain can be accessed any time after the design is completed.

5.7. Design Process within the Information Framework

The design problem as seen is discussed separately with respect to the three information domains. Design activities actually take place interactively among the paradigm composed of the three domains: Function, Environment, and Embodiment. Since the relationships between the information resided in the three domains describe all the aspects of the design problem, it is helpful to put the design problem into this

paradigm so that the interactions between the three domains can be seen clearly during the design process.

Figure 5.7 shows the structure of the Function-Environment-Embodiment design information framework. Three domains are separate entities but connected by the information accessing channels. The broken line groups the Embodiment domain into the Environment domain, which indicates that the Embodiment domain will finally become a part of the Environment domain because when the design is completed the Embodiment domain will hold the knowledge about a new existing design which forms a part of the environment. The design process itself as part of the environment components is also seated in the Environment domain.

To further explain the design process supported by the FEE structure, the figure has marked several numbers to indicate the progressive steps. (1), Within the Environment domain the design process begins with the recognition of some needs supported by the existing knowledge base. These needs as design requirements initiate the design process. (2), Then the design process formulates the requirements into functional statements and consults the Function domain for solutions. (3), Functional analysis and the function-form matching are made between the Function and Environment domain. (4), The intermediate results from the function to form matching process are saved into the Embodiment domain. (5), Design evaluation within the Embodiment domain evaluates the results and compares them with the existing designs. (6), When the

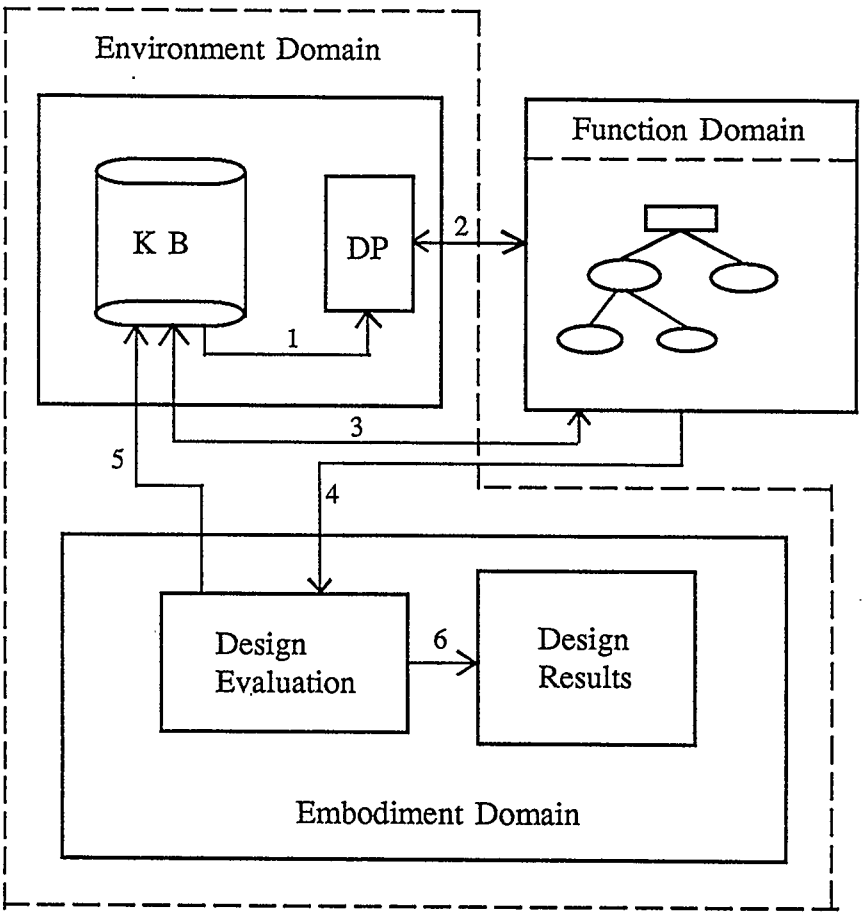

Figure 5.7. FEE Design Information Framework.

Design process starts from the need raised to the Design Process (DP) from the Knowledge Base (KB). DP consults the Function Domain and then Function Domain brings together the function and the forms within the KB and puts the selected forms into the Embodiment Domain. The selected forms are evaluated by interacting with existing KB, the Design Evaluation process finally saves the acceptable forms into Design Results.

results are acceptable they can be recorded as the final design results.

5.8. Summary

In this chapter a joint components selection process is presented and then described within the FEE structure. The design process and the interactions between the different kinds of design information are demonstrated. In the next chapter, this joint component selection process described in the FEE design framework will be implemented in an Object-Oriented Paradigm to demonstrate the advantages of the FEE information structure in supporting the design process.

CHAPTER 6.

OBJECT-ORIENTED DESIGN INFORMATION INTEGRATION

6.1. Introduction

This chapter presents the FEE design information structure which is implemented in an object-oriented paradigm. The object-oriented paradigm is supported by Object-Oriented Programming (OOP) languages which usually have specific properties different from those of structured languages. Pinson [1988] identifies the properties as abstraction, inheritance, encapsulation, and polymorphism as defined by Smalltalk-80 language. The implication of these properties introduced many useful concepts such as information-hiding, code reuse, and easy expansion of a system.

There are some OOP applications used for solving the design problems [Akagi (1988), Yokoyama (1990)], but they mainly apply to relatively simple problem-solving by representing either design knowledge or the some of design processes. This chapter describes the use of the OOP to implement the FEE design information structure in which a specific design problem can be solved. To achieve this goal the design structure should be easily represented as an object concept. The three domains of the design information structure described in Chapter 5 are implemented as three classes called Function Class, Environment Class, and Embodiment Class. An unique way of problem-solving in this

OOP paradigm is simply by passing messages between objects. All actions are achieved by sending messages from one object to another.

In the following sections, an object-oriented problem-solving paradigm supported by a Smalltalk-80 programming environment is introduced. Then the implementation of the FEE information system in the object-oriented paradigm is presented based on the design information of the pipe joint process described in the previous chapter. Finally, the joint component selection process is modelled and executed within the implemented FEE design structure to demonstrate the working principle and the facilitation of the design information interaction within this design information framework.

6.2. Object-Oriented Paradigm Supported by Smalltalk-80

Smalltalk-80[1] is a fully object-oriented language and programming environment. Everything in Smalltalk is an object and every activity happens upon objects. Smalltalk-80 supports four major properties known as abstraction, inheritance, encapsulation and polymorphism. These properties are accomplished by its powerful class hierarchy structure. The basic elements included in the class hierarchy are: classes, objects, instances, variables, messages, and methods.

Smalltalk-80 is based on a large class hierarchy which includes more than 200

[1] Smalltalk-80 is a trademark of ParcPlace Systems, Palo Alto, California.

Classes. Object Class is the superclass of all the other classes in the class hierarchy. New classes can be added into the existing hierarchy by either being an immediate subclass of the Object Class or being a subclass of other classes.

An object in Smalltalk is a component represented by some private data and a set of methods or operations. Conceptually, an object can be thought of as a virtual computer with a memory and a primitive instruction or operation set. The private data and state are its memory that is kept within the object. An object is also capable of computation and it can respond to any of a predefined set of messages.

In OOP, a class is an abstraction that captures the characteristics and operations common to a set of objects. In other words, a class is a description of a set of objects with similar characteristics, attributes, and behaviours. An instance is an individual object that is both described by and a member of a particular class. Logically, an object is an individual encapsulation of some states and operations. Since all instances of a class support the same set of operations, the methods or operations can be physically associated with the class. Only the state or private information related to a specific object resides in the instance.

Variables are the attributes of an object. There are two kinds of variables: instance variables and class variables; both are defined when a class (of objects) is defined. Instance variables represent the objects of a class in which the variables are

encapsulated. If a developer writes code outside of this class that references these variables the Smalltalk programming environment refuses to accept the code by responding with an error message which means instance variables are private and can only be accessed within an instance. Class variables are used to describe the characteristics of a class. They can be accessed by all objects in the same class directly or by the objects in other classes indirectly.

A method is a synonym for an operation which is invoked when a message is received by an object. Message protocol is a set of messages to which an object can respond. The essential activity in the Smalltalk is sending messages to objects. When an object receives a message which it recognizes, the object responds to the method represented by the message. The responses from the object is the solution this activity in intended.

Object-oriented problem-solving in the Smalltalk consists of identifying the objects, messages, and object-message sequences to effect a solution. In other words, the object-oriented problem-solving process includes the following steps [Pinson]:

(1). stating the problem to be solved;

(2). identifying the objects and classes in the problem;

(3). defining the messages to which those objects should respond; and

(4). establishing a sequence of messages to the objects that provide solution to the stated problem.

This sequence of the problem-solving process will be observed in the following description of the implementation of the FEE structure.

6.3. Object-Oriented Design Information Structure

In order to implement the Function-Environment-Embodiment design information structure in an object-oriented paradigm, the FEE structure is defined as three distinct classes according to its three separate domains, namely: Function Class, ENvironment Class, and Embodiment Class. It should be noted that the **EN**vironment Class is an application class which is different from the **En**vironment Class defined by Smalltalk-80 as a system class. Each of the three classes represents the domain knowledge as defined in the previous chapter. All three classes are defined as subclasses of the Object Class. As shown in Figure 6.1, the three classes are treated as separate classes under the Object Class without hierarchical relationships. The interactions between the three classes are achieved by sending messages to other classes or by responding to the messages received from other classes.

