INFORMATION TO USERS

While the most advanced technology has been used to photograph and reproduce this manuscript, the quality of the reproduction is heavily dependent upon the quality of the material submitted. For example:

- Manuscript pages may have indistinct print. In such cases, the best available copy has been filmed.

- Manuscripts may not always be complete. In such cases, a note will indicate that it is not possible to obtain missing pages.

- Copyrighted material may have been removed from the manuscript. In such cases, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, and charts) are photographed by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each oversize page is also filmed as one exposure and is available, for an additional charge, as a standard 35mm slide or as a 17"x 23" black and white photographic print.

Most photographs reproduce acceptably on positive microfilm or microfiche but lack the clarity on xerographic copies made from the microfilm. For an additional charge, 35mm slides of 6"x 9" black and white photographic prints are available for any photographs or illustrations that cannot be reproduced satisfactorily by xerography.
Vemuri, Koteswara Rao

A KNOWLEDGE-BASED APPROACH TO AUTOMATE GEOMETRIC DESIGN WITH APPLICATION TO DESIGN OF BLOCKERS IN THE FORGING PROCESS

The Ohio State University Ph.D. 1986

University Microfilms International 300 N. Zeeb Road, Ann Arbor, MI 48106
PLEASE NOTE:

In all cases this material has been filmed in the best possible way from the available copy. Problems encountered with this document have been identified here with a check mark √.

1. Glossy photographs or pages ______
2. Colored illustrations, paper or print ______
3. Photographs with dark background ______
4. Illustrations are poor copy ______
5. Pages with black marks, not original copy ______
6. Print shows through as there is text on both sides of page ______
7. Indistinct, broken or small print on several pages __√__
8. Print exceeds margin requirements ______
9. Tightly bound copy with print lost in spine ______
10. Computer printout pages with indistinct print ______
11. Page(s) ________ lacking when material received, and not available from school or author.
12. Page(s) __________ seem to be missing in numbering only as text follows.
13. Two pages numbered _______. Text follows.
14. Curling and wrinkled pages ______
15. Dissertation contains pages with print at a slant, filmed as received ______
16. Other ____________________________
    ____________________________________________________________________________
    ____________________________________________________________________________

University
Microfilms
International
A KNOWLEDGE-BASED APPROACH TO AUTOMATE GEOMETRIC DESIGN
WITH APPLICATION TO DESIGN OF BLOCKERS
IN THE FORGING PROCESS

DISSERTATION

Presented in Partial Fulfillment of the Requirements for
the Degree Doctor of Philosophy in the Graduate
School of the Ohio State University

By

Koteswara Rao Vemuri, M.S.

* * * * *

The Ohio State University

1986

Dissertation Committee:

Richard A. Miller

Taylan Altan

Amit Bagchi

Approved by

[Signature]

Adviser
Department of Industrial and Systems Engineering
To My Wife and Parents
ACKNOWLEDGEMENT

I wish to express my gratitude and appreciation to my adviser Dr. Richard A. Miller, for his constant guidance throughout my graduate studies at The Ohio State University.

I am also deeply indebted to Dr. Taylan Altan, co-adviser, for his continued involvement and suggestions during the course of this research, and Dr. Soo-Ik Oh, for giving me the opportunity and encouragement to work on this research topic. Thanks are also due to Dr. Amit Bagchi, member of the reading committee, for his valuable suggestions.

This dissertation is based on a study being conducted at Battelle Columbus Division, Columbus, Ohio, under the sponsorship of National Science Foundation (Grant No. MEA-8411956, with Dr. W. M. Spurgeon as the Program Manager at the initiation of the project and Dr. John E. Mayer, Jr. currently) and Air Force Materials Laboratory (Contract No. F33615-83-C-5052 with Dr. H. L. Gegel as the Program Manager at the initiation of the project and Mr. James Malas currently). The collection and classification of design knowledge is being done with NSF support, with partial support from the Air Force. Development of the computer program itself is being supported by the Air Force, as part of the Manufacturing Science Program to develop a Computer-Aided Engineering (CAE) System for Die (Mold) Design and Manufacturing. I would like to gratefully acknowledge these supports.
VITA

January 30, 1954  ..........  Born - India

1975  ............................  B.Tech. (Mech. Eng.), Indian Institute of Technology, Madras, India

1975–1977  .................  Diploma in Ind. Eng., NITIE, Bombay, India

1977–1979  ..................  Management Trainee, Garlick Eng., Bombay, India

1979–1980  ..................  University Fellowship, The Ohio State University, Columbus, Ohio

1980–1983  ..................  Graduate Research Assistant, Department of Industrial and Systems Engineering, The Ohio State University, Columbus, Ohio

1983–present  ..............  Researcher, Engineering and Manufacturing Technology Dept., Battelle Columbus Division, Columbus, Ohio

PUBLICATIONS


FIELDS OF STUDY

Major Field:  Systems Theory
Faculty Member - Dr. Richard A. Miller

Minor Field:  Computer Aided Design / Computer Aided Manufacturing
Faculty Member - Dr. Richard A. Miller
Dr. Taylan Altan

Minor Filed:  Artificial Intelligence
Faculty Member - Dr. B. Chandrasekaran

Minor Field:  Computer and Information Science (Graphics/Databases)
Faculty Member - Dr. Edwin Tripp
# TABLE OF CONTENTS

Dedication ............................................................................................................ ii  
Acknowledgement ................................................................................................ iii  
Vita ........................................................................................................................ iv  
Table of Contents ............................................................................................. vi  
List of Tables ...................................................................................................... ix  
List of Figures .................................................................................................... x  

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 The Research Problem</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Research Objective</td>
<td>10</td>
</tr>
<tr>
<td>1.3 Dissertation Organization</td>
<td>13</td>
</tr>
<tr>
<td>2. STATE OF THE ART</td>
<td>14</td>
</tr>
<tr>
<td>2.1 Knowledge-Based Approach to Mechanical Design</td>
<td>14</td>
</tr>
<tr>
<td>2.1.1 Problem Solving in Design</td>
<td>15</td>
</tr>
<tr>
<td>2.1.2 Mechanical Design Applications</td>
<td>17</td>
</tr>
<tr>
<td>2.2 Design of Blocker Forgings</td>
<td>19</td>
</tr>
<tr>
<td>2.2.1 Blocker Design Knowledge</td>
<td>20</td>
</tr>
<tr>
<td>2.2.2 Computerization of Blocker Design</td>
<td>22</td>
</tr>
<tr>
<td>2.2.2.1 Design using Interactive Graphics</td>
<td>22</td>
</tr>
<tr>
<td>2.2.2.2 Automatic Design</td>
<td>23</td>
</tr>
<tr>
<td>3. PROPOSED KNOWLEDGE BASE ARCHITECTURE</td>
<td>28</td>
</tr>
<tr>
<td>3.1 Problem Solving</td>
<td>28</td>
</tr>
<tr>
<td>3.1.1 Problem Solving in Geometric Design</td>
<td>29</td>
</tr>
<tr>
<td>3.1.2 Problem Solving in Blocker Design</td>
<td>31</td>
</tr>
<tr>
<td>3.2 Knowledge Base Architecture</td>
<td>37</td>
</tr>
<tr>
<td>3.2.1 Integration of Knowledge and its Use</td>
<td>37</td>
</tr>
</tbody>
</table>
LIST OF TABLES

Table 1: Suggested Dimensions in Blocker Design. [Altan 73] 21
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1</td>
<td>An Example Finish Forging, and Some of its Cross Sections. [Akgerman 73]</td>
<td>5</td>
</tr>
<tr>
<td>Figure 2</td>
<td>A Typical Forging Cross Section and a Possible Blocker Design (Shown in Dashed Lines). [Subramanian 78]</td>
<td>6</td>
</tr>
<tr>
<td>Figure 3</td>
<td>Defect Formation in Forging when Fillet Radii are too Small. [Chamouard 64]</td>
<td>7</td>
</tr>
<tr>
<td>Figure 4</td>
<td>Typical Sequences in the Die Forging of a Rib-and-web Part. Defects Occurring from too Thin a Web. [Sabroff 68]</td>
<td>8</td>
</tr>
<tr>
<td>Figure 5</td>
<td>Types of Structural Forgings Representative of Increasing Levels of Forging Difficulty. [Sabroff 68]</td>
<td>11</td>
</tr>
<tr>
<td>Figure 6</td>
<td>Separation of a Rib-Web Cross Section into Modular L Shapes. [Subramanian 77]</td>
<td>25</td>
</tr>
<tr>
<td>Figure 7</td>
<td>Preform Design for an L Shape. [Subramanian 77]</td>
<td>25</td>
</tr>
<tr>
<td>Figure 8</td>
<td>Features and Design Parameters for a Rib-Web Type Cross Section.</td>
<td>33</td>
</tr>
<tr>
<td>Figure 9</td>
<td>CAD/CAM Procedure for Forging Die Design. [Ficke 84]</td>
<td>35</td>
</tr>
<tr>
<td>Figure 10</td>
<td>Conceptual Structure of a Rib-Web Section.</td>
<td>44</td>
</tr>
<tr>
<td>Figure 11</td>
<td>Architecture of a Design Specialist.</td>
<td>45</td>
</tr>
<tr>
<td>Figure 12</td>
<td>An Example Finish Forging Cross Section.</td>
<td>66</td>
</tr>
<tr>
<td>Figure 13</td>
<td>Illustration of the Tangency Condition.</td>
<td>68</td>
</tr>
<tr>
<td>Figure 14</td>
<td>External Representation for the Example Forging Cross Section Shown in Figure 12.</td>
<td>69</td>
</tr>
</tbody>
</table>
Figure 15: Hierarchic Representation of Geometry in PROLOG. ................................. 72
Figure 16: PROLOG Structures to Describe a Section Geometry Internally. ...................... 73
Figure 17: Classification in Terms of Ribs and Webs, in PROLOG Implementation. .................. 74
Figure 18: Description in Terms of Design Parameters, in PROLOG Implementation. ................ 74
Figure 19: Internal Representation in PROLOG form, for the Example Forging Cross Section Shown in Figure 12. ............................................................ 75
Figure 20: Classification in Terms of Ribs and Webs, in LISP Implementation. ....................... 77
Figure 21: Convention for Naming Features and Edges in LISP Implementation. ..................... 78
Figure 22: Naming Features and Edges for the Example Forging Cross Section Shown in Figure 12. ............................................................ 79
Figure 23: Naming Features and Edges for a Relatively Complex Forging Cross Section. ................ 79
Figure 24: Frame Hierarchy for Representing Geometry Internally. ...................................... 80
Figure 25: Some Example Frames Illustrating the Internal Representation in LISP, for the Cross Section Shown in Figure 12. ............................................................ 81
Figure 26: Sample High Level Topological Queries About Geometry. ...................................... 84
Figure 27: Example Functions for Some High Level Topological Queries About Geometry. ........ 85
Figure 28: Rib–Web Type Parts, with Multiple Steps Near the Flash Land. .............................. 90
Figure 29: Convention for Symbolic Description of a Surface, Based on the Slopes of its Edges. ............................................................ 91
Figure 30: Symbolic Description of a Surface, Based on the Slopes of its Edges. ...................... 93
Figure 31: Symbolic Description of a Relatively Complex Surface, Based on the Slopes of its Edges. ............................................................ 93
Figure 32: Surface Description Grammar in Backus–Naur Form. ......................................... 94
Figure 33: Parsing Rib–Web Ordered Pairs from the Surface Patterns, for the Surfaces Shown in Figures 30 and 31. ........................... 95
Figure 34: Input–Output View of BID. .................................................... 98
Figure 35: Functional Relationships in BID. ......................................... 99
Figure 36: Architecture of BID, Indicating Module Interaction. .......... 101
Figure 37: BID Implementation Methodology. ..................................... 103
Figure 38: Architecture of the BID Design Module. ............................ 106
Figure 39: Variable Notation for Volumes. ........................................... 114
Figure 40: An Example Design Rule to Recommend Blocker Rib Height. . 123
Figure 41: An Example Design Rule to Recommend Blocker Draft Angle. . 124
Figure 42: Menu Options Provided in BID. .......................................... 127
Figure 43: Screen Display when BID is Executed in Automated Mode. . 130
Figure 44: Intermediate Stage After Some Ribs are Designed by BID. . 135
Figure 45: Intermediate Stage After All the Ribs and Some Webs are Designed by BID. .................................................. 135
Figure 46: Blocker Design for a Simple Forging Cross Section, Assuming Titanium Forging and Hydraulic Press. ............................. 136
Figure 47: Blocker Design for a Simple Forging Cross Section, Assuming Titanium Forging and Hammer. ......................................... 137
Figure 48: Blocker Design for a Simple Forging Cross Section, Assuming Steel Forging and Mechanical Press. ................................. 138
Figure 49: Blocker Design for a Simple Forging Cross Section, Assuming Aluminum Forging and Hydraulic Press. .......................... 139
Figure 50: Blocker Design for the Forging Cross Section Shown in Figure 12, Assuming Titanium Forging and Hydraulic Press. .............. 142
Figure 51: Blocker Design for the Forging Cross Section Shown in Figure 12, Assuming Steel Forging and Mechanical Press. ............... 143
Figure 52: Blocker Design for the Forging Cross Section Shown in Figure 12, Assuming Aluminum Forging and Hammer. ..................... 144
Figure 53: Mismatch in Neighboring L-shapes of the Blocker Sections. [Subramanian 77] .......................... 149

Figure 54: Closed-die Forging with Flash. ................................. 176

Figure 55: Suggested Preform Cross Sections for Various H-Shape Forgings in Steel. [Lange 77] ................................. 177

Figure 56: Preforming, Blocking, and Finishing Operations for an Example Forging. [Haller 71] ................................. 179

Figure 57: Planes and Directions of Metal Flow During Forging of Two Simple Shapes: (a) Planes of Flow, (b) Finish Forged Shapes, and (c) Directions of Flow. [Altan 73] ...................... 180

Figure 58: Planes and Directions of Metal Flow During Forging of a Relatively Complex Shape: (a) Planes of Flow, (b) Finish Forged Shapes, and (c) Directions of Flow. [Altan 73] ...................... 180

Figure 59: Relationships Between Process and Machine Variables in Hot Forming Processes Conducted in Presses. [Altan 73] ...................... 182

Figure 60: Classification of Forging Shapes. [Spies 59] ...................... 184

Figure 61: Outline of an Integrated CAD/CAM Approach for Hot Forging. [Altan 83] ...................... 188
CHAPTER 1

INTRODUCTION

1.1 THE RESEARCH PROBLEM

In the area of manufacturing, ever increasing computing power has spurred substantial interest in utilizing computers to improve productivity, which in turn has led to the phenomenal growth in computer-aided-design/computer-aided-manufacturing (CAD/CAM) applications. This is especially true of the discrete goods manufacturing industries, where design and planning activities are rendered more complex due to the dynamism induced by the ever changing products and product-mix. Even while computer applications supporting various stages of production in these industries are mushrooming, it is widely recognized that the first step towards complete automation (for the so called unmanned factory) requires full automation of the individual design and manufacturing activities. These include activities such as process planning, design, and numerical control (NC) programming. Among them, Design Automation, a critical and a predominantly manual task, poses a great challenge, in terms of theoretical as well as developmental efforts.
Knowledge-Based Approach to Design Automation: Computer applications typically require significant human interaction. They essentially provide analytical capabilities through techniques such as the Finite Element Method (FEM), and a quick and easy means for manipulating geometry through interactive graphics. These geometric manipulations are no different from what is done manually, but with the aid of computers they can be performed faster and more accurately. Thus, in spite of these impressive developments, the creative design and planning decisions are still left to the engineer to be carried out manually, with the computer merely providing an electronic sketch pad to try out the various design and planning ideas quickly, of course, aided by its vast storage and computational capabilities.