To further clarify the three classes, the subclasses within each of them need to be identified. The Function Class includes the information about the hierarchical relationships of the design functions. The information in this class is represented in different sub-classes. Figure 6.2 shows the class hierarchy under the Object Class. It also shows some hierarchies within each of the three classes. The Function Class is seen

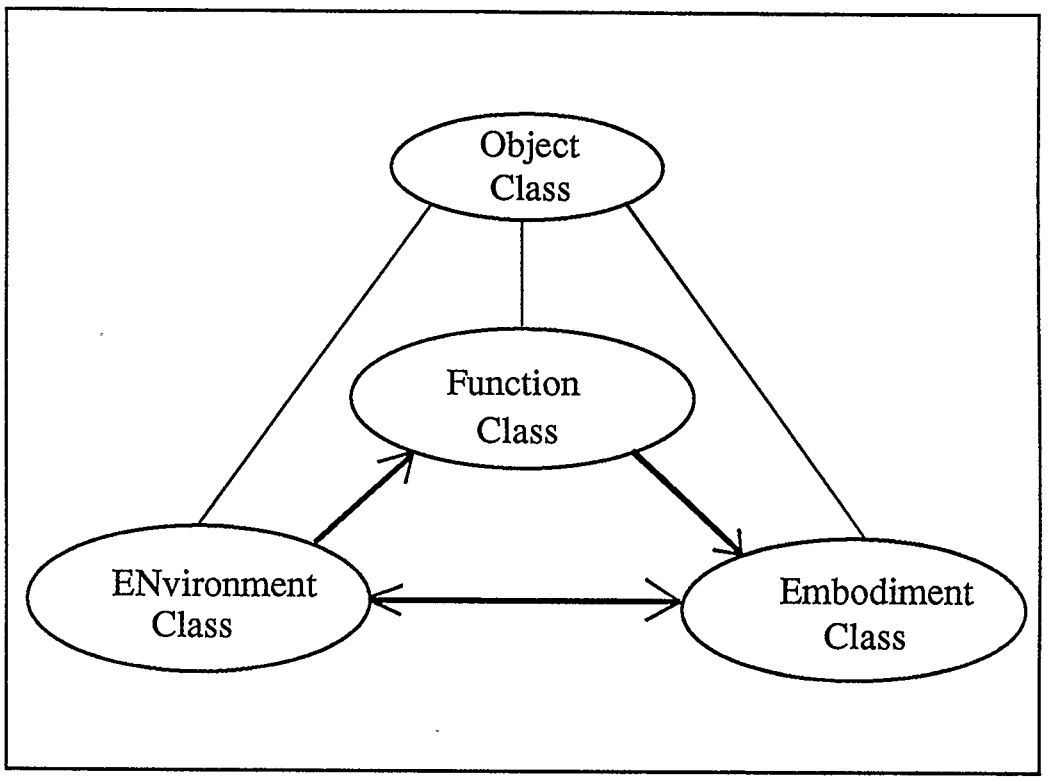

Figure 6.1. FEE Domains as Classes in OOP.

Three information domains are defined as subclasses of the Object Class. There are no hierarchies between the three domains. They interact with each other by sending messages.

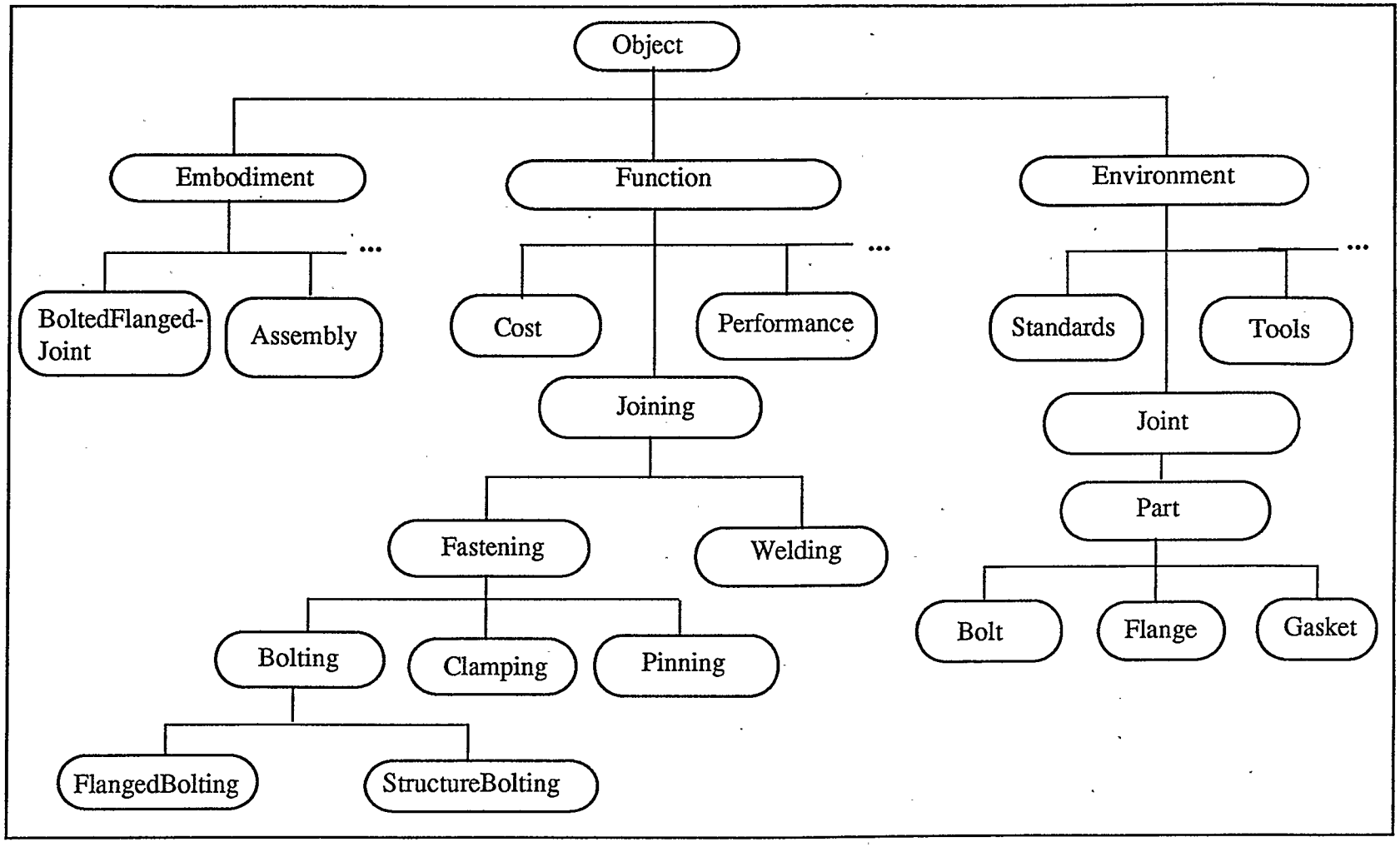

Figure 6.2. Joint Design Information Hierarchy

to include a variety of joining functions which formed a major subclass hierarchy in this class. Design functions in these subclasses can relate objects from outside of the Function Class such as from ENvironment class to form some function-form pairs. When some functions need to be satisfied by some physical forms, they are evaluated within this class hierarchy and the relations towards the forms within the ENvironment Class are set in order to find a certain form to satisfy the functional needs.

The ENvironment class has a relatively shallow class hierarchy compared with the functional hierarchies in the Function Class. The ENvironment as an abstract class holds many parallel sub-classes such as existing designs (in this case, the Joint Class), Standards Class, Tools Class, etc. Decision-making in the Function Class depend on the information available within this class. The Embodiment Class stores the information about the design results from the design stages. This information may be in a form of a hierarchical relationship of the physical configuration of the design corresponding to the hierarchical relationships of functions within the Function Class. These results can be accessed any time during or after the design process.

6.4. Joint Component Selection in the FEE Structure

The joint component selection process described in the previous chapter is applied as a design example for the implementation of the FEE structure in the OOP paradigm. On one hand this design problem provides the information on which the FEE is built, on

the other hand, the design problem is to deal with the specific design components which are naturally considered as object concepts in the OOP paradigm. Therefore, in the following context this design problem is treated as an object-oriented problem and then solved within the FEE framework.

6.4.1. Class and Object Identification

Figure 6.3 shows some pairs of class designation and their corresponding physical world entities. As described previously the three domains in the FEE structure are defined as three classes named as Function Class, ENvironment Class and Embodiment Class. Each of these classes represents its corresponding part of the world. Information in these three classes can further be classified into different subclasses and ordered into different hierarchies. Other design entities (which are actually defined as subclasses of the above three classes) involved include the joint to be designed, the existing components and the other design cost aspects such as cost, assemblability, etc. Each class represents a type of objects which have similar characteristics. For example, the Joint Class is an abstraction of all possible joints to be designed; for any given condition there is one instance of Joint to be defined. This instance will gradually become specified from its initial conception. Other classes such as the Flange Class, the Bolt Class, describe the already known knowledge about certain objects. Since Flange Class represents the set of existing flanges, one instance of this class represents a paticular flange with all its characteristics.

Design paradigm		*Physical world entity*
ENvironment	-->	*the real world environment in which design takes place and with which new design interacts*
Function	-->	*design principles*
Embodiment	-->	*design results*
Joint	-->	*joint to be designed*
Flange	-->	*flange existing in the environment*
Gasket	-->	*gasket existing in the environment*
Bolt	-->	*bolt existing in the environment*
Cost	-->	*cost of any entity in the problem domain*

Figure 6.3. Major Class Designation of the Joint Paradigm.

Each class represents a type of objects with similar characteristics.

6.4.2. Object Representation

A Class is an abstraction of a certain type of object. Figure 6.4 shows the Joint Class hierarchy in the FEE structure. From this class hierarchy it is noted that there are several instance variables defined within each class. It is those instance variables which describe the characteristics of the objects. For example, the Joint Class has four instance variables: 'bolt', 'flange', 'gasket', and 'flag'. These variables when filled with values will define an instance of the Joint Class, which means a joint is specified. Different values for these variables describe different joints.

The Joint Class has a sub-class: Part Class, which abstracts all the components of a joint. The instance variables in this class describe a part by its name, material, type and cost. Class hierarchy permits inheritance of all variables from superclass to sub-class. Therefore, the objects (which represent those three type of components) defined by the Flange Class, Gasket Class, and Bolt Class will inherit those instance variables defined in the Part Class. The instance variables inherited from the superclass together with those defined within subclasses can fully describe the characteristics of an object. For example, the Gasket Class has ten instance variables defined within the class itself. Only when these variables are used together with the inherited variables from the Part Class can a specified gasket be determined.