Medland [Medland 86] characterizes this by grouping the existing CAD/CAM applications into three main design phases, viz., primary design (conceptual), secondary design (Analysis: stress-analysis, kinematics etc.) and tertiary design (Manufacturing information: drafting, NC tape production etc.), and suggests that the CAD/CAM applications tended to concentrate exclusively on the secondary and tertiary phases of design, leaving conceptual design essentially to the design engineer. This is to be expected, since conceptual design is the least understood of all phases of design. However, the natural progression in this computerization trend is to try to develop systems which are capable of performing these creative decision making tasks also as far as possible, and a truly automated design system needs to be based on an understanding of the conceptual activities in this phase.
This involves complex problem solving, as yet ill understood. Hence, given that Knowledge-Based Systems (KBS) address computerization of complex, knowledge-intensive tasks, it is a natural discipline to study mechanization of design.

The Geometric Design Problem: This dissertation is concerned with applying the KBS approach for automating a particular class of mechanical design problems which deal with geometry design. While the concept of geometric design is illustrated later on, in brief, it involves modification of the geometry of a given object, subject to a variety of design constraints. The design constraints themselves are dependent on the particular problem under consideration. The geometric design problem can be found in a variety of mechanical design situations. In particular, many metal forming problems fall under this category, and an understanding of the generic design issues appropriate for these problems would be highly beneficial. Specifically, the geometric design problem is studied here in the context of the blocker design problem, encountered in the process of forging die design. The blocker geometry design problem thus provides a means for studying the generic issues in applying the KBS approach to geometric design.

The Blocker Geometry Design Problem: In the forging process called closed-die forging process with flash, design of blockers is of critical importance.

1. An overview of the design process in forging is included in Appendix-A.
2. This particular forging process is of relevance for the blocker design problem being considered in this dissertation.
It involves three dimensional (3D) geometry design. However, common practice in the design of blocker geometries is to consider planes of metal flow, i.e., certain cross sections of the forging. For a given forging cross section, the corresponding blocker cross section is then determined by altering the geometry of the forging cross section utilizing general guidelines/rules for blocker design, and making sure that area of the blocker cross section is equal to (or slightly larger than) the forging cross section.

For example, Figure 1 shows a forging and some of its cross sections. Blocker design for this forging can be obtained by designing blocker geometries along these sections, and then blending these blocker cross sections. In fact, this reflects the commonly followed procedure in the forging industry. Thus, the 3D blocker design problem can be considered as a two dimensional (2D) geometry design problem. Accordingly, further discussion in this dissertation is concerned with the 2D geometry design problem. Figure 2 illustrates this 2D blocker design problem by showing a typical finish forging cross section, and a possible blocker cross section for this forging cross section.

Design of blocker and preform geometries is the most critical part of forging die design, and it is a highly skill intensive and experience based activity. In closed die forging with flash, there are two general reasons for using a blocker. The first is when the shape of the final forging to be made is of such contour that the material will not or cannot flow smoothly or evenly, to fill the finish
impression adequately. The second reason is when the number of forgings to be made is of such magnitude, the finisher die will be subject to excessive wear before the required forgings can be made.

Hence, proper blocker configuration is necessary to eliminate the possibility of forming defects during forging. For example, Figure 3 illustrates how lap defects
can form because of fillet radii being too small, even though the die is adequately filled. Here, the material has continued to extrude from a web area into a rib, even after an upsetting action has begun in the rib. Continued extrusion causes the material to fold over on itself, leaving a defect at the base of the rib. Figure 4 illustrates the formation of laps in forging a preform which has too thin a web. In forming the ribs, metal normally flows towards the center, causing the web to increase in thickness. If the web in the preform is too thin, it will buckle, causing laps in the web. With proper design, metal can be adequately distributed within the blocker, to achieve the following objectives [Altan 83]:

Figure 2: A Typical Forging Cross Section and a Possible Blocker Design (Shown in Dashed Lines). [Subramanian 78]
Figure 3: Defect Formation in Forging when Fillet Radii are too Small. [Chamouard 64]

- Fill the finisher cavity without any forging defects.
- Reduce the amount of material lost as forging flash.
Reduce die wear by minimizing metal movement in the finisher die.

Provide the required amounts of deformation and grain flow so that desired mechanical properties are obtained.
From the above discussion, it is clear that determination of proper blocker configuration is a very difficult task, and is an art by itself requiring skills achieved only by years of experience. Although there are several publications in the technical literature on design of finish forgings, very little quantitative information on preform/blocker design has been published. Most of the time, die designers are forced to use trial and error (i.e., make a blocker die and try it), before they can achieve a satisfactory blocker design. As a result, extensive die try-outs are necessary, and expensive productive machine capacity and man-hours are spent in developing the blocker dies.

At present, Computer-Aided Design (CAD) of blocker cross sections can be carried out using interactive graphics. However, it is still the designer who provides the actual design expertise. A natural progression is to try to computerize this design decision making, as much as possible. Hence, it is proposed to study this process of designing blocker geometries, and to develop a knowledge-based system incorporating the design heuristics used in blocker design.
1.2 RESEARCH OBJECTIVE

As discussed above, a natural trend for the CAD/CAM applications is to move towards computerization of the problem solving/decision making capabilities required for design. Accordingly, the objective here is to analyze the blocker geometry design problem so as to develop a model of the design process, and an architecture reflecting this model for effective computer implementation. In developing the system architecture, the research also aims at abstracting out the principles valid even for the generic geometric design problem.

With respect to the blocker design problem, considering the complexity and the variety of factors involved in designing blocker geometries, it was decided to concentrate the research work on a specific family of forgings. Since most aerospace parts are of the structural type, i.e., rib-web type, as shown in Figure 5, there is serious interest in these types of forgings due to their large economic significance. Accordingly, only forgings with rib-web type cross sections are considered in the present investigation. It is expected that later the results and the methodology can be expanded to handle other types of forgings as well. Thus, the objective of the blocker design task can be briefly stated as: "Design of blocker cross sections for rib-web type parts, under closed-die forging with flash." The computer program being developed for the design of blocker geometries is named Blocker Initial-guess Design (BID). This dissertation summarizes the basic structure of BID and the development so far.
Figure 5: Types of Structural Forgings Representative of Increasing Levels of Forging Difficulty. [Sabroff 68]

Automation versus Assistance: While the objective of this knowledge-based approach is to computerize the design problem solving/decision-making capabilities of the human designer, it should be emphasized that the resulting computer
program is not expected to be fully autonomous; it is expected to require supervision and guidance from the human designer. Thus, ultimately, design is to be performed by the machine, but guided by the human designer [Scherlis 83, Balzer 81]. This human-aided machine design paradigm is in contrast to the machine-aided design (that is, CAD) paradigm, where it is the human who performs the design, with the machine merely providing the supporting tools without understanding the design decisions.

Changing the scope of the problem as above, from automation to assistance, does not alter the decision making skills required by the program. Instead, at the outset, this approach explicitly recognizes the relative complexity of the design problem space, and makes provision for the human being in the design loop to aid the system in more complex situations. That is, the designer can monitor the system behavior, and modify the response as and when necessary. Of course, at least conceptually, more and more complex situations could be programmed into the computer system, so that it could become relatively more self-reliant and autonomous. In the context of BID, while it is programmed so that it could perform blocker design in a fully automated mode, it is expected that it will be normally run only in an interactive mode, as discussed in detail in Chapter 5.
1.3 DISSERTATION ORGANIZATION

Chapter 2 provides an overview of the state of the art research in blocker design, and KBS approach to mechanical design. Chapter 3 then discusses the knowledge base architecture proposed for geometric design, in particular for the blocker design problem. This chapter also makes comparative evaluation of the suggested approach with relevant previous work on the KBS approach to mechanical design.

The geometry representation and recognition issues are discussed in Chapter 4. This is followed in Chapter 5 by a description of the BID system of computer programs, and a comparison of BID with similar efforts in blocker design. Finally, Chapter 6 concludes with a summary of contributions and recommendations for further research.
CHAPTER 2
STATE OF THE ART

This chapter discusses briefly the state of the art in (a) knowledge-based systems approach to mechanical design, and (b) design of blocker forgings. The objective of this discussion is only to provide a platform for detailed discussion in subsequent chapters; not a comprehensive review of these areas. Accordingly, no attempt is made for a general discussion on either the forging technology or the KBS approach. However, to provide a better perspective on the design problem at hand, a brief overview of the design process in forging is included in Appendix-A.

2.1 KNOWLEDGE-BASED APPROACH TO MECHANICAL DESIGN

Within the Artificial Intelligence (AI) discipline, based on the recognition that general-purpose problem solving strategies are too weak to solve most complex problems [Newell 69], and that the problem solving power is primarily derived from knowledge [Feigenbaum 77], development of knowledge-based systems has of late become an active area of research. Hence the slogan, "In Knowledge Lies the
Knowledge-engineering applications can be broadly classified [Hayes-Roth 83] into the following generic categories.

- **Interpretation**: Inferring situation descriptions from sensor data
- **Predictions**: Inferring likely consequences of given situations
- **Diagnosis**: Inferring system malfunctions from observables
- **Design**: Configuring objects under constraints
- **Planning**: Designing actions
- **Monitoring**: Comparing observations to plan vulnerabilities
- **Debugging**: Prescribing remedies for malfunctions
- **Repair**: Executing a plan to administer a prescribed remedy
- **Instruction**: Diagnosing, debugging, and repairing student behaviour
- **Control**: Interpreting, predicting, repairing, and monitoring system behaviours

### 2.1.1 Problem Solving in Design

Among these categories, diagnosis has been the most dominant paradigm in the early stages of KBS development. However, a number of researchers have of late begun the study of design problems also. These systems are concerned with developing configurations of objects that satisfy the constraints of the design problem. However, design automation is still at an early stage, since the design process is poorly understood. The key research problem here is developing better models of design. A comprehensive model of design so developed should address the following aspects of the design process [Mostow 85].
1. The state of the design. Design involves a series of artifact descriptions at various levels of detail.

2. The goal structure of the design process. If design is a purposive activity, goals guide the choice of what to do at each point. These goals are not artifact descriptions, but prescribe how those descriptions should be manipulated.

3. Design decisions. Given a goal, there may be several plans for achieving it. Design decisions represent choices among them.

4. Rationales for design decisions. The rationale for choosing a particular plan to achieve a goal explains why the plan is expected to work and why it was selected instead of the alternatives.

5. Control of the design process. Guiding design requires choosing which goal to work on at each point and choosing which plan to achieve it with.

6. The role of learning in design. Solving a design problem requires both general knowledge about the domain and specific knowledge about the problem. Learning is a way to acquire such knowledge.

The MOLGEN [Stefik 81] system, concerned with the design of gene cloning experiments, is one of the first KBS related to design. Although it lies on the borderline between design and planning, its relevance to design is based on how it divides the problem into almost independent subproblems, using constraint-posting to communicate limitations between the subproblems. In the context of chemical and civil engineering applications, Rychener discusses properties of the design problems and the steps to be taken in solving them [Rychener 83]. He also stresses the importance of allowing access to existing analytical tools of the various engineering disciplines. In electrical design, there are attempts to apply knowledge—
based approach for problems such as VLSI design [Mitchell 85]. In line with these developments, there is growing interest in applying KBS approach even to the mechanical design problems, as discussed below.

2.1.2 Mechanical Design Applications

While discussing mechanical design problems in general, Brown and Chandrasekaran classify them into roughly three categories [Brown 83]. Class 1 design, performed very rarely, is concerned with major inventions. It requires knowledge in a wide range of domains. Class 2 design is closer to the routine, but some aspects may require substantial innovation. Class 3 design is the most routine, following a set of relatively well established design alternatives, but still requiring an expert to perform the design task. Rasdorf names these three classes as problems requiring Creative knowledge, Innovative knowledge, and Routine knowledge [Rasdorf 85]. In reality, design knowledge can not be classified so rigidly; this classification merely simplifies what is really a spectrum from the most open ended to the most routine. However, it illustrates the levels of complexity in the design knowledge required to perform different design tasks. In light of the above discussion, most design problems currently attempted using the KBS approach can be considered closer to the routine design level.

The VEXPERT system [Dixon 84], developed for designing standard V-belt drives, represents one of the early attempts in applying KBS approach to mechanical design. Its design-evaluate-redesign architecture is based on an iterative
model of the design process, where a completed design is improved in successive stages of iteration. It uses (i) an algorithm to obtain initial design from the problem specifications, (ii) utility-decision theory for evaluation and acceptability decision, and (iii) production-rules written in OPS-5 [Forgy 77] for redesign. The redesign task itself is accomplished using independent knowledge sources and a blackboard. However, this design-evaluate-redesign approach is impractical for most design problems, since difficulty in evaluating the designs renders the approach computationally intractable on any but the simplest design problems.

A more powerful model based on design refinement was suggested [Brown 86] in the context of AIR-CYL, a system for designing Air Cylinders. This model addresses the knowledge representation issues relevant for routine design. Its architecture is based on a hierarchically organized community of design agents called specialists, and the design activity is divided into (i) requirements phase, (ii) rough design phase, (iii) design phase, and (iv) redesign phase. The specialists choose from relevant plans, make necessary commitments, and direct those at lower levels of abstraction to refine the design. Thus, design proceeds in increasing levels of specificity, with the least possible commitment being made at each level.

A more complex design task, requiring decisions about geometry, spatial layout, timing, forces etc., was implemented in PRIDE, a system for designing paper handling systems [Mittal 86]. It attempts to integrate knowledge-guided search, analytical tools, and collaborative design through a community of knowledge bases
which bring together expertise from many different specialists. The problem solving in this system can be considered as following a generate-test-analyze-advice-modify paradigm. Its knowledge-base contains objects (frames) that are organized into well-defined classes with appropriate protocols of behavior, and defined in terms of design goals and design methods. A design goal is both a description of some part of the overall design, as well as a concept around which knowledge is organized, while a design method specifies how to carry out a goal. Hence, a design goal as proposed here can be thought of as an autonomous specialist in the sense of the AIR-CYL system. Knowledge about testing and validating a proposed solution is contained in the design constraints attached to the goals. The system attempts to handle constraint failures through some extensions to the dependency directed backtracking. It also provides the capability for maintaining multiple designs simultaneously and switching between different partial designs, permitting exploration of various designs in parallel.

2.2 DESIGN OF BLOCKER FORGINGS

Discussion on the state of the art in blocker design can be divided into two separate parts: (a) discussion on the blocker design knowledge, and (b) trends in computerization of blocker design. Each of these aspects is discussed below.
2.2.1 Blocker Design Knowledge

Traditionally, design of forging dies is carried out using empirical guidelines developed over years of experience, and intuition. While there are several publications in the technical literature on design of finish forgings, very little quantitative information on blocker/preform design has been published. This information is usually kept company-confidential, by most experienced die designers. Clearly, there are no analytical methods available for blocker design. Hence, discussions on blocker design in the technical literature, which attempt to cover the topic in all its rich variety, restrict themselves to qualitative design guidelines, as illustrated by the examples given below [Altan 83]:

- In plan view, the blocker is slightly narrower than the finisher, about 0.5 to 1 mm on each side, so that it can fit into the finisher die.
- The blocker usually has larger fillet and corner radii, to enhance metal distribution.
- The areas of blocker cross sections are slightly larger (1 to 3 percent) than those of finisher cross sections.
- For forging high ribs in the finisher, it is at times necessary to have lower ribs in the blocker. At the same time, the web thickness in the blocker is larger than that in the finisher.
- In forging, in order to enhance metal flow towards the ribs, it is useful to provide an opening taper from the center of the web toward the ribs.
- In steel forgings, whenever possible, the ribs in the blocker sections should be narrower but slightly higher than those in the finisher sections. This reduces die wear.
In addition, Table 1 illustrates the type of quantitative information that is occasionally available in the literature, although very rarely.

Table 1: Suggested Dimensions in Blocker Design. [Altan 73]

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Aluminum Alloys</th>
<th>Titanium Alloys</th>
<th>Steels</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web, $T_B$</td>
<td>(1-1.5) $T_F$</td>
<td>(1.2 - 2.2) $T_F$</td>
<td>(1 - 1.2) $T_F$</td>
</tr>
<tr>
<td>Fillet, $R_{FB}$</td>
<td>(1.2 - 2) $R_{FF}$</td>
<td>(2 - 3) $R_{FF}$</td>
<td>(1.2 - 2) $R_{FF}$</td>
</tr>
<tr>
<td>Corner, $R_{CB}$</td>
<td>(1.2 - 2) $R_{CF}$</td>
<td>(1.5 - 2.5) $R_{CF}$</td>
<td>(1.2 - 2) $R_{CF}$</td>
</tr>
<tr>
<td>Draft, $\alpha_B$</td>
<td>$\alpha_F = 2 - 5^\circ$</td>
<td>$\alpha_F = 5 - 7^\circ$</td>
<td>$\alpha_F = 1 - 5^\circ$</td>
</tr>
<tr>
<td>Rib, $W_B$</td>
<td>$W_F - 1/32$</td>
<td>$W_F - (1/16 - 1/8)$</td>
<td>$W_F - 1/32$</td>
</tr>
</tbody>
</table>

*NOTE: Subscript B denotes blocker die, $F$ denotes finisher die.*

For forging complex parts, empirical design guidelines as above may not suffice, and trial-and-error procedures may be too time consuming and expensive. Hence, physical modeling is often used, with a soft material such as lead, plasticine or wax as a model forging material, and hard plastic or mild steel dies as tooling. Thus, preform shapes are determined with some experimentation on relatively low-cost tooling. In summary, very little design knowledge is available in the technical literature, on design of blocker forgings.
2.2.2 Computerization of Blocker Design

Attempts to computerize blocker design have taken two different approaches. The first (and the most common) approach is to design using interactive computer graphics. The second approach, although attempted very rarely, is to have a computer program perform the design automatically. Obviously, the second approach is harder. These computerization efforts are discussed below in greater detail.

2.2.2.1 Design using Interactive Graphics

Given that the blocker design knowledge available in the literature is mostly qualitative, attempts to computerize blocker design have been primarily confined to interactive design using graphics. Special graphics programs have been developed to facilitate interactive design of blocker cross sections [Badawy 82]. However, typical CAD/CAM systems can also be used for this purpose. With the help of this CAD software, and based on experience, the designer specifies how to modify the finisher cross section to obtain an acceptable blocker cross section. The main advantages offered by these interactive design programs are [Altan 83]:

- Cross sectional areas and volumes can be calculated rapidly and accurately.
- The designer can easily modify geometric parameters such as fillet and corner radii, web thickness, rib height and width, etc., and can immediately review the alternative design on the screen of the computer graphics terminal.
• The designer can zoom-in to investigate a given portion of the forging and can perform sectional area calculations for a given portion of the forging, where the metal flow is expected to be localized, i.e., where the metal would not flow into neighboring regions.

• If necessary, the designer may review the blocker positions in the finisher dies at various opening positions to study the initial die–blocker contact point during finish forging.

These geometric manipulations are basically no different from what is done manually, but with the aid of computers they can be done faster and more accurately. The computer merely provides an electronic sketch-pad, enabling various alternative designs to be quickly visualized, and may be help with activities such as volume calculations and drafting. Thus, it is still the designer who specifies the blocker cross section, based on his own expertise.

2.2.2 Automatic Design

In addition to interactive design using computer graphics, various attempts have also been made to automate the design of blocker cross sections using computers. For the purpose of computerization, these implementations have adopted one or the other approach given in the literature, to the exclusion of all other heuristic or experience based design knowledge.

For example, in the DIE FORGE system of computer programs [Subramanian 77], the forging cross section is divided into various L shapes as shown in Figure 6, where each L shape is further described in terms of geometric parameters shown
in Figure 7. The blocker L shapes are obtained by modifying the geometric parameters of each finisher L shape by a set of fixed multiplication factors. These blocker L shapes are then assembled to form the blocker cross section.

As pointed out by some researchers [Yu 85], this procedure has the obvious limitation that it may not be always possible to divide a forging cross section into L shapes. More importantly, the set of multiplication factors used in one design may not be suitable for another design. To obviate this problem, the user was given the option of changing these multiplication factors, while the program supplies a set of default values. For example, the user could specify the factor by which blocker fillet radius is to be increased compared to the finisher fillet radius, and this factor is then applied to all the fillet radii of a given cross section. This requires significant intervention on the part of the user. Given the wide variations possible in the forging process and the part geometry, the user needs to specify these factors most of the time. Also, even within a given cross section, same multiplication factor may not be applicable for all L shapes in the section, i.e., fillet radii in different L shapes of the section may need to be modified differently. In a later version of the program, the user was even allowed to modify the multiplication factors for each L shape within a given section. While this option makes the application program more flexible, it requires even more human interaction (though not graphically).
Figure 6: Separation of a Rib-Web Cross Section into Modular L Shapes. [Subramanian 77]

Figure 7: Preform Design for an L Shape. [Subramanian 77]
In a proposal to National Science Foundation (NSF) for compiling blocker design knowledge [Altan 84], the above concept was further extended using weighting factors. Corresponding to each geometric parameter and forging factor combination, there is a design weighting factor (DWF) which specifies how the geometric parameter should be modified for that forging factor. Multiplying relevant DWFs for a geometric parameter, total DWF for that geometric parameter is obtained, which is like the multiplication factor used in DIE FORGE. For example, rib width may have two different DWFs based on rib height and forging material. Multiplying these DWFs gives total DWF, which is the multiplication factor for obtaining blocker rib width from finisher rib width. Obtaining DWFs detailed as above is at best an extremely difficult task. In fact, interaction among the process variables is so pervasive that obtaining DWFs as if these variables are totally decoupled appears to be unrealistic.

Other researchers have also applied fixed procedures for generating blocker designs automatically. For example, to obtain blocker fillet radii, Yu and Dean [Yu 85] uniformly applied the expression suggested by Bruchanov and Rebelski [Bruchanov 55], while Biswas and Knight [Biswas 76] used the exponential curves suggested by Chamouard [Chamouard 64].

Thus, all these computer applications perform blocker design by using some fixed factors or unique expressions, irrespective of variations in the geometry and/or process conditions. Consequently, the results are inadequate except in a few
specific cases, severely restricting the domain of applicability of these implementations. Of course, the reason for such inflexible design procedures is the lack of clear cut methods for blocker design, forcing the researchers to adopt relatively fixed procedures. In order to reduce inflexibility, these automated design programs typically provide interactive graphic design as an alternative.
CHAPTER 3

PROPOSED KNOWLEDGE BASE ARCHITECTURE

The architecture of any knowledge-based system (for that matter, any system of computer programs) depends to a large extent on the nature of problem solving involved in the system being developed. Accordingly, this chapter starts with an attempt to characterize the problem solving involved in automating geometric design, specifically in the context of the blocker design task, followed by a discussion on the knowledge base architecture suggested for geometric design.

3.1 PROBLEM SOLVING

Problem solving can be defined as finding a way to get from some initial situation to a desired goal. In the context of diagnosis, which has been traditionally the most dominant paradigm in AI research, problem solving often meant searching for a system state (from a space of such states) which accounts for the observed diagnostic data. This is often accomplished by the generate-and-test method of problem-solving, with systems differing in the degree to which they make use of the data in generating hypotheses. While some systems generate
hypotheses independent of the data to be explained, others generate partial hypotheses which are then modified / extended on the basis of the data. For example, hypothesis generation in MYCIN [Shortliffe 76] is accomplished by backward chaining of the rules.

3.1.1 Problem Solving in Geometric Design

In general, problem solving in design differs significantly from that of diagnosis. At the outset, the goal state is not known in most design problems. The state space is typically not discrete, and it is unbounded. In addition, in many design problems, verification of the adequacy of a suggested solution is not feasible through simple computational means; it is at best judgmental, and is often possible only through extensive trial-and-error procedures. This implies, the generate-and-test method of problem solving is clearly not appropriate for most design problems.

The above reasoning for general design problems also holds good for the geometric design problems. In addition, in the context of geometric design problems, the state space can be specified by the coordinate space $\mathbb{R} \times \mathbb{R} \times \mathbb{R}$, where $\mathbb{R}$ stands for the set of real numbers. Of course, from practical stand point, designers rarely think of geometry design as one of modifying the coordinate data. Instead, design is done in terms of geometric parameters (such as widths, thicknesses, and radii) that are of interest for the particular problem under consideration. This concept is dealt with in greater detail in Chapter 4 on geometry representation, and also illustrated later in the context of blocker design.
In brief, the geometric information content of coordinate data is re-expressed as a set of *design parameters*. Thus, geometric design is accomplished by modifying these design parameters, based on factors that affect design decisions in that particular problem domain.

However, irrespective of whether geometric design is considered as modification of coordinate data or design parameters as discussed above, the solution space is still the infinite coordinate space. Of course, normally only a small subset of states from this state space will be geometrically valid for the problem under consideration, but even this subset can be often expected to be quite large. Hence, the generate-and-test paradigm is rarely relevant for the geometric design problems.

Instead, design seems to be accomplished by the application of appropriate design heuristics to navigate through the problem space, and to converge to some unique goal / hypothesis (in this case, a single design) with very little search or backtracking. In fact, R1 [McDermott 82] accomplishes the configuration task as above, by using the match method. Since solution to the design problem is obtained here by directly applying the design knowledge without any searching, this problem solving behavior can be called *design-synthesis*.

In addition, in some design problems, it may be possible to analytically test the design, and if necessary attempt redesign. This process can be iteratively
repeated till the design is acceptable. Depending on the particular design problem, this analytical validation may require using a simple analysis program, or a reasonably involved and time consuming FEM based program. In the case of simple analysis programs, it is possible to include analysis as part of the design system, if the heuristics for redesign are also available. This design-evaluate-redesign approach was in fact used [Dixon 84] for standard V-belt design.

In certain other design problems, while formal quantitative evaluation of designs is not feasible, the domain knowledge may enable different parts of the design to be successively refined. This design-refinement concept was used in the AIRCYL system [Brown 86], where design was accomplished in four different phases: requirements, rough design, design, and redesign.

Thus, geometric design can be typically achieved by synthesizing the design directly without any search, and occasionally using additional problem solving methodologies such as design-evaluate-redesign and design-refinement.

3.1.2 Problem Solving in Blocker Design

In the case of blocker design, problem solving can be viewed as modification of the corner points of the forging cross section, so as to obtain an acceptable blocker cross section. For example, blocker design for the example forging cross section shown in Figure 8 can be obtained by appropriately changing the $X$, $Z$, and
R values of the forging cross section corner points - where X and \( Z^3 \) refer to Cartesian coordinates of the corner points, and \( R \) refers to associated radii. However, as suggested earlier for geometric design problems, design is performed in terms of design parameters, and not coordinate data. Accordingly, blocker design for a rib–web type cross section can be expressed in terms of modification of appropriate design parameters of the forging cross section, such as those listed below, and also illustrated in Figure 8:

- Rib height and width,
- Web thickness,
- Draft angle,
- Corner radius,
- Fillet radius, etc.

By changing the procedures followed for modifying the forging cross section design parameters, the geometry of the blocker cross section can be changed, thereby producing different blocker designs. In order to obtain practical blocker designs, these procedures for modifying the design parameters should consider the effect of various forging factors on blocker design. Hence, the design parameter modification procedures are best expressed in terms of design heuristics developed

3. As per the convention in the forging technology literature, X and Y axes refer to plan area, while X and Z axes refer to vertical cross sections as shown in Figure 8.
Figure 8: Features and Design Parameters for a Rib-Web Type Cross Section.

over years of experience. Some of the important forging factors that need to be considered by these design heuristics are:

- Finished part geometry (geometric complexity),
- Material characteristics (forgeability and flow stress),
• Forging equipment (type, speed and load capacity),
• Forging tolerances (closeness to final machined shape),
• Production lot size, and
• Forging temperatures (die as well as forging material).