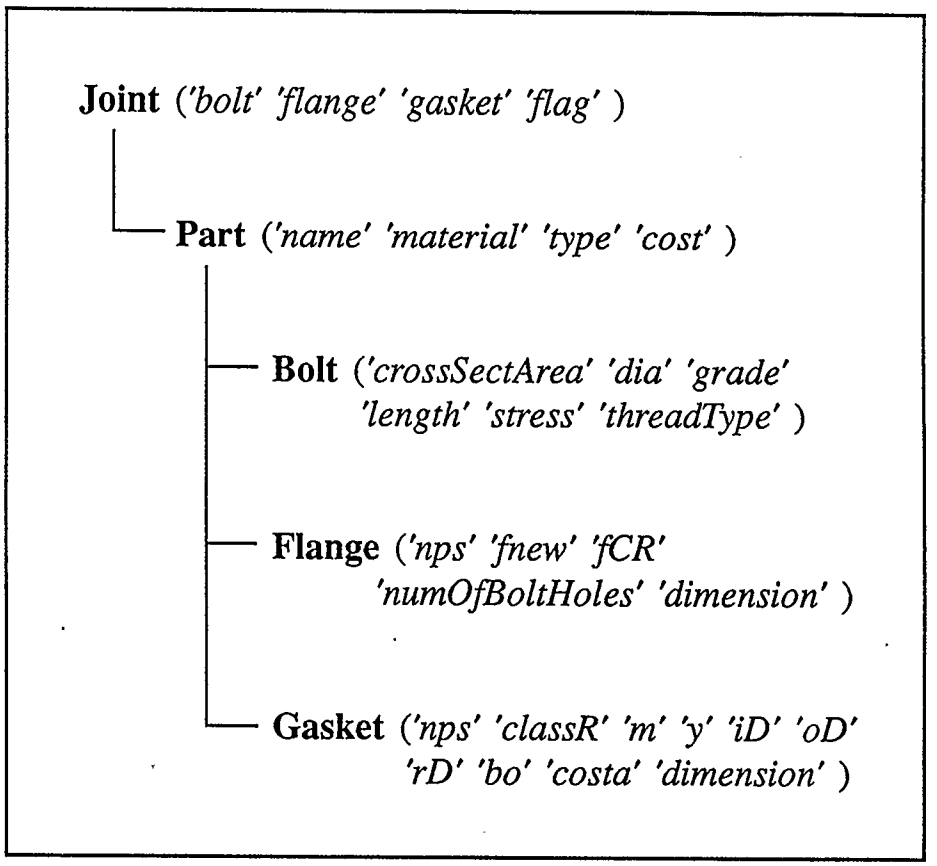

Figure 6.4. Major Classes in Joint Class Hierarchy.

Instance variables shown in the brackets are representing the characteristics of the objects defined by the class. Each component class inherites the superclass instance variables. For example, the 'name', 'material', 'type', and 'cost' are inherited by Flange, Bolt, and Gasket Class. The instance variables represent different objects when they have different values.

6.4.3. Message Protocol in Major Classes

Since the primary purpose of the process is to design a joint, the messages which are supporting the design process are then located in the Joint Class as seen in Figure 6.5. When the design process begins these messages are sent to an instance of the Joint Class. The response from the instance of the Joint Class will calculate the general behaviours of a joint as an instance of the Joint Class. In the similar situation, Figure 6.6 shows some messages designed in each of the Flange, Gasket, and Bolt Class. These messages represent the methods implemented within each individual class. The methods in these component classes are usually used for processing the data of instances in order to represent their characteristics in a functional perspective. Therefore, the message protocol should be named in some functional terms so that the design process is easy to match the functional requirements with these physical components to complete the design purpose.

Messages are the interfaces defined for the objects. Through the interface the information encapsulated within the objects can be accessed. For the joint design problem, the information about each component such as bolt, flange and gasket is hidden within itself. The characteristics of the information is in the form of codes and data. The design process needs the functional representation of the codes and data in order to easily connect the functions and forms. Therefore, the functional interface needs to be built to access the characteristics of the design objects. From Figure 6.6 the messages within

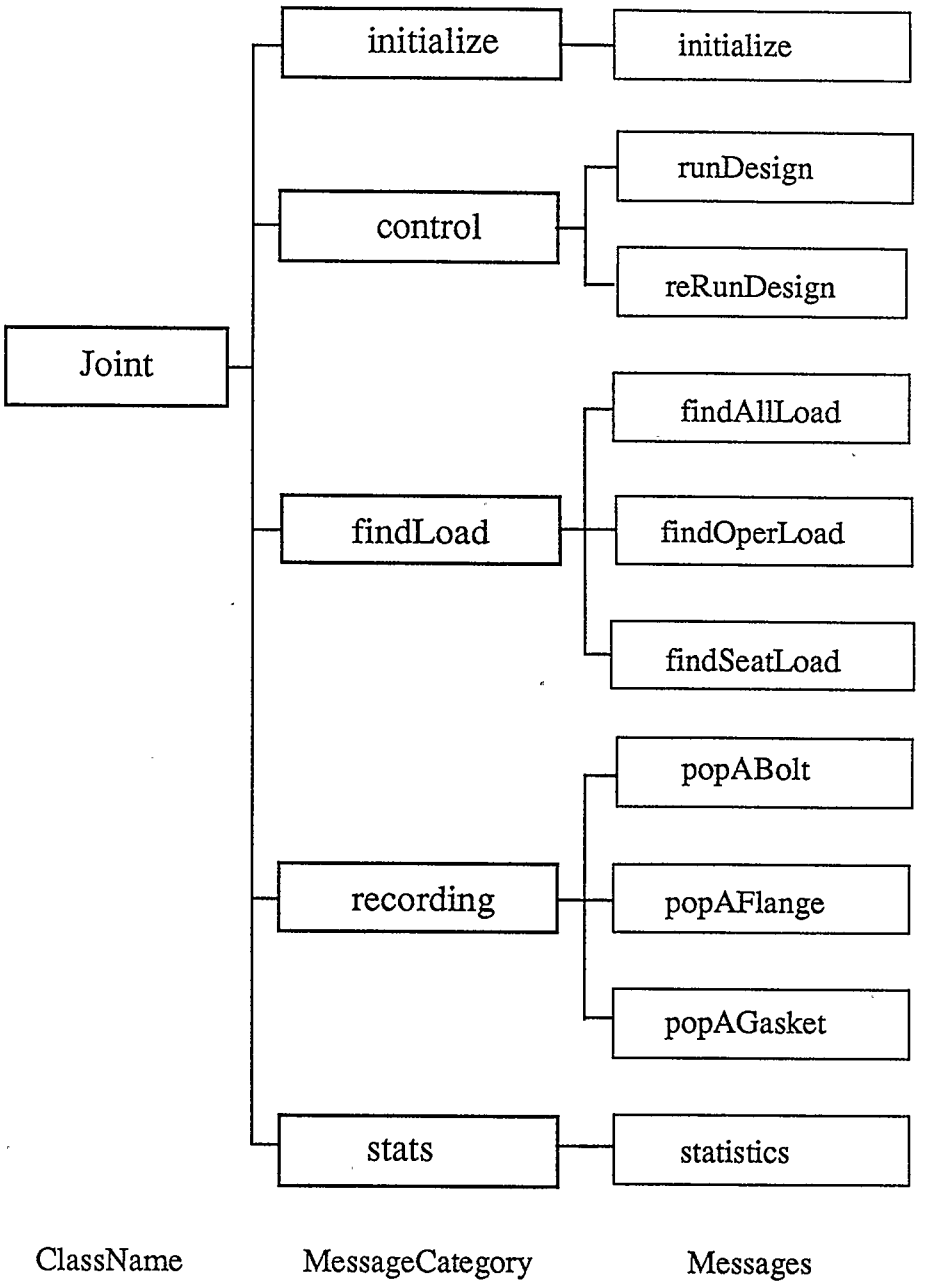

Figure 6.5. Joint Class Message Protocol.

Messages defined in Joint Class control the design process.

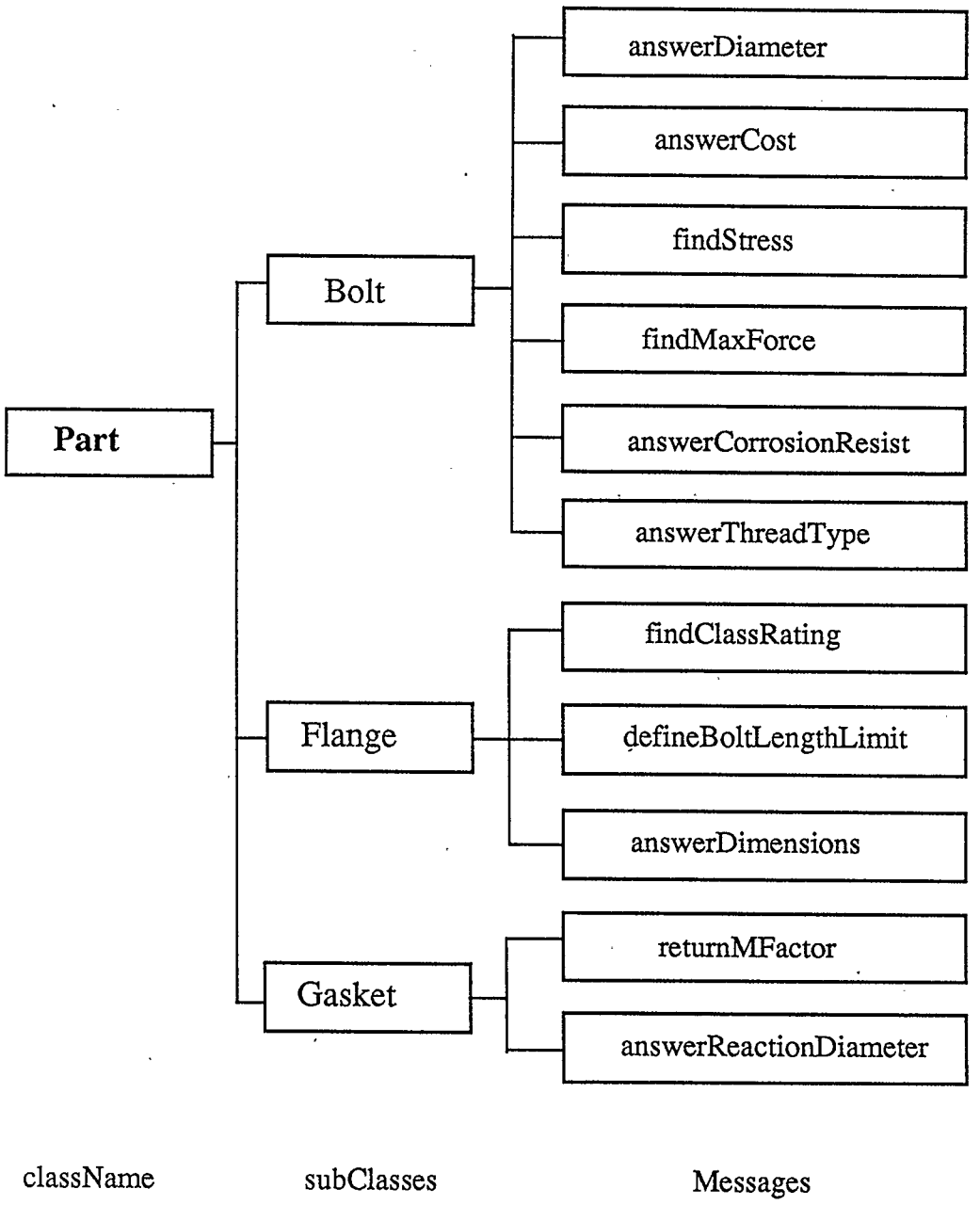

Figure 6.6. Message Protocol from different classes.