The blocker design problem discussed as above forms an integral part of the forging die design sequence, shown schematically in Figure 9. This CAD/CAM procedure involves [Oh 86]:

1. Transfer of machined part geometry from a customer or user, to the forging supplier via IGES (Initial Graphics Exchange Specifications).

2. Conversion of machine part geometry into a forging design geometry using guidelines associated with forging design and process limitations. This could also be performed by the forging user and transferred via IGES to the forge shop.

3. Finisher die design, including determination of flash dimensions, forging stresses, and forging loads.

4. Design of blocker and pre-blocker dies, including calculation of stock volume with flash allowance and estimation of blocker and pre-blocker die geometry.

5. Verification and modification of preliminary design using advanced FEM modeling techniques. These include load estimation, metal flow simulation, and local material property evaluation.

6. Computer numerical controlled (CNC) machining of the dies.

Thus, the blocker cross sections, designed using the design heuristics, are only initial guess designs, to be evaluated further using the Finite Element Analysis
Figure 9: CAD/CAM Procedure for Forging Die Design. [Ficke 84]

(FEA). However, at least for now, integrating FEA into the automatic blocker design loop is not a practical idea, for two reasons:

1. FEA of blocker cross sections is quite time consuming.
2. Heuristics for redesign of blocker cross sections will be far more difficult to obtain than those for initial-guess design.
Thus, the design-evaluate-redesign paradigm can not be used for developing an automated blocker design system. However, human designers can interactively perform the evaluation and redesign activities, while the computer system automatically generates an initial-guess design. As a result of the analysis, if the computer designs are not found adequate, they can be modified by the designer using interactive computer graphics. This iterative design procedure will be complete when the FEM based metal flow simulations show that the designs are acceptable, and this procedure can be expected to reduce the need for extensive die-tryouts. In addition, it is hoped that the design heuristics used by the program for making initial-guess designs can be improved over a period of time, thus reducing the need for extensive design iterations even on the computer.

While evaluate-and-redesign can not be incorporated into the blocker design system, it however seems feasible to perform local design refinement (at ribs, webs and design parameters level), and the feasibility for this will be discussed in further detail in later chapters. The available blocker design knowledge seems to indicate the need for only two phases of design-refinement, viz., design and redesign; there does not appear to be any need for a rough design phase. In conclusion, the blocker design task can be accomplished by directly synthesizing the design without any search, and using the design-refinement model with design and redesign phases.
3.2 KNOWLEDGE BASE ARCHITECTURE

Given the above problem solving approach, the important issue then is how to organize and represent the various kinds of knowledge required for geometric design, specifically in the context of blocker design, so that the knowledge can be best used by the design system. This is discussed below in detail.

3.2.1 Integration of Knowledge and its Use

The majority of the knowledge-based systems (especially, the early systems) follow the popular architecture of an uniform knowledge base and a separate inference engine. Typically, the knowledge base in these systems is in the form of rules (R1, MYCIN), frames (PIP), or networks (CASNET, PROSPECTOR), and the inference engine is domain independent, i.e., it is knowledge-free.

Apart from the fact that the same knowledge organization and control structure are forced on all aspects of the problem, this uniform architecture often complicates the control issue. For example, considering production systems, such large unstructured collection of rules can easily lead to unfocused system behavior, since each rule has the same potential for use at any given instance, causing several rules to become candidates for firing whether or not they are related. Given the lack of domain knowledge in the inference engine, production systems achieve focus by resorting to some kind of syntactic conflict resolution mechanism. Thus, control
issues such as conflict resolution are often artifacts of the representation, and not necessarily inherent in the problem solving process. Consequently, the distinction between control at the problem solving level and control at the representation level is often blurred, occasionally with undue emphasis on the control constructs at the representation level.

Even under the uniform architecture, the knowledge base is typically partitioned so as to simplify control and achieve better focus. For instance, most production systems create contexts, which partition the rule-base into rule-sets. R1, for example, divides the task into 6 major subtasks. In all, it recognizes 84 different contexts, and each context on an average has about 8 rules associated with it [McDermott 82]. Metarules [Davis 76] have also been proposed to organize production rules according to some metaknowledge criterion. Studies of cognitive performance also suggest that most skilled performers organize their knowledge into successively more abstract concepts, i.e., they chunk their knowledge into a hierarchy of concepts, for effective problem solving. Thus, even under the uniform knowledge base and inference engine architecture, focus is typically achieved with some such implicit attempt at organization of knowledge.
The emerging hybrid AI development environments quite effectively address these shortcomings of the uniform architecture. They are based on the premise that uniformity of representation and programming methodology can be sacrificed to improve naturalness of expression, efficiency, and flexibility. This is achieved by (1) permitting the problem space to be partitioned, and (2) allowing each partition to be handled with the representation and reasoning technique most appropriate for it. Of course, the issue of how to partition the problem space is still left to the application system developer.

In this context, Chandrasekaran [Chandrasekaran 83a] suggests explicit organization of knowledge in terms of conceptual structures. Under this framework, domain knowledge may be decomposed into substructures, where each substructure has its own distinctive type of problem solving. A given substructure can be further decomposed into a hierarchy of specialists, which employ the same type of problem solving but differ in their conceptual content. Knowledge and its usage is thus integrated, in contrast to the uniform knowledge base and inference engine architecture. Consequently, each component of the knowledge base knows how to use its knowledge, leading to a more focused control and problem solving behavior.

It is also suggested that there is a taxonomy of problem-solving regimes, and there

4. The reference here is to the development environments such as KEE, Knowledge Craft, ART, and LOOPS, which permit combining multiple programming paradigms. This is in contrast to the tools such as EMYCIN, TIMM, and M.1, which are based on a single programming paradigm.
are attempts [Chandrasekaran 86] to develop various task-specific languages which embody these problem-solving regimes for a set of generic tasks. The argument here is that any complex problem can be decomposed into a set of generic tasks, each of which possesses a homogeneous knowledge structure and control regime. The problem-solving regimes for these generic tasks then become the building blocks in developing systems for more complex tasks.

In conclusion, uniform architecture and control is often unnatural, even though that may be all that is possible or necessary for certain problems. For most complex problems however, a knowledge organization which integrates knowledge and its use promises to improve not only naturalness of expression, but also provide better focus during the problem solving process and higher efficiency. Accordingly, the knowledge base architecture proposed here is based on this approach for integrating knowledge and its use.

3.2.2 Knowledge Organization

The following discussion regarding design knowledge and its organization is based on the blocker design task. However, as this design system is developed to a sufficient level of maturity, the resulting knowledge organization and control could be utilized as a shell for similar geometric design problems.

At the outset, in line with the above thinking of integrating knowledge and its use, the knowledge organization proposed for blocker design is based on
distribution of knowledge and control, and not on adopting a uniform knowledge base and inference engine. Accordingly, each knowledge-based module of this system is to be developed using the KBS paradigm most appropriate for it. To this end, analysis of the blocker design task indicates that apart from the volume calculations and the graphics display capabilities, which are easily derived from existing implementations in FORTRAN, problem solving in blocker design requires: (1) topology/geometric knowledge, and (2) design knowledge. Since information regarding part geometry needs to be continuously accessed by the design system, the organization and representation chosen for geometry should be conducive for quick and easy access to this information. Due to this importance of geometry representation for efficient implementation of the blocker design system, it is discussed separately in Chapter 4. Hence, further discussion below is concerned only with organization of the design knowledge.

As a first step towards developing this organization, it is useful to characterize the varieties of design knowledge required in the context of blocker design. As suggested earlier, design of a blocker cross section can be considered as modification of the design parameters based on various forging factors, using the design heuristics acquired over years of experience. However, specifying how these individual design parameters are modified is not enough; it is also important to specify how the design task under consideration progresses from one design activity to another, completing the entire design. This is particularly evident when we consider that modification of certain design parameters, e.g., web thickness, is
dependent to a large extent on the procedure followed for balancing the volumes, and not merely on the design heuristics. This implies that the sequence in which design activities are performed is of critical importance. If the individual activities such as specification of design parameters are termed design actions, then the body of knowledge encompassing the progression of design actions under a specific design situation can be called a design plan (Examples of design plans and design actions are discussed in detail in Chapter 5 on implementation). In a sense, design plans can be considered macroscopic in the overall impact of their decisions, while design actions are microscopic.

The issue then is how to organize this knowledge about design plans and design actions in such a way that focus (or control) is easily maintained throughout the problem solving process. This control problem is particularly important as the size of the knowledge base grows. Lack of proper knowledge organization can lead to extended search for the applicable piece of design knowledge, leading to degradation of system performance. Thus, effective organization of knowledge is imperative for efficient access to relevant design knowledge. To this end, concepts [Sowa 84] have been used to represent knowledge in terms of what-type structures [Goldstein 79, Brachman 77]. In contrast, Gomez and Chandrasekaran [Gomez 81] suggest viewing concepts as labels which organize the

5. The control problem here refers to deciding which particular piece of design knowledge is relevant for a given design situation.
how-type knowledge, so that they can be considered as clusters of production rules. These concepts themselves can form a hierarchical structure, thereby distributing the knowledge among various concepts in the hierarchy.

In the context of blocker design, the underlying conceptual structure can be identified in terms of features and design parameters, as shown in Figure 10. Each node in this hierarchy represents a design concept, containing a set of procedures which codify the different kinds of design knowledge (including knowledge about design plans, design actions, as well as how to use them) required to perform design at that node. This set of procedures can be viewed as a design specialist which designs according to the concept represented at the corresponding node in the hierarchy. Thus, the specialist hierarchy for blocker design closely reflects the hierarchy selected for representing geometry (as discussed in Chapter 4). A design specialist, in performing some of its design actions, can call on lower level design specialists. Thus, higher levels in the specialist hierarchy represent successively more abstract concepts. The design specialists use a blackboard [Nii 86] for communication with each other.

A design specialist in the hierarchy accomplishes the requested design task by choosing a particular design plan from its competing design plans, as shown in

6. The idea of features is dealt with in greater detail in Chapter 4 on geometry representation. In brief, ribs and webs are considered to be the features of relevance in the context of blocker design.
Figure 10: Conceptual Structure of a Rib-Web Section.

Figure 11, based on preconditions of the individual plans, and constraints imposed by the request. To illustrate the point, the top level design specialist has to design the blocker cross section by coordinating among the rib and web design specialists. However, design of ribs and webs is rarely independent; it is typically based on constraints such as volume balancing imposed by the design of other ribs and webs. For example, if there is a small web next to a large rib, the web may not be able to provide adequate material to the neighboring rib, for proper filling during finish forging. Coordinating among such conflicting constraints and arriving at the final design is accomplished by the top level design specialist through some design plan, using different plans under different design situations.
In executing a selected design plan, a design specialist typically performs certain design actions directly, in addition to possibly calling on lower level design
specialists. These design actions may be based on a set of design heuristics or some fixed algorithmic procedure, as shown in Figure 11. In either case, design actions are typically atomic in nature, such as changing the value of some parameter. However, some of them may require more complex actions, in which case they may be accomplished by further subdivision into various design goals.

In a given design problem, if the underlying conceptual structure does not have competing design plans at any node in the hierarchy, then that design task can be modeled as a fixed hierarchy of goals, and the design problem can be considered as a weak class three\textsuperscript{7} type design problem. More precisely, the complexity of the design task under consideration depends on the depth (number of levels) of the conceptual hierarchy, and the degree to which different nodes in the hierarchy employ alternate design plans.

3.2.3 Representation of Knowledge

Given the knowledge organization suggested as above, it is then necessary to specify the representation(s) suitable for different kinds of knowledge. At the outset, it should be recognized that a given piece of knowledge can be represented in a variety of ways, even though these representations are likely to differ in the degree to which they facilitate implementation. For instance, it is relatively straight

---

7. The classification referred here is the one suggested for mechanical design problems [Brown 83], as discussed in Chapter 2.
forward to convert a rule based system implemented in OPS [Forgy 77] into a system based on frames with procedural attachment, using (say) a blackboard architecture [Nii 86]. MYCIN [Shortliffe 76] is an excellent example for such conversion [Szolovits 78]. Hence, given that a variety of representations are possible, the ones chosen should be such that they facilitate quick, easy, and efficient implementation.

In the context of geometric design, in particular blocker design, effective representation of knowledge is required for design inputs and design decisions. The representations suggested for these kinds of knowledge are discussed below.

3.2.3.1 Design Inputs

Design inputs include whatever information is required before the design system can proceed with its design task. This information can be divided into three types, and the representation suggested in each case is discussed below.

Geometry to be Designed: In terms of choosing an appropriate representation, this is perhaps the most important input, since geometry and topology information is accessed/modified constantly by the design system. Hence, the representation chosen should enable quick access to geometric details as well as topological aspects of the object. Given this importance of geometry representation, it is discussed separately in Chapter 4, with only the basic concepts being described below. In brief, geometry information is embedded within the topology information, and it is
represented using a hierarchy of frames with inheritance properties. In addition, various functions are provided to supply desired information about high level topological relationships. In general, if the number of these ad hoc queries increases significantly, it may be preferable to replace them with a formal language which provides a uniform query interface.

Design Constraints: Any design task typically has some constraints which restrict the space of possible solutions. In the case of blocker design, this includes information about the desired flash allowance, process variables such as forging material, forging equipment etc. The design rules obtained so far in the context of blocker design are not very discriminatory in terms of these variables. For example, while there are design rules which design differently for steel and aluminum, there are no such rules which can discriminate among (say) steels with different specifications. Similarly, while there are design rules which distinguish between mechanical presses and hydraulic presses, there are none which can discriminate among (say) mechanical presses with different specifications. Given this low level of discriminating power in the design rules compiled so far, process information is included along with the geometry of the part itself.

However, this is not to say that design rules with such a high degree of discriminating power do not exist; it is just that they are not uncovered as yet. If blocker design knowledge can be encoded to such a high degree of detail, it is preferable to maintain separate knowledge bases for process variables such as
forging material and forging equipment. Each of these knowledge bases could be easily organized as a separate hierarchy of frames, with lower levels in the hierarchy being successively more discriminatory. If a frame hierarchy becomes fairly large, and a high degree of inferencing is required based on this frame hierarchy, then the corresponding knowledge base could be modeled as an auxiliary specialist or a consultant [Mittal 80], to provide a clearer and more uniform interface to the design system.