Messages defined in the subClasses of the Part Class provide operations applicable to the different subClasses.

each class can be thought of as functional interfaces because they do convey certain functional meanings for the design process. Through this functional interface the different kinds of data carried within the components which will be described in Figure 6.7 can be connected with higher level design functions.

6.4.4. Selective Data Initialization

There is much design data related to some types of design problem. It is preferable to initialize only the useful range of data for the specific design problem so that the programming efficiency can be maintained and the memory space can be saved. For the Flange, Gasket, and Bolt Class concerned, the objects in each class have a very wide range of behaviours represented by the existing design data. It is useful to choose only the related range of data according to the design conditions defined by the application.

For the three classes mentioned above, the data initialization is mostly to build up data objects. Data-enriched objects have useful knowledge to readily respond to any messages. For example, if a certain strength is expected from a Bolt Object, a message is sent to the Bolt to request a value of the strength from the Bolt Object. The Bolt Object can provide an answer if it has the necessary information to calculate the strength. Likewise, if all the objects in the design process have the information about themselves and the associated information processing tools, the object-oriented paradigm will be more

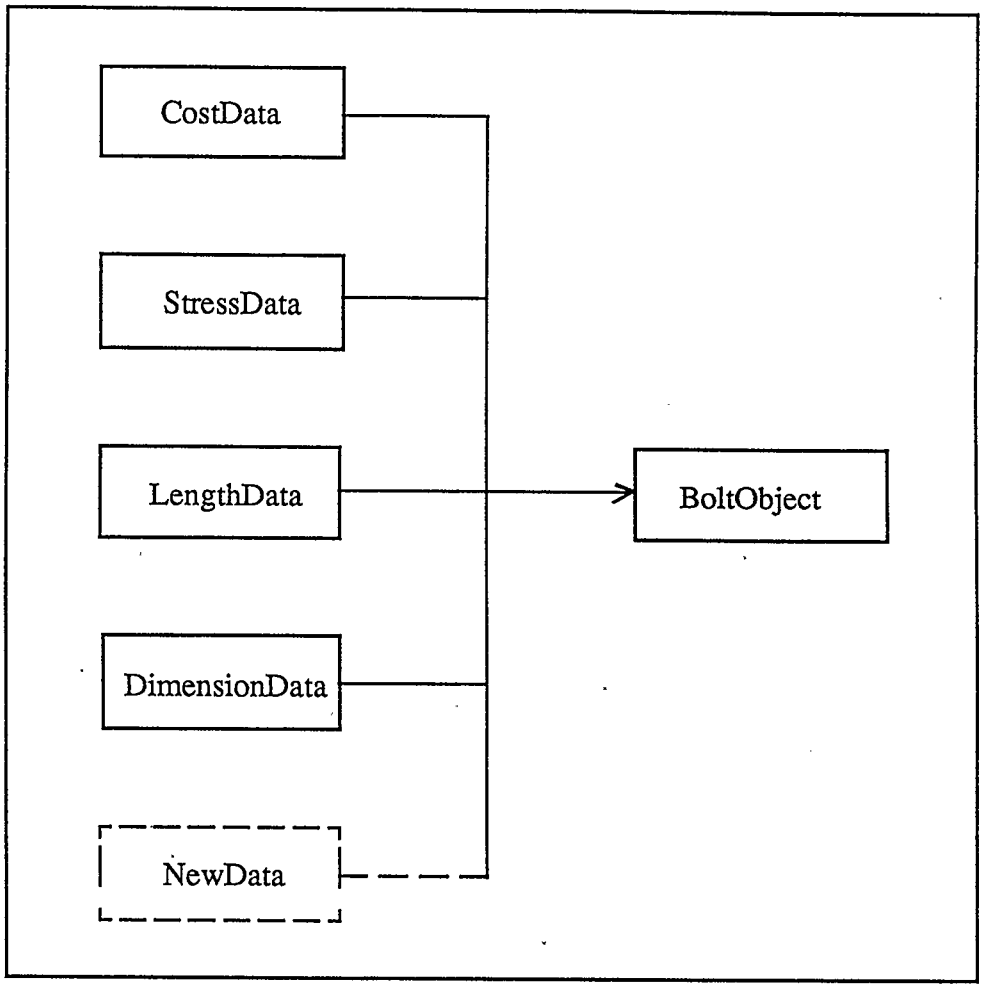

Figure 6.7. Bolt Data Construction.

Data initialization makes the different data from different sources available to construct the data object. The Object in OOP is considered carrying data with it. These data can only be accessed by sending messages to it. Stress-Data is from Table5.11, DimensionData is from Table 5.12. Different data comes from different sources. NewData can also be added into the object at any time.

effective in solving the design problem. Figure 6.7 shows the data object construction from the selected data sources. The process of building up a data object is to fill the instance variables of the object with the corresponding values from different databases such as standards and design codes. For example, the StressData is from Table 5.11, which fills the instance variable 'stress' of the Bolt; the DimensionData is from Table 5.12, which fills the instance variables about the dimensions; the LengthData and the CostData in the similar way provide values for their corresponding variables of the object. This indicates that the existing design data in various forms or standards are being connected within the design objects and consequently with the design process.

Once the data are put into the variables of the design objects no matter what form they may take, the objects are able to respond to the messages by processing these data to provide answers for the messages. Because the messages are represented as functional interfaces these data within the objects are connected with functional meanings.

6.4.5. Design Process Main Driver Program

As shown in Figure 6.8, the joint design process begins with the input of design conditions. Through the interface interaction, the designer inputs the design requirements such as design pressure, design temperature, and the nominal pipe size to the design process. These parameters are the primary considerations for the design purpose. There may be other design constraints such as cost limits, tool availability imposed on the

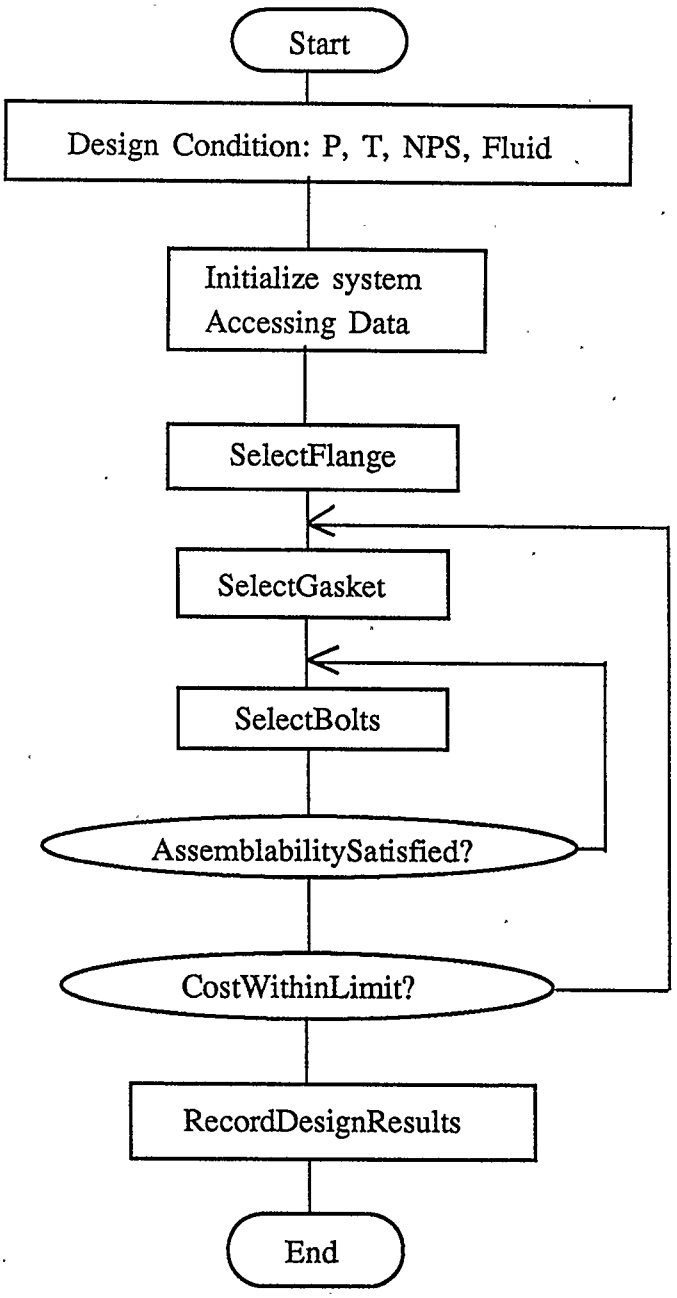

Figure 6.8. Joint Design Process Main Driver Flow Chart

design process, which can also be input to the system at the beginning of the design process. According to the design requirements, only the useful data is initialized. Then it comes to the stage that the selection of the components dominates the whole process. Three major components are decided by various decision-making operations through the process of satisfying the design requirements. Once the components are selected, they must be checked against each other to ensure the assemblability and the cost requirements. If the requirements are not met, some iterations are made to search for a more appropriate design solution. When a satisfying solution is found the design process records the results with related design information.

To implement the design process described above, a main driver program is created in Smalltalk-80 as shown in Figure 6.9. Several messages are sent to an instance of the Joint to initiate the design actions. In this case, there are four messages sent to the 'new self' object which represents the 'joint' to be designed. These messages are recognized by any Joint to be designed and therefore incur a series of actions to solve the design problem. The code in this figure shows three important points: the 'jointDesign' is defined within the Joint Class; the method or message name is the 'jointDesign'; and the sequence of actions is defined in the last line. The responses from the 'joint' to the messages 'title' and 'init' will create a title in the record of this design session and initialize all the design data objects. The response to the 'run' message will activate the actual selection process of the components. Other aspects such as assemblability and cost evaluation are also performed during the design process. Responses to the 'stats' will

record the design results and put them into the Embodiment Class.

```
'From Objectworks for Smalltalk-80(tm), Version 2.5 of 29 July 1989 on 25
October 1992 at 11:06:04 pm'!
!Joint class methodsFor: 'jointDesign'!

jointDesign
"This is the main driver method to the bolted joint design process.
The start-up of the process is made in this method by initializing the
Class itself. Other processes are activated following the initialization."

(self new) title; init; run; stats.
```

Figure 6.9. Smalltalk-80 Code for the Design Main Driver Program

6.4.6. Open-Ended Interface Structure

Object-oriented design involves two views of objects. One is the internal view which only applies to the programmer, and the other is the external view which is designed for the user to manipulate objects. The interface of a class in Smalltalk is this kind of external view. The class interface encompasses the abstraction of the behaviour common to all instances of the class. The internal view is the implementation of a class

which comprises the representation of the abstraction as well as the mechanisms that achieve the desired behaviour. In Smalltalk, class interface is a list of messages representing the methods defined within the class. The implementation of methods and instance variables are encapsulated. It is to the programmer's advantage to change the implementation while a user can only access the object via class interface. Therefore, the object-oriented programming has a stable structure because the implementation can be changed without changing the user interface.