System Parameters: With most design systems, it is desirable to be able to alter certain aspects of the system behavior either before starting a specific design, or as a design task is progressing. In the case of blocker design for instance, common-drafts and radius-increment, discussed in detail in Chapter 5, may need to be changed for a particular design. Hence, these are best represented as global variables, for quick and direct access/modification at any level in the program.

3.2.3.2 Design Decisions

Design decisions are part of design specialists, design plans, and design actions. The representation(s) chosen should be transparent enough so that design decisions being made by these design agents can be easily identified. Frames with procedural attachment offer an attractive choice for lending this transparency. This representation also enables usage of the object programming paradigm, which in turn provides the capability to use communication as an effective means for control. While this representation scheme will be used in the complete LISP [Steel
84. Winston [81] version of BID, a combination of LISP and CRL–OPS is used in the current LISP/Knowledge Craft [Knowledge Craft 85] version, where CRL–OPS is an implementation of OPS [Forgy 77] within Knowledge Craft (KC). Briefly, design decisions are implemented in the current LISP/KC version using CRL–OPS production rules and LISP procedures. Further details of implementation are discussed in Chapter 5.

3.2.4 Deep Knowledge

Most knowledge–based systems are built based on compiled heuristics of the domain expert, which can be said to represent surface knowledge. This knowledge is shallow in the sense that it does not possess any deep understanding of the domain. In contrast, deep knowledge is based on first principles and general theories that an expert uses when faced with hard problems. Systems based on these deep models can provide causal explanations for conclusions, rather than simple inference chains and confidence scores. Hence, if possible, it is useful to provide KBS with appropriate domain specific causal models. Of course, the critical task here is to recognize the causal relationships that exist in the domain.

In the context of blocker design, it was pointed out that design can be considered as specification of design parameters such as rib width, draft, and fillet radii, based on applicable forging factors such as forging material, and forging equipment. These forging factors are important for blocker design only because they influence metal flow behavior one way or the other, thereby
enhancing/reducing die fill. This influence on metal flow behavior is in turn manifested as specifications on various design parameters, leading to compiled design heuristics.

For example, a design heuristic might suggest that in the case of steel forgings, rib height in the blocker should be more than that in the finisher. However, this is not a direct causal relationship. The reasoning behind this heuristic could be: (i) steel forgings, typically used in the automotive industry, are made in large quantities, (ii) since lot sizes are very large, die life is of critical importance, (iii) to improve die life, backward extrusion around the fillet radii (from web to rib) should be minimized, which implies a blocker rib should contain sufficient volume to adequately fill the finisher rib all by itself, and (iv) if a blocker rib has to provide enough material to fill the finisher rib by itself, given that the blocker ribs are always narrower than the finisher ribs (so as to position the blocker ribs in the finisher die cavity), the blocker rib height has to be more than the corresponding finisher rib height.

These causal relationships behind compiled design heuristics could be modeled in terms of a hierarchy of concepts, and given this hierarchy of concepts, the same design specifications can be obtained from any intermediate concept in the hierarchy. For instance, even if minimizing backward extrusion is necessitated for some other reason, the above causal relationships could still be used to specify increase in rib height. Similarly, the die-chill concept can be used to link
die/forging temperatures to design specifications, possibly through other intermediate concepts such as contact time.

In addition, KBS based on deep models are advantageous for handling the innumerable combinations of forging factors, since each factor can itself take a wide variety of values within certain practical limits. For example, while forging equipment broadly refers to machines such as hydraulic presses, mechanical presses, screw presses, and hammers, even within a given type of forging machine, there can be important variations in terms of tonnage, deformation rate, contact time etc. Similarly, there are numerous forging materials with widely varying material characteristics, which are in turn dependent on the forging temperatures.

Specifying design parameter modification for each combination of these forging factors is not practical, particularly since these modifications are presumably different for different forging cross sections. Obviously, such a combinatorial explosion should be avoided. Causal models offer a solution in this respect, by enabling reduction of these infinite variations in forging factors into weights for a finite set of concepts such as (i) ease of metal flow, (ii) geometric complexity, (iii) time available for deformation, and (iv) forging accuracies required, which can then be used in directly specifying the design parameters.

Thus, the basic idea is that forging factors can be linked to design parameters through a hierarchy/network of intermediate concepts in the domain. This
provides a much richer representation for the domain knowledge, enabling more robust solutions even on the fringes of the problem space under consideration. Also, it is relatively easy to extend the problem space to cover newer process limits in terms of temperatures, materials etc. Such detailed development undoubtedly requires thorough understanding of the influence of various forging factors on the forging process, and capturing bulk of the designer's implicit domain knowledge in terms of explicit causal models. Obviously, this is at best a very difficult task. However, given the advantages noted above, it may be worth attempting in the long run.

3.3 COMPARISON WITH RELATED KNOWLEDGE-BASED SYSTEMS

In geometric design problems, design actions such as calculation of a rib or web volume, blending adjacent radii, etc., are relatively complex in the kinds of computations that need to be performed. This need for complex spatial reasoning and accurate geometric manipulations, instead of purely linguistic reasoning, adds to the complexity of geometric design problems. That is, the primitive actions to be carried out by the design rules of geometric design problems are often quite complex, compared to those required by the rules in other domains. The relative complexity of these design primitives renders knowledge-based approach to
mechanical design problems (in particular, geometric design problems) harder compared to some other domains.

Review of the work done so far in applying KBS approach to mechanical design (as discussed in Chapter 2) indicates that while geometry is considered by all these systems one way or the other, it is considered only in terms of specifying dimensions for one or more components, and it forms only a part of the entire design task. In contrast, for the design problem being considered here, geometry design is the central activity, requiring specification of geometric details for relatively complex shapes. This implies that the representation used for geometry and topology of the part is of crucial importance for successful implementation of this design system, leading to the generic name geometric design, and distinguishing it from previous systems based on KBS approach to mechanical design.

In spite of this major difference from previous design systems, there is one hidden similarity. For the class of geometric design problems being considered here, designing a geometry can be broken up into designing a few components of the geometry. That is, even though the geometry is a single entity, it can be viewed conceptually as an assembly of a set of individual features (such as ribs and webs). When geometry is thus viewed as an assembly of features, geometric design can be interpreted to be similar to design of an assembly of components. For this reason, the conceptual structure used here resembles that of the AIR-CYL system [Brown 86], which is based on a component hierarchy.
The early KBS used a single programming paradigm to the exclusion of all others. For example, R1 [McDermott 82] uniformly uses the rule-based representation. In contrast, the current implementation is based on hybrid development, where each module is implemented using the paradigm most appropriate for it. An attempt is also made to combine usage of different programming languages, as deemed appropriate.

The problem solving here is based on synthesizing the design directly as in R1, without any backtracking. However, it considers a design refinement strategy with design and redesign phases, in contrast to the AIR-CYL system which uses four phases: requirements, rough design, design, and redesign. While the requirements phase could be added later on for verifying inputs etc., rough design phase is not necessary here, as discussed earlier.

Problem solving in the VEXPERT system [Dixon 84], on the other hand, is based on design-evaluate-redesign strategy, due to the existence of relatively simple methods for evaluating completed designs. However, this generate-test kind of problem solving is not currently feasible in the context of blocker design, given (a) the prohibitive cost and time consuming nature of verifying a blocker design, and (b) the difficulty in identifying redesign strategies for complete designs. With the advent of powerful FEA programs [Oh 81], verification is currently feasible outside the design loop. In order to be able to eventually incorporate evaluation and redesign within the design loop, the speed and cost of computation has to go
down even further, and also various redesign strategies have to be identified at the level of complete designs.

While analysis at the complete design level can not be currently incorporated in the design loop, it is feasible (and in fact, necessary) to include computations such as volume calculations and graphics. Often, such capabilities are already implemented in traditional languages. Hence, discussions on KBS approach for engineering design [Rychener 83, Mittal 86] often emphasize the need for integration with such analytical routines. In line with this thinking, the current implementation utilizes the existing software in FORTRAN for volume calculations as well as graphics.

The VEXPERT system organizes knowledge using a blackboard architecture. While the architecture suggested for BID does include a blackboard for communication among the design specialists, it is based more on distribution of knowledge (and associated control) among various specialists. In this sense, it is more similar to the architecture used in the AIR-CYL system, and to a lesser degree to that used in the PRIDE system, even though there are some variations in the terminology. For example, design goals as used in the PRIDE system are similar to the notion of design specialists. Also, design actions as suggested here are similar to steps in the AIR-CYL system, and design methods in the PRIDE system.
In order to handle failures, the PRIDE system uses dependency-directed backtracking along with advice to the problem solver from the constraint that failed. No such backtracking mechanism is provided in the current system, since the domain knowledge available does not warrant undoing the partial designs and starting over. Any occasional failures (as with the inability to provide necessary volume in a given feature) can be used for modifying the designs created so far, rather than for complete redesign. Similarly, given the domain knowledge accumulated so far, pursuing multiple design options in parallel is not contemplated as in the case of the PRIDE system. Thus, enhancing the system with further capabilities depends primarily on the accumulation of more domain-specific knowledge.
CHAPTER 4

GEOMETRY REPRESENTATION

This chapter deals with representation of geometry in the computer, and suggests a representation suitable for automating geometric design using a knowledge-based approach, specifically in the context of the blocker design problem.

4.1 THE CAD/CAM APPROACH TO GEOMETRY REPRESENTATION

Geometry representation in a CAD/CAM system is provided by a Geometric Modeling System (GMS), chosen based on the intended applications. A GMS is a computer based system which facilitates entering, storing and modifying object representations. Some of the well known commercial CAD/CAM systems, for example, are UNIGRAPHICS-II\textsuperscript{8}, ANVIL-4000\textsuperscript{9}, and CADDS-4\textsuperscript{10}. CAD/CAM

---

10. By Computervision, Bedford, Massachusetts.
systems were first introduced as interactive two dimensional (2D) systems, which usually represented objects as simple lists of points, lines and arcs. These were later extended to three dimensional (3D) wire frame systems, which, while being useful, had serious deficiencies with respect to incompleteness of representation (ambiguity in the object represented), and representational validity (allowing nonsense objects). Accordingly, current research thrust is in moving towards solid models, which are distinguished by their unambiguous representations of solids.

Of the many families of representational schemes developed for solid models [Requicha 83], only three representational schemes, viz., Boundary Representation (B−Rep), Constructive Solid Geometry (CSG), and Simple Sweep, are popular. Most existing solid modelers are based on either B−Rep and/or CSG, while Simple Sweep is sometimes used for input. Irrespective of the representational scheme used, providing wider geometric coverage has been one of the primary research topics in solid modeling.

Thus, emphasis in geometry representation has been primarily in geometric definition, and requirements of the intended engineering applications have not been typically taken into account. Currently, applications such as graphics and mass property calculations can be handled automatically in most systems (for example, PATRAN\textsuperscript{11}, ANVIL−4000, and UNIGRAPHICS−II). Also, these representations

\begin{footnote}
11. By PDA Engineering, Santa Ana, California.
\end{footnote}
facilitate usage of analytical techniques such as Finite Element Analysis (FEA), kinematic analysis etc. Most other applications are typically supported using interactive/semi-automatic programs.

However, as the GMS are evolving, attention is turning more and more to the intended engineering applications [Pratt 84, Henderson 84a]. For example, PATRAN allows for the definition of non-geometric characteristics such as material properties and variable densities, while UNIGRAPHICS-II permits assignment of attributes (user defined non-geometric data) to entities and parts. These representations are based on the recognition that objects not only have shapes and sizes but also other properties such as texture and material composition, and that physical objects in the real world deviate from the graphical models. Thus, there are attempts at alternate representations of geometry, to support automation of various engineering applications.

4.2 GEOMETRY REPRESENTATION TO SUPPORT COMPUTER-BASED AUTOMATION

As pointed out above, in developing engineering applications for computer-based automation, mere geometry representational capability for display and analysis purposes, while essential, is not sufficient. In this context, even though solid modeling and surface modeling systems provide an excellent framework for
communication between the individual CAD/CAM applications, they fall short of some of the informational needs of automated applications [Woo 84, Pratt 84].

Some CAD/CAM systems, as pointed out earlier, do include information on surface finish, tolerances etc., so as to meet the needs of various engineering applications. However, while this technological information is a necessity for many applications, it still does not provide the right kind of information about geometry to support computer-based automation. What is required is the ability for the application program to be able to reason in terms of higher level concepts of geometry, just as the human designer does.

When a designer conceives an object to be designed, he develops a mental model or image of different components of the object, and their spatial relationship or adjacency. During design, he works on this mental model of the object, which has more information than mere geometric details. These components of the object which are spatially related are often referred to as form features [CAM-I 81]. In the context of blocker design, ribs and webs are the form features of relevance. Henderson [Henderson 84a] refers to this adjacency information as semantic knowledge of the part, and it is often informally referred to as topological information. This topological information can serve as a framework into which geometric details such as points, lines, arcs, surface descriptions etc., are embedded. In fact, this structure permits geometric details to be represented in any convenient form, without altering the topological framework.
Thus, whenever a problem requires reasoning to make decisions based on part geometry, it is typically necessary to think in terms of some higher level topological concepts of part shape, i.e., in terms of part features such as holes, slots, bosses and ribs, in addition to the low level details such as points, lines and arcs. Since these part features become important due to their functional role in the overall part shape, they could be termed *functional entities*. These functional entities or form features can be different for different applications, depending on the needs of the individual applications. Thus, the application program should not only be able to represent the geometric details for display and analysis purposes, but it should also know or be aware of the form features relevant for the intended application, so that it can think and reason in terms of these features in making planning and design decisions.

As an illustration of the concept of form features in the context of process planning, the manufacturing operation selected could be based on the form feature under consideration [Pratt 84]. For example, a generic cylindrical hole could be specified as a feature which can be manufactured by any one of several processes, including drilling, boring, or a combination of some such processes. While each form feature could be associated with a set of appropriate manufacturing techniques, selection of the actual manufacturing technique in a given context can be made by selecting from this set, keeping in view other relevant variables. Thus, Pratt argues that the underlying model should be aware of the form features of the part. This concept is also supported by the works [Kyprianou 80, Chang 82, Choi
82] and [Staley 83]. Similarly, the need for an application program to be aware of the form features of the part can be seen easily in other engineering applications such as design, NC programming, and inspection. Recognizing this need for features information, Computer Aided Manufacturing International, inc. (CAM-I) has prepared a glossary of form features [CAM-I 81]. Also, to support process planning, Henderson [Henderson 84b] developed a system called FEATURES, which identifies and extracts features as volumes to be removed.