Figure 6.10 shows a partial interface of the Bolt Class. In the design process, it often happens that the designer lacks access to some kind of knowledge or information. This information may be available through the research accomplishment in other fields or implementations of certain knowledge base or expert systems. Therefore the design process can establish some open slots to adapt the information needed whenever it is available. For example, the messages printed in italics in the Figure indicate that those methods they are representing are still not available. This kind of open structure encourages the continual improvement of the design process because the better methods can always be added into the design process whenever they are available. Therefore the design flexibility in adapting further information is achieved.

In addition to the above, new messages and methods can always be readily added into the design process through the information structure. This indicates that the design information structure within the object-oriented paradigm has much potential to facilitate

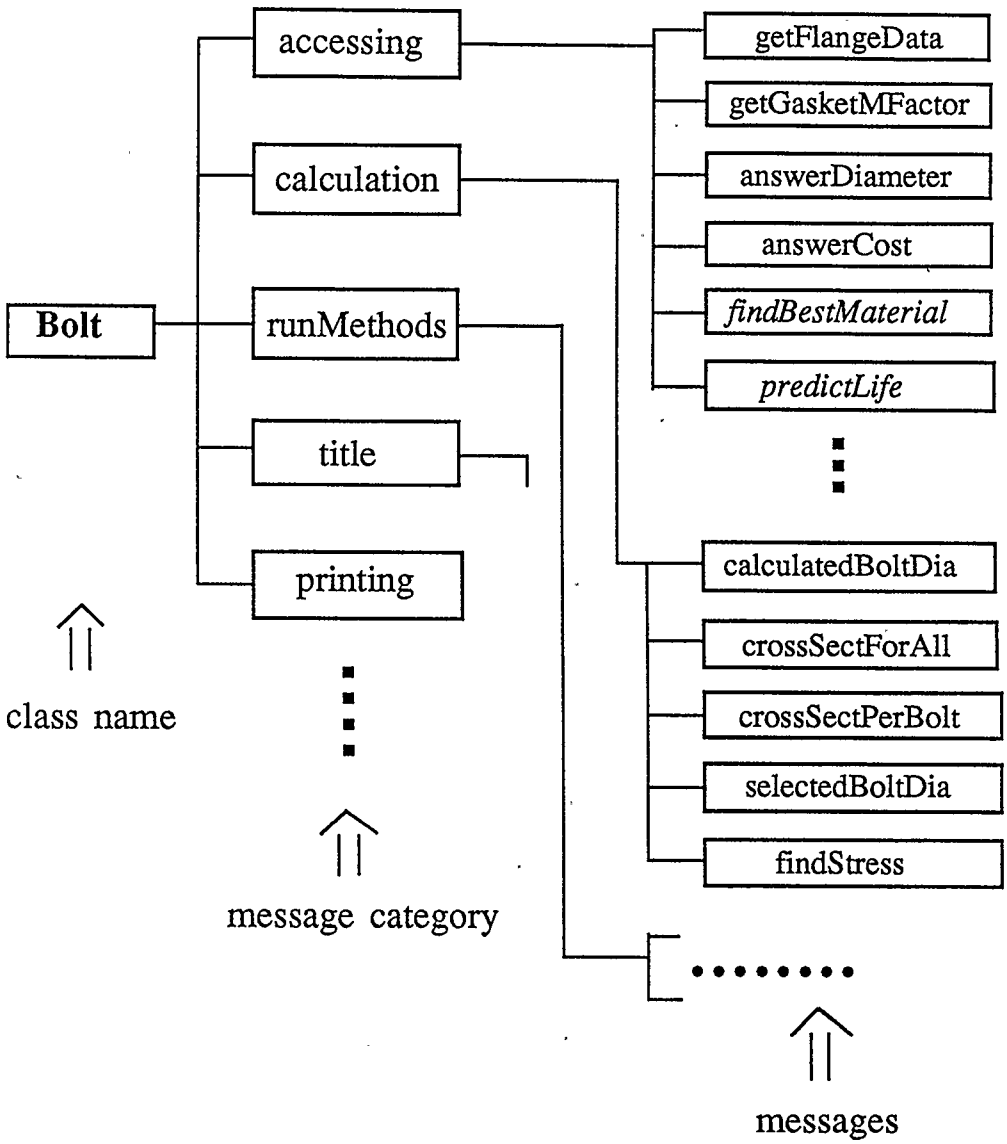

Figure 6.10. Open-ended Interface Structure
Methods in Bolt Class are used for accessing the Bolt information. Further messages can be easily added into the system. Open slots can be set to access information from other sources.

the design problem-solving process.

6.4.7. Design Process within the Object-Oriented FEE Structure

As a conclusion of the discussion made previously on the object-oriented joint component selection process within the FEE structure, a simplified overall FEE design information structure is shown in Figure 6.11 to demonstrate the information interactions during the design process between the objects defined within the different classes. The essential part of the design process is through the interactions between the Joint Class in the ENvironment Class and the Joining Class in the Function class. The 'design process' within the Joint Class serves as a process manager to pass the User input design requirements to the Joining Class. Within this class functional analysis is made to choose the FlangeJoining function to satisfy the design requirements. In order to achieve this function the subfunctions must be satisfied through the ENvironment Class. For example, the 'facing' function needs to be satisfied by a pair of flanges and therefore a message is sent to selection the certain flanges among many flanges in the Flange Class. When a selection is made the result is saved into the Embodiment Class. In a similar way, other functions in the Function Class can be satisfied by certain parts in the ENvironment Class and the results can also be saved into the third class.

Other subclasses may also be involved in the design process in a similar way as described above. The results of the joint component design process implemented within

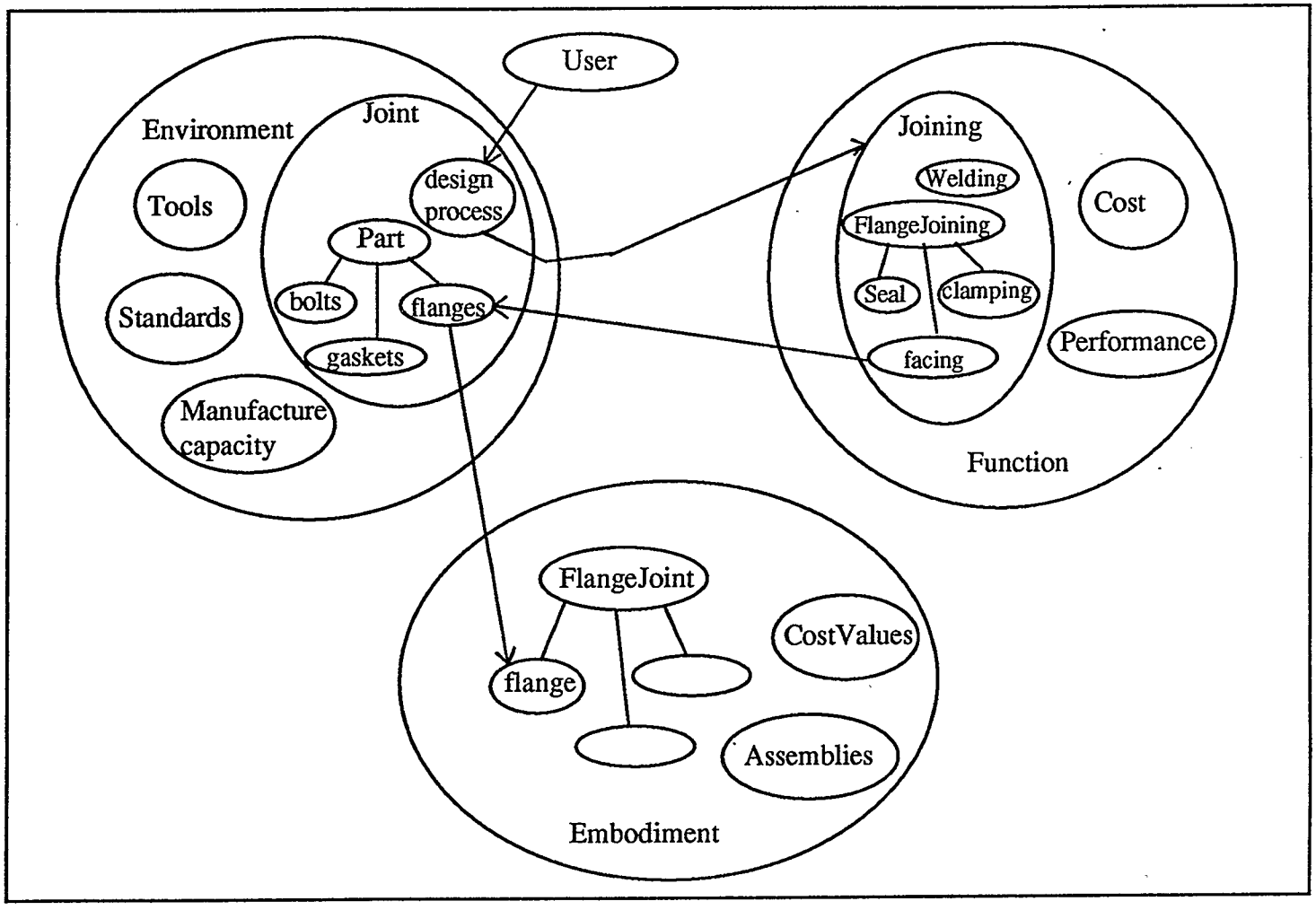

Figure 6.11. Object-oriented interactions between different classes in the FEE structure.

this paradigm will be explained in the following context.

6.4.8. Joint Design Results

The Joint design process implemented in the FEE structure is used to solve a component selection problem of a joint design. Using the Smalltalk programming environment the joint design conditions are input into the Joint Class as Class variables which are recognized by all objects within this class. The design process defined within the ENvironment Class compares these inputs such as design temperature, design pressure, and nominal pipe size with the knowledge in Function Class, and offers a joining method for the 'Pipe Joining' in the form of a Bolted Flange Joint. Upon decision of this type of joint, the design process again interacts with the Function Class for further direction. Then the selection of the three components is advised. The design process then goes to the Part Class to find parts. Three classes of parts are defined within Bolt, Flange, and Gasket Class. Messages sent to each instance of these classes are understood by the interfaces defined with the same names as the messages. By responding these messages such as 'findForce', 'findDiameter', 'findMaterial', etc. the corresponding object will answer appropriate values through processing its own data or accessing related information. With the progress of the design process, the results through each of the steps are recorded. When the components are selected they are checked against each other for assemblability. If there is any problem the decision is made to choose an alternative bolt material and then re-select the bolt. Finally the selected components are

listed. The cost related to the design is evaluated at the end of the design to provide cost information. Appendix Two lists some design process recordings from three design executions. These results demonstrate that the design process modelled within the object-oriented FEE structure is capable of connecting the design information from different aspects for the specific design purpose such as selecting the appropriate components for the proposed joint design.

Results No.1 shows that the design process has found appropriate components in the first iteration at given conditions. The major control of the design process is by checking the assemblability of the three components. For example, the bolt diameter is checked against the flange hole. When they match with each other, the design process can proceed. The result of the design process is a list of components and a cost evaluation based on the components selected. In an actual design situation meaningful values for different costs may be obtained from different sources. Here as a demonstration of the principles of cost evaluation most of the values are arbitrarily set except for the bolt cost values which are from a sales department.

Results No.2 shows that an iteration is needed to find a set of appropriate components. The re-run process uses another bolt material which results in a different bolt diameter selection.

The Results No.3 shows that the design process has not determined an appropriate

bolt because the design database is not big enough. In this case, the database for bolt materials has only six different materials. If the design data is sufficient, good results may be achieved.

6.5. Summary

Object-Oriented Programming language has been used for the implementation of the FEE design information framework in this chapter. The results from the design example show that the FEE design framework has the capability of integrating the different design information from variety of data sources. On the other hand, the design information structure implemented in the Smalltalk-80 encourages the incremental programming to problem-solving. This feature will facilitate the design process in making design changes, improving system performance and expanding the system capacity. The design problem, for example, the joint design problem, can be an open-ended problem which can be improved with new components and materials. The design process should always be improved to achieve a better efficiency.

CHAPTER 7 CONCLUSION AND FUTURE DIRECTION

7.1. Introduction

It has been recognized that one important problem in design process is the lack of design information/knowledge integration. The solutions to this problem are being pursued by a variety of design research activities. Although many approaches have been pursued, there are few effective methods which have as yet been proven to achieve this goal. Methods for concurrent design have been aimed at integrating the design information from a variety of different sources but the generalized methods of integration remain to be researched. As a supportive tool for concurrent design, this thesis presents a design information structure which classifies the design information/knowledge into three distinct domains in which the design functions and forms are logically connected. Although the information and knowledge content provided by a routine design problem within the proposed structure has not yet addressed most aspects of the design process, the information structure can still be seen as a useful platform to facilitate the representation and connection of the different types of design knowledge during the design development process.

7.2. Conclusions

7.2.1. A Useful Design Information Framework

This thesis has presented a Function-Environment-Embodiment (FEE) design information structure for design information/knowledge integration. The FEE information structure classifies the information into three distinct domains according to the nature of different types of the information involved in the design process. The knowledge in the Environment domain represents the physical design environment which includes different knowledge aspects such as existing designs, components, tools, methods, and working conditions, etc. The knowledge in this domain forms the material basis for any new design to be developed. The function domain holds the information about the functional principles behind the explicit physical embodiment of the design. This information includes the form-function relationships which guide the directions of the design development process. For given design requirements the knowledge in this domain is consulted so that the decisions to achieve the requirements can be made. The design information in the Embodiment domain is in a parallel structure with that of the Function domain. It represents the results of the design process in a progressive hierarchical order from abstract to detail. These results are recorded with the progress of the function-to-form transformation. When the design is completed the Embodiment domain becomes a record of an existing design, which logically becomes a new portion of the knowledge in the Environment domain.

The advantage of representing existing design knowledge in this structure is to vprovide a mechanism to organize the large amount of knowledge involved in the design process in a better formulated and easily accessible way so that the design process can be expedited by using the existing knowledge instead of creating it from scratch.

7.2.2. Object-Oriented Knowledge Representation

Since the FEE design information structure has organized the design knowledge into three domains and each domain has many sub-domains to represent certain kinds of knowledge, the object-oriented concept is used to represent the different domains into different knowledge classes. At the same time design entities at a detailed level of the design process are represented as objects within certain classes. Since the objects are made to carry their physical characteristics extracted from the existing knowledge bases and have the ability to respond to functional requests, the interactions between different objects during the design process can then reflect functional attributes. Therefore, in a similar way the interactions between the design stage and other manufacturing stages can be facilitated by using function-oriented communications.

7.2.3. Design Knowledge Recording

Differing from the traditional design process within which the design results at different levels are separated and not closely threaded together, the design process

implemented in the FEE design framework has an automatic design recording capability to record the progress of the design development. By recording the design information about every design stage, a full description of the function-to-form transformation process can be obtained. Therefore, it is possible to review the design process at a later stage to evaluate the design decisions. Furthermore, the design recordings can be used as valuable examples in the process of design method teaching. When the results are proved to be useful they can be saved to enrich the existing design knowledge.

7.3. Discussion and Future Directions

The design information structure shown in this thesis has demonstrated usefulness in facilitating information interactions in the product design process. Although some of the function-to-form transformation process at a detail level has been represented, the knowledge related to the design process within the different domains of the structure still needs to be clarified and detailed. Difficulties are encountered during this process because the existing knowledge is often hidden or is in the form of obscure data and codes instead of existing as a functional representation.

Further development of the FEE information structure can be pursued by organizing and representing the existing knowledge into appropriate categories to increase the knowledge content of the structure. As pointed out above the functional representation of the existing knowledge remains to be the key problem in achieving that

purpose. Since the logical connections between the design function and the existing form are primary to the knowledge integration from different domains, they should be also established during the process of the knowledge organization.

In order to establish the functional connections between the design function and the various data of the existing design knowledge, it is necessary to build up functional interfaces for the present existing forms of design. Through these interfaces the functional implications of the existing design data can be represented by some pre-defined methods.

REFERENCES

Akagi, S. and K. Fujita, 1988: Building an expert system for engineering design based on the object oriented knowledge representation concept. ASME Advanced Design Automation, Vol. 1

Akman, V., P.J.W. ten Hagen and P.J. Veerkamp, 1989: Intelligent CAD System II. Spring-Verlag

API Standard 601, 1985: Metallic Gaskets for Raised-Face Pipe Flanges and Flanged Connections (Double-Jacketed Corrugated and Spiral-Wound). American Petroleum Institute, Sixth Edition, September

API Standard 605, 1981: Large-Diameter Carbon Steel Flanges. American Petroleum Institute, ANSI Third Edition Approval, July 8

ASME Code (a) 1989: TABLE 3-320.2, Effective gasket width, Section VIII -- DIVISION 2

ASME Code (b) 1989: TABLE ABM-1, Allowable bolt stress values in tension for ferrous bolting materials for use with flanges designed in accordance with appendix 3, SECTION VIII -- DIVISION 2

Bedworth, D.D., M.R. Henderson, and P.M. Wolfe, 1991: Computer-Integrated Design and Manufacturing. McGraw-Hill, Inc.

Booch, G., 1991: Object-Oriented Design with Applications. The Benjamin/Cummings publishing Company, Inc.

Chao, N.H., S.C.Y. Lu, edited, 1989: Concurrent Product and Process Design. ASME DE-Vol. 21, PED-Vol. 36

Coad, P. and Edward YourDon, 1991 b: Object-Oriented Design. Yourdon Press Computing Series

Cohen, P.H., S.B. Joshi, 1990: Advances in Integrated Product Design and Manufacturing. ASME PED-Vol. 47

Cross, N., 1989: Engineering Design Methods. John Wiley & Sons Inc.

Cullum, R.D. 1987: Handbook of Engineering Design. Butterworths

DeLorge, D., 1992: Product Design and Concurrent Engineering. SME Blue Book Series, Published by CASA, SME.

Dieter, 1983: Engineering Design, A materials and Processing Approach. McGraw Hill

Fauvel, O.R. 1991: Expanded use of function language in mechanical design. Paper prepared for Canadian Congress on Applied Mechanics, June

Fauvel, O.R. 1992: A Design Information System. IEEE 1992 Frontiers in Education Conference

Finger, S. and J.R. Rinderle, 1989: A Transformational approach to mechanical design using a Bond Graph Grammar. Design Theory and Methodology, ASME conference proceedings, New York

Hubka, V. 1982: Principles of Engineering Design. Butterworths

IMechE (a) 1986: Pipe Joint, Part 1: Gasket: a state of the art review. The Institution of Mechanical Engineers London

IMechE (b) 1986: Pipe Joint, Part 3: Metallic pipe joints. The Institution of Mechanical Engineers London

Ishii, K., 1990: The role of computers in simultaneous engineering. ASME Computers in Engineering, Vol. 1

Jones, S.W. 1973: Product Design and Process Selection. Butterworths, London

LaLonde, W.R. and J.R. Pugh, 1990: Inside Smalltalk, Volume 1. Prentice-Hall Inc.

Long, M.W., S.G. Bailey, and W.R. Shawver, 1988: An expert system for mechanically fastened joint design. ASME computers in engineering, Vol 1

Michaels, J.V. and W.P. Wood, 1989: Design to Cost, New Dimensions in Engineering. John Wiley & Sons, Inc.

Nevins, J.L. and D.E. Whitney, 1989: Concurrent Design of Products & Processes. McGrow Hill

N.R.C (National Research Council), 1991: Improving Engineering Design, designing for competitive advantage. National Academy Press

Pahl, P. and W. Beitz, 1984: Engineering Design. The Design Council, Springer-Verlag

Pinson, L.J. and R.S. Wiener, 1988: An Introduction to Object-Oriented Programming: SMALLTALK. Addison-Wesley Publishing Company

Ramchandran, N., A. Shah, and N.A. Langrana, 1988: Expert system approach in design of mechanical components. ASME computers in engineering, Vol 1

Rasmussen, J., 1986: Information Processing and Human-Machine Interaction. North-Holland

Rawling, J.W., 1991: Simultaneous Engineering - the Competitive Edge. Proceedings of ICCIM 1991, Organized by Nanyang Technological University, Singapore

Robinson, P. (Edited), 1992: Object-Oriented Design. Chapman & Hall

Rynchener, M.D., 1988: Expert System For Engineering Design. Academic Press Inc.