4.2.1 Hierarchic Representation for Geometry

Given that the need to identify and think in terms of form features is recognized, the next question is how should this topological/geometric information be organized to permit quick and easy access to varying information needs. It was suggested above that applications need to explicitly recognize the spatial relationships of various part features, as well as how each feature is further divided into lower level details. This necessity for the application program to be able to interpret the spatial relationships that exist in the part at different levels of abstraction indicates that a hierarchic representation of the part would be most appropriate, where the higher levels refer to form features and the lowest level refers to geometric details. This hierarchic representation is illustrated in subsequent sections, using an example forging cross section.

Non-uniqueness of Representations: While a particular hierarchic representation based on form features may be better suited than the others for a
given application, it is unlikely to find any single representation that is best suited across all possible applications. In fact, the features of interest may themselves differ from application to application. In the context of GMS, Requicha points out [Requicha 80] that the experience accumulated to date in geometric modeling indicates that no single object representation scheme is uniformly best when many applications must be accommodated, and that redundant data often plays a pivotal role in achieving efficiency. Similarly, even in the feature based topological description of a part, no unique representation is best for all applications. However, developing some generic features of relevance to support various applications, and the nature of representations in terms of these features, could be an interesting topic for further research. Work by CAM-I [CAM-I 81], and Weiler [Weiler 85] are relevant in this context.

4.2.2 Internal Representation versus External Representation

Given that the geometric representations for different applications may need to be different, a GMS should not try to cover all these applications through a single representation. Instead, it should provide a common thread linking various applications, but be independent of any specific application. This implies, the feature-based hierarchic representation developed for a specific application is internal only to that particular application, and hence can be called \textit{internal representation}. In contrast, the geometry representation external to the application, from which this internal representation is derived, can be called \textit{external representation}. 
Derivation of Internal Representation: Since GMS provide the common thread linking all applications, the application programs should be able to derive their internal representations from those provided by the GMS. At the outset, this requires the ability to identify form features, which can be done using a feature analyzer. As mentioned previously, there are in fact attempts to develop various systems for feature extraction. However, this procedure is highly inefficient. After all, this feature information is available at the time of modeling, but it is lost in the process of representing all the low level details of the geometry. Pratt [Pratt 84] suggests that at the time of geometric model creation, pointers can be set up in the modelers' data structure to link sets of surfaces belonging to the same feature. This explicit presence of feature information in a modeler's data base enables easy derivation of feature based representation suitable for a given automated application.

Weiler [Weiler 85] also suggests this approach, and discusses data structures suitable for supporting such solid models. In this context, he characterizes solid models as being evaluated/unevaluated, which is a rough measure of the amount of work necessary to obtain information about the objects being represented. That is, in an evaluated form, the adjacency information is available explicitly rather than having to be derived every time. Thus, given an evaluated GMS, the task of deriving internal representations for different applications is highly simplified.
Some of the geometry representation concepts discussed above are now illustrated, using the example forging cross section shown in Figure 12.

Figure 12: An Example Finish Forging Cross Section.
4.3 EXTERNAL REPRESENTATION USED IN BID

From practical standpoint, a design system such as BID should be interfaced to a commercial CAD/CAM system. However, CAD/CAM systems differ widely in how they represent geometry. Hence, as a practical solution, a suitable representation was chosen as the external representation for now, so that translators could be written at a later stage to obtain this representation itself from any desired CAD/CAM system. This could be accomplished either through direct translators from CAD/CAM systems, or preferably through Initial Graphics Exchange Specification (IGES). Of course, this implies that the external representation chosen currently should address the coverage problem suitably. That is, with this external representation, it should be possible to represent any geometry which may be encountered in practice, at least within the scope of the problem domain under consideration. The external representation chosen here, while not perfect in its coverage, is adequate for representing the forging cross sections as explained below.

The external representation chosen here is polygonal representation, which basically implies that the cross section can be viewed as a polygon with associated radii at the vertices of the polygon. This representation assumes that all vertex radii are tangent to the adjacent edges. This implies that the situation shown in Figure 13A is valid, while that shown in Figure 13B is not valid - since the
tangency condition is not satisfied at point However, in the case of forgings, since there is no abrupt change in the curvature normally, this tangency assumption seems to be valid in nearly all practical cases. Thus, the cross section is represented in terms of the coordinates of the vertices, along with associated radii at these vertices. Thus, external representation for the example forging cross section shown in Figure 12 is illustrated in Figure 14.

4.4 INTERNAL REPRESENTATION USED IN BID

If the example cross section shown in Figure 12 is to be used by a design program to automatically modify the ribs and webs, it can be seen that the coordinate information, while being very natural for the purposes of graphics and area calculations, is at a very low level of detail. It would be clearly useful if the
title('Example cross-section').
description('For prototype development').
forging-type(conventional).
forging-material(steel).
forging-equipment(hydraulic-press).

point(1, 0.50, 1.60, 0.0).
point(2, 0.56, 2.68, 0.1).
point(3, 1.08, 2.62, 0.1).
point(4, 1.12, 1.76, 0.15).
point(5, 3.14, 1.76, 0.15).
point(6, 3.16, 2.32, 0.1).
point(7, 3.54, 2.36, 0.1).
point(8, 3.56, 1.60, 0.0).
point(9, 3.52, 1.00, 0.1).
point(10, 3.05, 1.00, 0.1).
point(11, 3.02, 1.44, 0.1).
point(12, 2.46, 1.44, 0.1).
point(13, 2.40, 1.10, 0.1).
point(14, 1.94, 1.10, 0.1).
point(15, 1.88, 1.44, 0.1).
point(16, 1.36, 1.44, 0.1).
point(17, 1.32, 0.64, 0.125).
point(18, 0.56, 0.64, 0.125).

Figure 14: External Representation for the Example Forging Cross Section Shown in Figure 12.

program can recognize these individual features (ribs and webs), and think in their terms during the design process. Of course, the lower level design parameters should be linked to these ribs and webs, forming a hierarchic representation for the part. With this hierarchic representation, the program will be able to think in terms of the global features of the part.
However, it should be recognized that this is by no means the only representation suitable for blocker design, even though this representation seems to meet the requirements best. For example, this hierarchic representation could be done in terms of L-shapes, as discussed in Chapter 2. However, dividing the cross section into L-shapes does not seem to reflect the thinking of the designer during blocker design. In fact, designs based on L-shapes often lead to mismatch between neighboring L-shapes of a given rib or web (which will be discussed in detail in Chapter 5, and also shown in Figure 53). Of course, this mismatch could be removed by some geometry modification procedure. However, the designer never seems to go through such procedures for matching the heights. While such clever but ad hoc solutions may meet the performance goals occasionally, they fail to capture the thinking process of the designer, and can often end up wrestling with unnatural complications as discussed above. Thus, rib-web classification appears to be by far the best choice in terms of representing the geometry, in the context of the blocker design problem.

On the basis of the above discussion, internal representations based on rib-web classification have been implemented in the computer languages PROLOG and LISP. These languages, based on symbolic manipulation, make the internal representations relatively clean and transparent compared to the languages such as FORTRAN, which are primarily suitable for numeric computation. These PROLOG and LISP representations are discussed below in detail. However, these representations should only be taken as illustrative of the principle; it is more
important to recognize the need for some internal representation to aid the design task.

4.4.1 Internal Representation in PROLOG

Internal representation in PROLOG is implemented using PROLOG structures. Figure 15 shows the representational hierarchy used in PROLOG, with the topological concepts at the top of the tree, and the geometric details at the bottom of the tree. Figure 16 lists some of these PROLOG structures in generic form, for explanation of the variables. Figure 17 illustrates classification in terms of ribs and webs, using the example forging cross section shown in Figure 12. This figure corresponds to the top layers of the hierarchy shown in Figure 15. To this rib-web classification, Figure 18 adds geometric details in terms of the design parameters, completing the internal representation. So, this figure corresponds to the entire hierarchy shown in Figure 15. The complete PROLOG internal representation for this example forging cross section is now shown in Figure 19, where the design parameter variables such as H1, W1 correspond to those shown in Figure 18. Thus, the internal representational hierarchy is relatively easily implemented in PROLOG, and it is more easily understood than something implemented in a scientific language such as FORTRAN. However, even this representation lacks clarity to some extent, and the LISP implementation based on frames is far cleaner, as discussed below.
Figure 15: Hierarchic Representation of Geometry in PROLOG.
forging-u-section (Number of ribs in upper-section,
[...list of the names of ribs and webs, from
left to right...]).

forging-l-section (Number of ribs in lower section,
[...list of the names of ribs and webs, from
left to right...]).

u-rib (Name of the rib, position*,
Number of left surfaces, Number of right surfaces,
[...list of structures describing the rib...]).

u-web (Name of the web, Number of segments in the web,
[...list of structures describing the segments in the web...]).

left (Serial number of the left face,...left face description...).

top (...description of rib top...).

center (...X, Y coordinates of rib top...).

right (Serial number of the right face,...right face description...).

segment(Serial number of the segment,...segment description...).

* This variable can take on two different values, viz., "side", or
"middle". The value "middle" implies that the central vertical
line of the forging cross-section is contained within the "top" of
the rib under consideration. This information will be used to find
out which are the inner and outer surfaces. (Outside surfaces
shrink away from the die surfaces, while the inside surfaces shrink
towards the die plugs or projections).

Note: 1. 1-rib and 1-web structures are similar to
u-rib and u-web

2. u: upper 1: lower

Figure 16: PROLOG Structures to Describe a Section Geometry Internally.
Figure 17: Classification in Terms of Ribs and Webs, in PROLOG Implementation.

Figure 18: Description in Terms of Design Parameters, in PROLOG Implementation.
forging-section  (forging-u-section, forging-l-section).

forging-u-section (2, [fu-rib1, fu-web1, fu-rib2]).

forging-l-section (3, [fl-rib1, fl-web1, fl-rib2, fl-web2, fl-rib3]).

u-rib (fu-rib1, side, 1, 1,
      [left (1, flash-radius(0.0), draft(A1), height(H1)),
       top (corner-radius(R2), slope(-A2), width(W1)),
       center (X1, Y1),
       right (1, corner-radius(R3), draft(A3), height(H2))].

u-web (fu-web1, 1,
       [segment (1, fillet-radius(R4), slope(A4), width(W2))].

u-rib (fu-rib2, side, 1, 1,
       [left (1, flash-radius(0.0), draft(A5), height(H3)),
       top (corner-radius(R6), slope(A6), width(W3)),
       center (X2, Y2),
       right (1, corner-radius(R7), draft(A7), height(0.0))].

l-rib (fl-rib1, side, 1, 1,
       [left (1, flash-radius(0.0), draft(A8), height(H4)),
       top (corner-radius(R9), slope(A9), width(W4)),
       center (X3, Y3),
       right (1, corner-radius(R10), draft(A10), height(H5))].

l-web (fl-web1, 1,
       [segment (1, fillet-radius(R11), slope(A11), width(W5))].

l-rib (fl-rib2, middle, 1, 1,
       [left (fillet-radius(R12), draft(A12), height(H6)),
       top (corner-radius(R13), slope(A13), width(W6)),
       center (X4, Y4),
       right (1, corner-radius(R14), draft(A14), height(H7))].

l-web (fl-web2, 1,
       [segment (1, fillet-radius(R15), slope(A15), width(W7))].

l-rib (fl-rib3, side, 1, 1,
       [left (fillet-radius(R16), draft(A16), height(H8)),
       top (corner-radius(R17), slope(A17), width(W8)),
       center (X5, Y5),
       right (1, corner-radius(R18), draft(A18), height(0.0))].

Note:  fu: forging-upper-section  fl: forging-lower-section
       u: upper  l: lower

Figure 19: Internal Representation in PROLOG form, for the Example Forging Cross Section Shown in Figure 12.
4.4.2 Internal Representation in LISP

In blocker design, it is not sufficient if the total volume is balanced (equal to or slightly larger than the finisher volume); it is also important to ensure that volume is distributed in the right places in the blocker. Accordingly, the LISP implementation attempts to balance the volume even locally, in addition to overall volume balancing. Thus, division of cross section into ribs and webs is done more precisely than in the PROLOG implementation, as shown in Figure 20. The fillet volume is included here as part of the web, and not as part of the rib. The reasoning behind this is that blockers exist because ribs are difficult to fill, and the webs and fillets in the blocker together provide the necessary material to fill the rib during finishing operation.

In the LISP implementation, ribs and webs are described in terms of edges containing all the necessary geometric details. The convention used for naming these features and edges is shown in Figure 21, and it can be observed that the same naming convention uniformly applies to features as well as edges. Figures 22 and 23 illustrate respectively how this naming convention is used to describe the example forging cross section already seen in Figure 12, and another forging cross section with variations to the simple rib and web shapes.

The representational hierarchy is now shown in Figure 24. This representation scheme, as well as the feature/edge naming convention chosen here, are general
Figure 20: Classification in Terms of Ribs and Webs, in LISP Implementation.

enough to handle any variations to the basic rib and web shapes (with multiple steps on ribs and webs). At each node in the hierarchy shown in Figure 24, knowledge about geometry is conveniently represented in LISP using frames. Usage of frames reduces the number of global symbols required for naming the data structures holding geometry information, since any piece of information regarding geometry can be easily accessed through local bindings. From the internal representation of the example forging cross section seen in Figure 12, some sample frames illustrating the hierarchic representation of geometry are shown in Figure 25.
It can be seen that all these frames are linked through relations such as section-of, and feature-of, as shown in Figure 24. In fact, it can also be observed that these relations have inverse relations such as has-section and has-feature. However, inheritance is permitted only for the standard relations, viz., is-a and instance. All the other relations do not permit inheritance, but
Note: To Obtain an Edge Name, Append Corresponding Feature Name in Front of the Symbol Shown, (e.g., FUR-1-LV-1)

Figure 22: Naming Features and Edges for the Example Forging Cross Section Shown in Figure 12.

Note: To Obtain an Edge Name, Append Corresponding Feature Name in Front of the Symbol Shown (e.g., FUR-1-LV-1)

Figure 23: Naming Features and Edges for a Relatively Complex Forging Cross Section.
Figure 24: Frame Hierarchy for Representing Geometry Internally.
Figure 25: Some Example Frames Illustrating the Internal Representation in LISP, for the Cross Section Shown in Figure 12.
they enable links among the frames through inverse relations, leading to the hierarchic organization of geometric knowledge. Also, this descriptive power of frames render the frame based knowledge representation as above cleaner and transparent, compared to other implementations. Thus, frames provide a natural choice for organizing geometric knowledge internally in LISP.