Srinivasan, S. and R.H. Allen, 1989: Partitioning and guided search: A generalized approach to problem solving in preliminary design. Design Theory and Methodology, ASME conference proceedings, New York

Suh, N.P., 1990: The Principles of Design, Oxford series on Advanced Manufacturing. Oxford University Press, New York

Ulrich, K. and W. Seering, 1987: A computational approach to conceptual design. International Conference on Engineering Design, ICED 1987, Boston, MA

VDI 2221 Guidelines, 1986: Systematic Approach to the Design of Technical Systems and Products. Translation of the German edition 11/1986

Waldron, M.B. and C.W. Chan, 1988: Object-oriented system for component selection. ASME Computers in engineering, Vol 1

Wright, M.A., T. Wright, And D.G. Jacson, Jr. 1988: An expert system for selection of axial fans. ASME computers in engineering, Vol 1

Yokoyama, T. 1990: An object-oriented and constraint-based knowledge representation system for design object modelling. Sixth IEEE Conference on Artificial Intelligence Applications, March

Zeidner, L. and Y. Hazony, 1992: Seamless Design-to-Manufacture (SDTM). Journal of Manufacturing Systems, Vol 11 /No.4

APPENDIX ONE

Metal Temperature (degrees Fahrenheit)	Pressure (pound per square inch gage) Class					
	75	150	300	400	600	900
100	140	285	740	990	1480	2220
200	130	260	675	900	1350	2025
300	115	230	655	875	1315	1970
400	100	200	635	845	1270	1900
500	85	170	600	800	1200	1795
600	70	140	550	730	1095	1640

Table 5.1. Class Ratings for Plant Piping and Pressure Vessel Flanges (Extracted from Table 1-A in [API 605]).

NPS	Flange OD	Flange Thickness	Raised-Face Diameter	Bolt-Circle Diameter	Number Of Bolts	Bolt Diameter	Bolt-Hole Diameter	Bolt Length
26	30.00	1.31	27.75	28.50	36	5/8	0.75	4.00
28	32.00	1.31	29.75	30.50	40	5/8	0.75	4.00
30	34.00	1.31	31.75	32.50	44	5/8	0.75	4.00
32	36.00	1.38	33.75	34.50	48	5/8	0.75	4.25
34	38.00	1.38	35.75	36.50	52	5/8	0.75	4.25
36	40.69	1.44	38.00	39.06	40	3/4	0.88	4.50
38	42.69	1.50	40.00	41.06	40	3/4	0.88	4.50
40	44.69	1.50	42.00	43.06	44	3/4	0.88	4.50

Table 5.2. Dimensions for Class 75 Welding Neck Flanges
(Extracted from Table 2 in [API 605]).

NPS	Flange OD	Flange Thickness	Raised-Face Diameter	Bolt-Circle Diameter	Number Of Bolts	Bolt Diameter	Bolt-Hole Diameter	Bolt Length
26	30.94	1.62	28.00	29.31	36	3/4	0.88	4.75
28	32.94	1.75	30.00	31.31	40	3/4	0.88	5.00
30	34.94	1.75	32.00	33.31	44	3/4	0.88	5.00
32	37.06	1.81	34.00	35.44	48	3/4	0.88	5.25
34	39.56	1.94	36.25	37.69	40	7/8	1.00	5.75
36	41.62	2.06	38.25	39.75	44	7/8	1.00	6.00
38	44.25	2.12	40.25	42.12	40	1	1.12	6.25
40	46.25	2.19	42.50	44.12	44	1	1.12	6.50

Table 5.3. Dimensions for Class150 Welding Neck Flanges
(Extracted from Table 3 in [API 605]).

NPS	Flange OD	Flange Thickness	Raised-Face Diameter	Bolt-Circle Diameter	Number Of Bolts	Bolt Diameter	Bolt-Hole Diameter	Bolt Length
26	34.12	3.50	29.00	31.62	32	1 1/4	1.38	9.25
28	36.25	3.50	31.00	33.75	36	1 1/4	1.38	9.25
30	39.00	3.69	33.25	36.25	36	1 3/8	1.50	9.25
32	41.50	4.06	35.50	38.50	32	1 1/2	1.62	10.75
34	43.62	4.06	37.50	40.62	36	1 1/2	1.62	10.75
36	46.12	4.06	39.75	42.88	32	1 5/8	1.75	10.75
38	48.12	4.38	41.75	44.88	36	1 5/8	1.75	11.50
40	50.12	4.56	43.88	46.88	40	1 5/8	1.75	11.75

Table 5.4. Dimensions for Class 300 Welding Neck Flanges
(Extracted from Table 4 in [API 605]).

NPS	Flange OD	Flange Thickness	Raised-Face Diameter	Bolt-Circle Diameter	Number Of Bolts	Bolt Diameter	Bolt-Hole Diameter	Bolt Length
26	33.50	3.50	28.00	30.75	28	1 3/8	1.50	10.00
28	36.00	3.75	30.00	33.00	24	1 1/2	1.62	10.50
30	38.25	4.00	32.25	35.25	28	1 1/2	1.62	11.00
32	40.75	4.25	34.38	37.50	28	1 5/8	1.75	11.75
34	42.75	4.38	36.50	39.50	32	1 5/8	1.75	12.00
36	45.50	4.69	38.62	42.00	28	1 3/4	1.88	12.75
38	47.50	4.88	40.75	44.00	32	1 3/4	1.88	13.00
40	50.00	5.12	43.00	46.25	32	1 7/8	2.00	13.75

Table 5.5. Dimensions for Class400 Welding Neck Flanges
(Extracted from Table 5 in [API 605]).

NPS	Flange OD	Flange Thickness	Raised-Face Diameter	Bolt-Circle Diameter	Number Of Bolts	Bolt Diameter	Bolt-Hole Diameter	Bolt Length
26	35.00	4.38	28.62	31.75	28	1 5/8	1.75	12.00
28	37.50	4.56	30.88	34.00	28	1 3/4	1.88	12.50
30	40.25	4.94	33.12	36.50	28	1 7/8	2.00	13.25
32	42.75	5.12	35.25	38.75	28	2	2.12	13.75
34	45.75	5.56	37.50	41.50	24	2 1/4	2.38	15.00
36	47.75	5.75	39.75	43.50	28	2 1/4	2.38	15.25
38	50.00	6.00	41.50	45.75	28	2 1/4	2.38	15.75
40	52.00	6.25	43.75	47.75	32	2 1/4	2.38	16.25

Table 5.6. Dimensions for Class 600 Welding Neck Flanges
(Extracted from Table 6 in [API 605]).

NPS	Flange OD	Flange Thickness	Raised-Face Diameter	Bolt-Circle Diameter	Number Of Bolts	Bolt Diameter	Bolt-Hole Diameter	Bolt Length
26	40.25	5.31	30.00	35.50	20	2 1/2	2.62	14.75
28	43.50	5.81	32.25	38.25	20	2 3/4	2.88	16.00
30	46.50	6.12	34.50	40.75	20	3	3.12	16.75
32	48.75	6.31	36.50	43.00	20	3	3.12	17.25
34	51.75	6.75	39.00	45.50	20	3 1/4	3.38	18.25
36	53.00	6.81	40.50	47.25	24	3	3.12	18.25
38	57.50	7.50	43.25	50.75	20	3 1/2	3.62	20.00
40	59.50	7.75	45.75	52.75	24	3 1/2	3.62	20.50

Table 5.7. Dimensions for Class 900 Welding Neck Flanges
(Extracted from Table 7 in [API 605]).

Flange Size (NPS)	Class 150			Class 300			Class400		
	ID	OD	RD	ID	OD	RD	ID	OD	RD
26	26.50	27.50	28.56	26.50	28.00	30.38	26.25	27.50	29.38
28	28.50	29.50	30.56	28.50	30.00	32.50	28.13	29.50	31.50
30	30.50	31.50	32.56	30.50	32.00	34.88	30.13	31.75	33.75
32	32.50	33.50	34.69	32.50	34.00	37.00	32.00	33.88	35.88
34	34.50	35.75	36.81	36.50	36.00	39.13	34.13	35.88	37.88
36	36.50	37.75	38.88	39.75	38.00	41.25	36.13	38.00	40.25
38	38.37	39.75	41.13	41.75	41.25	43.25	38.25	40.25	42.25
40	40.25	41.88	43.13	42.50	43.25	45.25	40.38	42.38	44.38

Table 5.8. Spiral-Wound Gasket Dimensions for API Standard 605 Flanges (inches), ID--Gasket Inside Diameter; OD--Gasket Outside Diameter; RD--Center Ring Diameter. (Extracted from Table A-5 in [API 601])

Flange Size (NPS)	Class 600			Class 900		
	ID	OD	RD	ID	OD	RD
26	26.13	28.13	30.13	27.25	29.50	30.00
28	27.75	29.75	32.25	29.25	31.50	35.50
30	30.65	32.63	34.63	31.75	33.75	37.75
32	32.75	34.75	36.75	34.00	36.00	40.00
34	35.00	37.00	39.25	36.25	38.25	42.25
36	37.00	39.00	41.25	37.25	39.25	44.25
38	39.00	41.00	43.50	40.75	42.75	47.25
40	41.25	43.25	45.50	43.25	45.25	49.25

Table 5.9. Spiral-Wound Gasket Dimensions for API Standard 605 Flanges (inches), ID--Gasket Inside Diameter; OD--Gasket Outside Diameter; RD--Center Ring Diameter. (Extracted from Table A-5 in [API 601])

(APPENDIX ONE)

Gasket Material	m factor	y factor
Elastomers with cotton fabric insertion	1.25	400
Elastomers with cotton fabric insertion 3-ply	2.25	2200
Elastomers with cotton fabric insertion 2-ply	2.5	2900
Elastomers with cotton fabric insertion 1-ply	2.75	3700
Spiral-wound metal asbestos filled carbon	2.5	10000
Spiral-wound metal asbestos filled stainless or monel	3.0	10000
Corrugated metal asbestos inserted-iron or soft steel	3.0	4500
Corrugated metal asbestos inserted-monel or 4-6% chrome	3.25	5500
Corrugated metal asbestos inserted stainless steel	3.5	6500
Grooved metal-stainlesssteel	4.25	10100
Solid flat metal-iron or soft steel	5.5	18000

Figure 5.10. Selected Gasket m and y Factors
(Extracted from [ASME Code a])

Allowable Bolt Stress, ksi (Multiply by 1000 to abtain psi), for Metal Temperature, degrees Fahrenheit, Not Exceeding.										
Spec. No.	100	200	300	400	500	600	650	700	750	800
High Alloy Steel										
SA-193	21.2	21.2	21.2	21.2	21.2	21.2	21.2	21.2	21.2	19.5
SA-193	18.8	16.7	15.0	13.8	12.9	12.1	12.0	11.8	11.5	11.2
SA-193	18.7	16.5	14.4	12.9	12.0	11.4	11.2	11.0	10.8	10.6
SA-193	18.8	17.9	16.4	15.5	15.0	14.3	14.1	13.8	13.7	13.6
SA-193	18.7	17.7	15.6	14.3	13.3	12.6	12.3	12.1	11.9	11.7
SA-193	18.7	17.8	16.3	15.2	14.2	13.4	13.1	12.8	12.5	12.3
SA-193	18.7	17.8	16.5	15.3	14.3	13.5	13.3	12.9	12.7	12.5

Table 5.11. Allowable Bolt Stress Values in Tension for Ferrous Bolting Materials For Use with Flanges Designed (Extracted from [ASME Code b]).

Nonimal Size, in.	F (w.a.f.), in.	G (w.a.c.), in.	H (height), in.	Lt (t.l. for b.l.), in. =<6 in.	>6 in.
1/2	3/4	0.866	11/32	1.250	1.500
5/8	15/16	1.083	27/64	1.500	1.750
3/4	1 1/8	1.299	1/2	1.750	2.000
7/8	1 5/16	1.516	37/64	2.000	2.250
1	1 1/2	1.732	43/64	2.250	2.500
1 1/8	1 11/16	1.949	3/4	2.500	2.750
1 1/4	1 7/8	2.165	27/32	2.750	3.000
1 3/8	2 1/16	2.382	29/32	3.000	3.250
1 1/2	2 1/4	2.598	1	3.250	3.500
1 3/4	2 5/8	3.031	1 5/32	3.750	4.000
2	3	3.464	1 11/32	4.250	4.500
2 1/4	3 3/8	3.897	1 1/2	4.750	5.000
2 1/2	3 3/4	4.330	1 21/32	5.250	5.500
2 3/4	4 1/8	4.763	1 13/16	5.750	6.000
3	4 1/2	5.196	2	6.250	6.500
3 1/4	4 7/8	5.629	2 3/16	6.750	7.000
3 1/2	5 1/4	6.062	2 5/16	7.250	7.500
3 3/4	5 5/8	6.495	2 1/2	7.750	8.000
4	6	6.928	2 11/16	8.250	8.500

Table 5.12. Hex Bolt Dimensions (Extracted from ANSI B18.2.1-1972, as published by ASME).
w.a.f.--width across flats; w.a.c.--width across corners;
t.l.--thread length; b.l.--bolt lengths.

APPENDIX TWO

(Result No. 1)

'From Objectworks for Smalltalk-80(tm), Version 2.5 of 29 July 1989 on 3 November 1992 at 1:52:22 pm'!

Joint Design Process Protocol
(3 November 1992 12:16:00 pm)
UNITS: English System:
Temperature: Fahrenheit;
Force: pound;
Stress: ksi;
Length: inches.

Design working conditions are:

DesignT=250
AmbientT=0
DesignP=800
NominalPipeSize=26

The following is to find loads:
ClassRating= 400
Operation Load, Wm1= 590336.0
Gasket Seating Load, Wm2= 362131.0

Number of Bolts = 28
Bolt Stress at DesignT= 18.7
Bolt Stress at AmbientT= 14.4

Sa in psi = 18700.0
Sb in psi = 14400.0

am1= Wm1/Sb = 40.9956
am2= Wm2/Sa = 19.3653
Greater value = 40.9956

crossSectPerBolt= 1.46413
calculatedBoltDia= 1.3657
selectedBoltDia= 1.375

The boltDiameter defined by flange = 1.375
The boltDiameter found in design = 1.375

The Bolt selected is proper for installing on the Flange defined in earlier process.

Here are design results:

Parts List :

Selected Bolt:
material: sa193C; grade: 2; type: Hexagonal; threadType: coarse; stress: 14.4; diameter: 1.375; length: 10.0; cost: 443.114

Selected Flange:
material: CarbonSteel; type: WeldingNeckFlange; classRating: 400; nominalPipeSize: 26; cost: 125.5
dimension:
flangeThickness: 3.5; flangeOD: 33.5; raisedFaceDiameter: 28.0; boltCircleDiameter: 30.75

Selected Gasket:
material: spiral-wound1; type: spiralWound; classRating: 400; nominalPipeSize: 26; cost: 80; m factor: 2.5; y factor: 10000
dimension:
innerDiameter: 26.25; outerDiameter: 27.5; ringDiameter: 29.38

The Following is a Cost Evaluation Results:
UNIT: Dollar

PartsCost = 648.61
FlangeCost = 125.5
GaskCost = 80.0
BoltsCost = 443.11
LaborCost = 50.0
EquipCost = 120.25
Build Up Cost = 818.86
Mainten Cost = 403.99
TotalDesignCost == 1222.85

Statistics END

!

(APPENDIX TWO)

(Result No. 2)

'From Objectworks for Smalltalk-80(tm), Version 2.5 of 29 July 1989 on 3 November 1992 at 1:51:50 pm'!

Joint Design Process Protocol
(3 November 1992 10:28:47 am)
UNITS: English System:
Temperature: Fahrenheit;
Force: pound;
Stress: ksi;
Length: inches.

Design working conditions are:

DesignT=450
AmbientT=30
DesignP=700
NominalPipeSize=28

The following is to find loads:
ClassRating= 400
Operation Load, Wm1= 586801.0
Gasket Seating Load, Wm2= 389325.0

Number of Bolts = 24
Bolt Stress at DesignT= 18.7
Bolt Stress at AmbientT= 12.0

Sa in psi = 18700.0
Sb in psi = 12000.0

am1= Wm1/Sb = 48.9001
am2= Wm2/Sa = 20.8195
Greater value = 48.9001

crossSectPerBolt= 2.0375
calculatedBoltDia= 1.61107
selectedBoltDia= 1.625

The boltDiameter defined by flange = 1.5
The boltDiameter found in design = 1.625

The Bolt from the design process is not proper for the flange, re-run is necessary!!

****This is re-run process: ****

The following is to find loads:
ClassRating= 400
Operation Load, Wm1= 586801.0
Gasket Seating Load, Wm2= 389325.0

Number of Bolts = 24
Bolt Stress at DesignT= 18.8
Bolt Stress at AmbientT= 15.0

Sa in psi = 18800.0
Sb in psi = 15000.0

am1= Wm1/Sb = 39.1201
am2= Wm2/Sa = 20.7088
Greater value = 39.1201

crossSectPerBolt= 1.63
calculatedBoltDia= 1.44098
selectedBoltDia= 1.5

The boltDiameter defined by flange = 1.5
The boltDiameter found in design = 1.5

The Bolt selected is proper for installing on the Flange defined in earlier process.

Here are design results:

Parts List :

Selected Bolt:
material: sa193D; grade: 2; type: Hexagonal; threadType: coarse; stress: 15.0; diameter: 1.5; length: 10.5; cost: 473.556

Selected Flange:
material: CarbonSteel; type: WeldingNeckFlange; classRating: 400;

nominalPipeSize: 28; cost: 135.5
dimension:
flangeThickness: 3.75; flangeOD: 36.0; raisedFaceDiameter: 30.0;
boltCircleDiameter: 33.0

Selected Gasket:
material: spiral-wound1; type: spiralWound; classRating: 400;
nominalPipeSize: 28; cost: 85; m factor: 2.5; y factor: 10000
dimension:
innerDiameter: 28.13; outerDiameter: 29.5; ringDiameter: 31.5

The Following is a Cost Evaluation Results:
UNIT: Dollar

PartsCost = 694.06
FlangeCost = 135.5
GaskCost = 85.0
BoltsCost = 473.56
LaborCost = 50.0
EquipCost = 120.25
Build Up Cost = 864.31
Mainten Cost = 403.99
TotalDesignCost == 1268.3

Statistics END

!

(APPENDIX TWO)

(Result No. 3)

'From Objectworks for Smalltalk-80(tm), Version 2.5 of 29 July 1989 on 3 November 1992 at 1:52:33 pm'!

Joint Design Process Protocol
(3 November 1992 12:18:47 pm)
UNITS: English System:

Temperature:	Fahrenheit;
Force:	pound;
Stress:	ksi;
Length:	inches.

Design working conditions are:

DesignT=350
AmbientT=60
DesignP=1200
NominalPipeSize=26

The following is to find loads:
ClassRating= 600
Operation Load, Wm1= 922630.0
Gasket Seating Load, Wm2= 370697.0

Number of Bolts = 28
Bolt Stress at DesignT= 18.7
Bolt Stress at AmbientT= 12.9

Sa in psi = 18700.0
Sb in psi = 12900.0

am1= Wm1/Sb = 71.5217
am2= Wm2/Sa = 19.8234
Greater value = 71.5217

crossSectPerBolt= 2.55435
calculatedBoltDia= 1.80387

selectedBoltDia= 1.875

The boltDiameter defined by flange = 1.625
The boltDiameter found in design = 1.875

The Bolt from the design process is not proper for the flange, re-run is necessary!!

****This is re-run process: ****

The following is to find loads:
ClassRating= 600
Operation Load, Wm1= 922630.0
Gasket Seating Load, Wm2= 370697.0

Number of Bolts = 28
Bolt Stress at DesignT= 18.8
Bolt Stress at AmbientT= 13.8

Sa in psi = 18800.0
Sb in psi = 13800.0

am1= Wm1/Sb = 66.8572
am2= Wm2/Sa = 19.7179
Greater value = 66.8572

crossSectPerBolt= 2.38776
calculatedBoltDia= 1.74406
selectedBoltDia= 1.75

The boltDiameter defined by flange = 1.625
The boltDiameter found in design = 1.75

The Bolt from the design process is not proper for the flange, re-run is necessary!!

****This is re-run process: ****

The following is to find loads:
ClassRating= 600
Operation Load, Wm1= 922630.0
Gasket Seating Load, Wm2= 370697.0

Number of Bolts = 28

Bolt Stress at DesignT= 21.2
Bolt Stress at AmbientT= 21.2

Sa in psi = 21200.0
Sb in psi = 21200.0

am1= Wm1/Sb = 43.5203
am2= Wm2/Sa = 17.4857
Greater value = 43.5203

crossSectPerBolt= 1.5543
calculatedBoltDia= 1.40712
selectedBoltDia= 1.5

The boltDiameter defined by flange = 1.625
The boltDiameter found in design = 1.5

The Bolt from the design process is not proper for the flange, re-run is necessary!!

****This is re-run process: ****

The following is to find loads:
ClassRating= 600
Operation Load, Wm1= 922630.0
Gasket Seating Load, Wm2= 370697.0

Number of Bolts = 28

Not easy to choose a bolt to match with the flange selected, by using a few material alternatives. If many material alternatives are available, a good match is possible. Otherwise, suggest to reconsider the gasket selection, then, the flange. Forced stop here.

!