Regarding the details of representation, there are some points worth noting:
(a) The coordinate data is duplicated in the edges. That is, the point common to two adjacent edges is maintained in both edges. This redundancy was found to be useful for accessing information during design. Of course, this requires added effort in maintaining consistency in the representation during design, i.e., making sure that a point common to two adjacent edges has the same coordinate values.
(b) The angles are maintained in degrees (not in radians) in the internal representation, since designers normally think in terms of degrees.

A brief note is in order here regarding PROLOG versus LISP versions of geometry representation discussed above. Comparing the representations shown in Figures 19 and 25, the relative difficulty PROLOG has in expressing abstract data structures is self-evident. While LISP makes this task easy, PROLOG makes it relatively hard. What makes the geometry representation easy in LISP is the usage of frames, together with associated relations. Of course, inheritance and procedural attachment render the frame based representation even more powerful and expressive.
4.4.3 Advantages of Internal Representation

The need for internal representation was brought out in the beginning of this chapter, based on some general discussion. However, in light of the implementations discussed above, it is worth reiterating and illustrating some of the major advantages offered by such geometric representations.

(a) Transparency of Representation: This is the most obvious advantage offered by internal representation. The representational hierarchy clearly lays down the relationships among various components, and how the geometric details are embedded within the topological description.

(b) High Level Queries: This is perhaps the most important advantage of an internal representation. For example, in the kind of reasoning required during blocker design, it is often necessary to know which is the neighboring feature, so that information about it can be inferred. With coordinate representation for geometry, this would be relatively tedious and time consuming. Given that the need for such information arises quite often during design, deriving it every time from coordinate data can cause the problem solving process to be bogged down by the details - apart from the fact that this is highly inefficient. On the other hand, using the evaluated internal representation with its high level of descriptive power, functions (or macros) can be easily provided to interrogate the data bases and quickly answer such queries. In fact, writing these functions is also clean and
simple. To illustrate this point, some sample queries implemented so far are listed in Figure 26.

- `get-neighbor-feature`  (feature left-or-right)
- `get-neighbor-edge`  (edge left-or-right)
- `get-top-neighbor-edges`  (feature &optional left-or-right)
- `get-neighbor-points`  (feature left-or-right)
- `get-edge-parent`  (edge-name)
- `get-fact`  (query feature-or-edge-name)
- `get-specific-features`  (subsection-name rib-or-web)
- `get-feature-edges`  (feature-name &optional edge-type)
- `get-section-edges`  (section &optional upper-or-lower)

Figure 26: Sample High Level Topological Queries About Geometry.

In addition, a few example LISP functions which implement some of these queries are shown in Figure 27.

While these functions are so easily provided given the internal representation, even the concept of a feature does not exist in the coordinate representation. Extending this idea a little further, if the informational needs of these queries
(defun get-specific-features (ss-name feature-type)
  (get-values-if ss-name 'has-feature #'(lambda (x)
    (equalp feature-type
      (get-fact rib-or-web x))))
)

(defun get-edge-parent (edge-name)
  (case (get-fact left-right-or-top edge-name)
    (left (get-value edge-name 'left-of))
    (top (get-value edge-name 'top-of))
    (right (get-value edge-name 'right-of))))

(defun get-neighbor-features (feature l-or-r)
  (let* ((ss-name (get-value feature 'feature-of))
          (pf-list (get-values ss-name 'has-feature))
          (index (position feature pf-list)))
    (if (equalp l-or-r 'left)
      (setf index (- index 1))
      (setf index (+ index 1)))
    (if (or (< index 0) (>= index (length pf-list)))
      nil
      (elt pf-list index))))

Figure 27: Example Functions for Some High Level Topological Queries About Geometry.

increase, a specialized formal language tailored to the application domain could also be provided instead of such ad hoc functions, offering a uniform query interface over and above the already high level frame language.
(c) Linguistic Interface: The internal representation and the associated naming convention can be used as the basis for defining a formal language of communication between the human designer and the design system, as described below. For instance, for some special reason, if a design parameter has to be given a particular value overriding what is specified by the design heuristics, that message could be communicated to the design system linguistically, using a sentence such as the one shown below.

\[
\text{(bur-1 height 1.4 inch)}
\]

| edge/feature name | attribute/slot | value |

The above example message specifies that the height of the first upper rib in the blocker should be 1.4 inches. This linguistic communication capability could also be used for other purposes. For example, by removing the value information (that is, 1.4 inches) in the above example sentence, it could be interpreted as a query about the rib height. Of course, the syntax and semantics of the sentences would depend on the formal language actually defined.

A linguistic interface based on such communication capability could be of use in two different situations, as discussed below.

1. Interactive Modification: After the design system performs automatic design, either partially or completely, linguistic communication could be used to interactively query/modify one or more design parameters. This
provides an alternative to the commonly used interactive graphics design, and it may be preferable over interactive graphics in certain design situations.

In design programs based on interactive graphics, instructions to the computer are communicated by combining the usage of cursor (to point out items on the screen) and responding to prompts from the computer. This often suffices as a simple and convenient means for specifying the action to be taken by the computer. However, this is not always the most convenient way to instruct the computer program as to what to do. For example, if the user wishes to instruct the computer program to move certain amount of material from one part of the section to various other parts of the section according to some criterion, using interactive graphics will make it relatively awkward to communicate this idea. However, a formal language could easily accommodate such complexity in the instructions.

2. Prior Specification of Design Instructions: More importantly, the design parameter values which should override those specified by the design heuristics could be supplied to the design system as part of the input itself. The design system could then use these values (supplied as part of the input) rather than the values suggested by the design rules, during the automatic design stage itself. Thus, designs based on specific assumptions (say, a predefined draft on a surface) could be obtained, without having to resort to interactive modification after automatic design by the system.

Of course, defining such a language would require the ability to linguistically refer to all the items of interest. In this context, the internal representation and the associated naming convention provides a natural means for linguistically referring to different parts of a rib-web type cross section. This in turn facilitates the process of defining the syntax and semantics of a formal language, as
illustrated above. Such a language could provide a common medium of communication between the computer and the human designer, and it can also serve as a step towards the ultimate goal of voice-activated CAD. Finally, instead of being used independently, the graphic and linguistic interfaces can complement each other in their usage. That is, each interface can be used to accomplish those tasks (such as query and modification) for which it is better suited.

(d) Problem Extension: Attempting to cover more complicated geometries will indicate even greater need for the internal representation. Given the complexity of blocker design, while automating three dimensional (3D) blocker design is an extremely difficult task, one of the requirements for such an effort would be the ability to say how material moves across the sections, and from which section to which section. The high level description provided by the internal representation will make at least this aspect of the problem feasible.

4.5 GEOMETRY RECOGNITION

The discussion in this chapter is so far concerned with why and how geometry should be represented internally. In the context of BID, the question of how the internal representation is derived from the external representation has not been addressed, and this will be discussed below. In brief, the surfaces are first described symbolically, transforming the problem into a simple pattern recognition
problem, and the internal representation is then derived using an appropriate parsing algorithm. Obviously, the exact description to be used for the surfaces can vary depending on the geometries and form features that are of interest for the problem domain under consideration. However, the overall methodology discussed below in the context of blocker design should also be useful for geometry recognition problems in general.

As discussed above, the task of developing the internal representation is divided into two subtasks: (a) Surface Description, and (b) Feature Recognition. Once feature recognition is successfully accomplished, describing geometry in the internal representation form is fairly trivial. The subtasks mentioned above are discussed below in detail.

4.5.1 Surface Description

In rib–web type forgings, ribs are distinguished from webs primarily based on slopes of the corresponding edges. Small slope on the web surface is normally considered a part of the web and not the rib. Such small tapers are in fact often used in the webs to facilitate material flow to the ribs during finishing operation. However, if there is an edge with a large taper (say 70 degrees), then that edge is normally considered a part of the rib. An exception to this is in the case of webs near the flash land, as shown in Figure 28 and discussed later in further detail. Thus, discrimination of ribs and webs on the forging surface can be based on the slopes of the corresponding edges. An angle of 45 degrees seems reasonable as a
discriminant angle. Changing this angle will imply changing the definition of what are considered ribs and what are considered webs. However, small changes in the discriminant angle should not make much difference in most realistic situations. Based on this discriminant angle, the upper and lower surfaces of the forging are both described in terms of symbols, as below.

\[
\begin{array}{c}
\text{(a)} \\
\text{Small Width (w): Step Considered Part of the Rib}
\end{array}
\]

\[
\begin{array}{c}
\text{(b)} \\
\text{Large Width (W): Step Considered Part of the Web}
\end{array}
\]

Figure 28: Rib-Web Type Parts, with Multiple Steps Near the Flash Land.
First, using the coordinate data, lines are defined for each pair of adjacent points. Each line is then given a specific symbol (#, +, 0, -, or =), as shown in Figure 29, based on its angle of inclination. Note that angles in the I\textsuperscript{st} and IV\textsuperscript{th} quadrants only are considered, since directionality of the lines is not important. Given the above notation for assigning symbols to the lines, each surface can be described by a string of symbols.

\begin{center}
\begin{tabular}{c|c}
\hline
Symbol & Angle Range \\
\hline
"#" & 90° ≥ θ ≥ 45° \\
"+" & 45° > θ > 0 \\
"0" & θ = 0 \\
"-" & 0 > θ > -45° \\
"=" & -45° ≥ θ ≥ -90° \\
\hline
\end{tabular}
\end{center}

Notes: θ : Angle of Inclination of an Edge.

Figure 29: Convention for Symbolic Description of a Surface, Based on the Slopes of its Edges.

For example, the upper and lower surfaces of the example forging cross section, seen in Figure 12, can be symbolically described as shown in Figure 30. The symbolic description for a more complicated section is illustrated in Figure 31.
Using these surface descriptions (patterns), the features on the surface are then identified as below.

4.5.2 Feature Recognition

Given the above symbolic description for the surface, the problem is reduced to one of identifying the features from this description. This can be accomplished by defining a formal language over the alphabet #, +, 0, -, and =, and performing a sentence analysis over this language. The grammar defining the vocabulary/well-formed sentences of this language is shown in Figure 32, as a set of productions. The formalism or notation in which these productions are written is called the Backus–Naur Form (BNF) [Rosen 67].

What is required now is to perform sentence analysis of the surface patterns, so as to identify the ribs and webs. This is easily accomplished by a simple parsing algorithm, which uses the productions defined in Figure 32 to search for rib and web patterns in the surface pattern. The ribs and webs so identified are organized into a list of rib–web 2-tuples or ordered-pairs, as illustrated in Figure 33. From these ordered-pairs, the internal representation is now generated using the already generated information about each line.

The pattern recognition approach used here is valid for identifying any rib–web type part, except in the case of parts with multiple steps near the flash land. In this case, as shown in Figure 28, the decision of whether to divide this region
Figure 30: Symbolic Description of a Surface, Based on the Slopes of its Edges.

Figure 31: Symbolic Description of a Relatively Complex Surface, Based on the Slopes of its Edges.
into a single rib as in Figure 28a or into a web followed by a rib as in Figure 28b, is judgmental. This decision can be based on the relative magnitudes of the line lengths, i.e., the width $W$ to height ratios. This has not been implemented in the current version of BID, due to lack of information about practical values for these ratios. For this special case, the pattern recognition approach needs to be
**EXAMPLE-1**

**UPPER PATTERN**  
#-=0#+=
PAIRS  
[([#-= 0) (#+= )]]

**LOWER PATTERN**  
#0=0#0=0#0=
PAIRS  
[([#0= 0) (#0= 0) (#0= )]

**EXAMPLE-2**

**UPPER PATTERN**  
#0#-=0+=0#+=0=  
PAIRS  
[( #0) (#-=#0=+= 0) (#+= 0=)]

**LOWER PATTERN**  
#0#+=0+=0#+=0=  
PAIRS  
[( #0) (#+= -0+) (#+= )]

Figure 33: Parsing Rib-Web Ordered Pairs from the Surface Patterns, for the Surfaces Shown in Figures 30 and 31.

either augmented by a simple procedure which takes these ratios into account, or enhanced to include these ratios themselves into the pattern recognition problem.
CHAPTER 5

IMPLEMENTATION AND RESULTS

The previous two chapters laid a foundation for organizing and representing the knowledge required for geometric design (including design knowledge, and knowledge about the geometry), with special reference to blocker design. Based on the knowledge organization suggested in these chapters, the current chapter deals with the specific implementation details of the Blocker Initial-guess Design (BID) program.

5.1 OVERVIEW OF BID

5.1.1 The BID System

As brought out in the first two chapters, given the ill-structured nature of the blocker design problem, the exploratory style of AI programming, which emphasizes early prototype development and incremental refining, is very appropriate. This allows understanding of the problem as well as exploration of various solution methodologies to progress in successive stages, complementing each other at every stage. In line with this thinking, BID development also progressed
in stages, with improvements at each stage in the knowledge base as well as the solution methodologies. Major milestones of the BID development effort so far are as below.

- Development of a proof-of-concept demonstration program in PROLOG, based on information available in the literature.
- Development of a LISP/Knowledge Craft (KC) version, after consultation with die designers from a few companies, and studying some existing designs.

Further discussions on implementation are based on the LISP/KC version of BID. Before discussing the BID architecture in detail, it is useful to briefly reiterate the input and output requirements of BID. As shown in Figure 34, given a finish forging cross section and applicable process variables, BID aims at designing an appropriate blocker cross section. The discussion below is concerned with the overall architecture of BID, to help achieve this objective.

5.1.2 System Architecture

As pointed out in Chapter 4, while CAD/CAM systems typically use some form of coordinate data representation to facilitate analysis and display of geometries, the process of design however is usually done in terms of design parameters (such as fillet and corner radii, rib-height, web-thickness, and draft angle). It was observed that the coordinate data representation (external representation) should be transformed into some convenient form of parametric
representation (internal representation), before the task of design is attempted. Of course, this necessitates an inverse transformation from the internal representation to external representation, once the design is complete. Figure 35 describes these primary system functions in terms of a block diagram. As shown in this figure, design of blocker sections is achieved by executing in succession (a) conversion from external to internal representation, (b) design, and (c) conversion from internal to external representation.

However, it should be noted that Figure 35 describes only functional relationships among the major activities to be performed during blocker design; it does not indicate either relative complexity of these activities, or the modules
NOTE: • The input "forging data" refers to geometry of a forging cross section as well as process variables such as forging material, equipment, etc.
• The design module designs blocker cross sections based on appropriate design heuristics.

Figure 35: Functional Relationships in BID.

actually implementing these activities. For example, while conversion from the external representation to the internal representation requires some pattern recognition capability in order to identify the form features, conversion from the internal representation to the external representation is a fairly trivial task. This is
due to the fact that obtaining the external representation from the internal representation is just a matter of presenting the geometric details already embedded in the internal representation in some suitable form, while generating the internal representation requires extraction of the topological information from geometric details of the external representation. Also, from the perspective of blocker design, both these representation conversions are merely supportive activities to the actual design task.

Apart from the above differences in the relative complexities of the major activities indicated in Figure 35, BID also includes the ability for graphical presentation of the design, the ability for volume calculations to support design etc. All these activities are implemented in BID using various modules, and an architecture reflecting how these modules are integrated is shown in Figure 36. Implementation of each of these modules is detailed below, after briefly discussing some of the salient features of the methodology followed in implementing these modules.

Implementation Methodology: It was pointed out in Chapter 3 that the early AI systems emphasized adopting an uniform programming methodology, which leads to parsimony of the underlying representation scheme, and a relatively small development environment. However, it was also suggested that no single methodology is best suited for all types of problem solving, and that naturalness and efficiency can be maximized by selecting the representation and reasoning
Note: 1) Link is Software-Disabled, When "BID" Is in Fully Automated Mode.
2) Architecture of the Design Module Is Shown Separately.

Figure 36: Architecture of BID, Indicating Module Interaction.

technique appropriate for each of the functions to be performed, which is in fact
the premise of hybrid programming environments. Influenced by this thinking, an
tempt is made to implement each module in BID using the methodology
appropriate for that particular module, rather than selecting one uniform methodology for the entire blocker design problem. This concept is also extended to include usage of the computer language appropriate for a given module.

In addition, most AI application programs developed to-date typically exist independent of other application programs, i.e., they are not integrated with other AI/non-AI application programs. In contrast, as already shown in Figure 9, BID needs to be integrated with analysis programs as well as commercial CAD/CAM systems, which are typically implemented in FORTRAN. Also, given the extensive use of FORTRAN in developing CAD/CAM systems, a vast library of analysis/graphics utilities already exist in FORTRAN, to support various engineering applications. Hence, in the development of BID, it was decided to use FORTRAN whenever appropriate, particularly if the capabilities required for certain modules already exist in FORTRAN.

Thus, in implementing a given module in BID, an attempt is made to use the computer language as well as the solution methodology best suited for that particular module. Also, for maximum flexibility, it is useful to maintain the user interface as a separate module, so that it can be invoked from anywhere within LISP. Figure 37 summarizes the salient features of this implementation methodology, as well as briefly illustrate how this methodology is used in implementing the individual modules of BID. Implementation details of each of these modules is discussed below, based on the above methodology.
1. Use appropriate computer languages, for effective integration of symbolic / numeric computation.

- FORTRAN Volume Calculations, Graphics
- LISP Design Modules, User Interface
- DCL(VAX/VMS) File, Process interfaces

2. Integrated use of relevant KBS paradigms

- Frames for geometry representation
- Rules to encode design knowledge
- Blackboard to maintain focus during design

3. User interface as a separate module, so that it can be invoked from anywhere within LISP.

**Figure 37: BID Implementation Methodology.**

Implementation of the BID Modules: As shown in Figure 37, the Volume Calculations module for calculation of volumes for arbitrary polygons with corner radii, and the Graphics Display module are both in FORTRAN, adopted from already existing software. Hence, it was required only to provide appropriate LISP-FORTRAN interfaces. In the context of volume calculations, balancing of volumes is done in LISP using the volume calculation routines in FORTRAN, with the LISP/FORTRAN communication being provided directly through the operating
system. In the case of graphics, given the large amount of information that needs to be communicated to FORTRAN from LISP, the interface is provided through files. That is, after the display file for graphics is written, a separate process is spawned from LISP to execute the graphics program.

The User Interface module is provided as an independent set of functions, which can be activated by any LISP module. However, when BID is executed in fully automated mode, the system automatically disables activation of the user interface by the Get Internal Representation and the Design Blocker Section modules, as illustrated in Figure 36. Also, when activated, the user interface assumes that the required global objects are already available for access.

General LISP functions such as those for obtaining higher level topological information from the internal representation (as discussed in Chapter 4) are all grouped into a library of LISP Utilities, and they are accessible to all modules.

The modules Get Internal Representation, Design Blocker Section, and Get External Representation are all implemented based on LISP, and can call other modules as shown in Figure 36. Given the relative complexity and the importance of the design module compared to the other modules, the details of its architecture are discussed separately. While these three modules can be executed one after another in a fully automated mode, if desired, each of these modules can also be activated independently. Thus, the BID system accomplishes blocker design using a
set of interacting modules, where the architecture of the design module itself is as discussed below.

5.1.3 The Design Module Architecture

The architectural foundations of the design module are based on the concepts outlined in Chapter 3. Accordingly, the architecture of the design module is modeled as a hierarchy of design specialists, as shown in Figure 38, based on the conceptual structure of a rib-web section, and the design decisions are classified into design plans and design actions, based on the architecture of a design specialist. To provide a quick review, a design action refers to individual activities such as specification of a design parameter value, while a design plan refers to the body of knowledge encompassing the progression of design actions under a specific design situation. Specific examples of design plans and design actions are discussed in the next section.

In implementing the design module, the objective has again been integration of relevant KBS paradigms. Accordingly, in the LISP/KC version of BID, geometry representation is implemented in terms of frames, as discussed in detail in Chapter 4, and the design knowledge is implemented using a combination of LISP and CRL-OPS, an implementation of OPS within Knowledge Craft. While the design plans are implemented in LISP, the design actions are implemented in CRL-OPS.
Figure 38: Architecture of the BID Design Module.

To maintain focus within CRL-OPS, a blackboard is used for directing the flow of control. In fact, most rule-based systems such as R1 [McDermott 82] use some form of context mechanism to group the rules, so as to achieve desired control behavior. In BID, focus is maintained by using not just a single context, but a hierarchy of contexts. At any given instance, the blackboard maintains the currently active hierarchy of contexts (as well as other active information), so that rules can be fired based on the then active context hierarchy. There are two
generic rules to accomplish this context switching. Thus, the implementation within CRL-OPS essentially follows a blackboard architecture. The task of scheduling a knowledge source in the blackboard architecture is achieved through the two OPS rules which switch contexts at appropriate times. This implies that OPS is used only to scan the rules during each iteration, with precise control behavior being achieved using the blackboard and the context switching rules. The fact that this blackboard architecture imposed over the OPS rule scanner achieves good control behavior is easily established by noting that the conflict set on an average contains only two rules.

Some of the important features of the BID system architecture can be now summarized as below.

1. The BID system integrates multiple KBS paradigms as well as multiple computer languages, as illustrated in Figure 37.

2. The design module architecture permits access to multiple design plans for each design specialist, and hence use of alternate design plans under different design situations is relatively simple.

3. The architecture makes it comparatively easy to modify the knowledge base, i.e., to add or modify design plans and design actions. This is not to say that additions or deletions can be done by novices to the system, but it should be comparatively easy at least for those familiar with the system. Of course, referring to the distinction made in the kinds of design knowledge, it is easier to modify design actions compared to design plans. Changing design plans is more complex primarily because they do not perform one simple action. Instead, they typically involve a variety of design actions, causing many changes to the state of the design, as illustrated by the example design plans in the next section.
4. The design heuristics used by designers from different companies differ from each other. That is, each design group has its own design culture. Accommodating such variations is made easier by this architecture, since it is comparatively easy to change design plans and design actions, as explained above. Given this flexibility in modifying the knowledge base, each company could modify it to reflect its own design practices. Thus, the BID system can serve as a shell for blocker design.

5. The BID system implementation permits it to be run either in a fully automated mode, or in a semi-automated mode, allowing interruption after each feature design (for possible interactive modification or verification). However, it is expected to be used more in a semi-automated mode, (say) as a design assistant, where the designer watches what BID does automatically and intervenes only when it is necessary to make some modifications. The system's usefulness lies in the fact that it will tirelessly use all the design knowledge incorporated into it, without forgetting the details.

In conclusion, this section described the architecture of the BID system, illustrating how the implementation methodology effectively integrates numeric and symbolic computation, as well as multiple KBS paradigms, for achieving a flexible and extensible implementation.
5.2 DESIGN KNOWLEDGE

Having presented an overview of the BID system, the discussion now concentrates on the design knowledge required for implementing BID, which can be divided into knowledge about design plans and design actions. Since this knowledge is not readily available, it is important to make a systematic effort for acquiring the knowledge. Accordingly, this section starts with an outline of the effort made for acquisition of design knowledge, and is then followed by a discussion on design plans and design actions relevant for blocker design.

5.2.1 Knowledge Acquisition

In any knowledge-based system, acquisition of domain knowledge is of critical importance. In the context of blocker design, this assumes an even greater importance due to the difficulty involved in obtaining the design knowledge. Blocker design knowledge can be obtained from two sources of expertise: (a) literature, and (b) practicing die designers. As noted in the discussion on the state of the art, while there is a fair amount of information in the technical literature on design of forgings, information on blocker design, particularly quantitative information, is virtually non-existent. Also, blocker design knowledge is typically considered by the forging companies to be proprietary information, contributing to their competitive edge in the market place.
Thus, gathering quantitative information on blocker design is a very difficult task. In fact, this is the main reason why most computer implementations for blocker design have taken a restricted approach to the problem. Hence, it was decided at the outset that information gathering should rely heavily on practicing die designers, and literature would only provide general background as a starting point. Accordingly, the following steps indicate the plan under which knowledge acquisition progressed.

1. As a first step, existing literature on blocker design was reviewed. In addition, to initiate the information gathering process from the forging industry, a questionnaire was prepared on the blocker design task, and was mailed to various forging companies along with a brief summary of the project objectives and the approach being taken. Also, engineering drawings for two typical rib-web type forging cross sections were prepared and mailed, requesting them to outline their approach for blocker design for these sections.

2. As expected, given the time needed by the designers to reply to these questionnaire/drawings, the response was slow and not encouraging. To speed up the process, it was decided to obtain some existing design drawings, study them, and then arrange discussions with the designers.

3. By following this strategy, technical assistance was being obtained from the forging companies listed below, typically in terms of providing design drawings, and in some cases with technical discussions on blocker design.

- Detroit Forge, Detroit, MI - visited
- Eaton, Marion - visited
- Ford, Canton, OH - visited
- National Machinery, Tiffin, OH - visited
- Aluminum Precision Products, Santa Ana, CA
- Cameron Iron Works, Inc., Houston, TX
In addition, discussions were held with an experienced die designer, hired as a consultant.

Some of the blocker design knowledge compiled is included in an interim annual report [Vemuri 86] on blocker design, submitted to the National Science Foundation (NSF). Further discussion on design plans and design actions are based on this report.

5.2.2 Design Plans

At the outset, it should be recognized that a design specialist may have a variety of design plans, since no single design plan is likely to meet the design needs of all possible blocker design situations encountered by that specialist. For example, at the blocker section design specialist level, while design of ribs followed by design of webs may be suitable in general, in certain cases it may be preferable to reverse the sequence. Also, the fact that designs produced by different designers often differ from each other indicates possible variations in the design plans used. It is possible that these differences are only in the details, but it can
be expected that at least in some cases the basic philosophy of design (or the set of design plans used) itself is different. This clearly indicates the existence of a rich variety of design plans. Based on discussions with forging die designers, a set of design plans have been identified. Before describing some of these design plans in detail, major considerations that went into synthesizing this set of design plans are outlined below.

5.2.2.1 Design Considerations

1. The cross section of a rib–web type part can be thought to consist of a number of ribs and webs. Modifying these ribs and webs essentially leads to various blocker designs. Ribs, as defined here, include draft angles and corner radii, and webs include associated fillet radii. Fillets are considered part of the webs, since fillets (together with the web portion) are designed primarily with the objective of providing adequate material to fill the ribs, and exact values of blocker fillet radii are otherwise irrelevant to the ribs.

2. During the forging process, material has to be moved up into the ribs. Thus, filling of ribs is more difficult, and hence their design in blockers is more critical. On the other hand, design of webs in blockers is primarily done to provide the necessary total volume, as well as volume to fill the neighboring ribs. (Web design is occasionally influenced, although to a lesser degree, by the reduction in height required for obtaining desired material properties). Hence, during blocker design, design of ribs should precede design of webs.

3. As for individual rib design, rib widths in blocker have to be smaller than the corresponding widths in the finisher. Hence, based on rib height, blocker rib design can be divided into the following two categories:
a. Blocker rib height > Finisher rib height

b. Blocker rib height < Finisher rib height

4. Volume balancing should be done not only on the entire cross section, but also more locally - at ribs and webs level. To enable discussion of volumes at ribs and webs level, a standard notation is developed for specifying the variables referring to these volumes. Figure 39 illustrates this variable notation.

5.2.2.2 Blocker Design Plans

Given the above design considerations, a possible design plan for the blocker design specialist can be briefly stated as:

1. Section width design,
2. Rib design (including localized volume distribution),
3. Web design (including localized volume balancing), and
4. Total volume balancing

This design plan, used in the current implementation, does not take into account the possible need for redesign. For example, while designing a particular web, localized volume balancing may sometimes require an unrealistic increase in the web thickness (see example designs). This problem is particularly acute if the finisher web thickness is already close to the height of a neighboring rib, or the web (width) is too small compared to the neighboring ribs. In such situations, one of the lower level design specialists may fail in its design task. Then, the blocker
Figure 39: Variable Notation for Volumes.
design specialist should explore alternative design plans, such as distributing the volume differently to the neighboring features, or requesting the individual feature design specialists to relax specification of one or more of the design parameters. These redesign strategies can themselves depend on the prevailing forging factors and the part geometry. In addition, under certain forging factors/part geometries, it may be preferable to design webs even before designing the ribs.

Total Volume Balancing: As part of the above design plan, volume is balanced at the individual rib and web level. While it is essential to do this localized volume balancing, it is also necessary to ensure that total volume of the blocker is greater than the finisher volume, with appropriate flash allowance. If there is either too much/too little flash allowance, the blocker design needs to be modified to achieve the desired flash allowance. So, the actions of this task can be summarized in the following steps:

1. calculate total volume of the blocker, and hence the actual flash allowance, i.e., the extra percentage over finisher volume.

2. If the actual flash allowance obtained as above is not within acceptable limits, which may happen due to various approximations during local volume balancing, modify the blocker design to bring the flash allowance within acceptable limits.
5.2.2.3 Rib Design Plans

Design of a rib essentially involves modification of the following design parameters:

1. Rib height,
2. Rib width,
3. Draft angles, and
4. Corner radii.

Of these, modification of the last three design parameters is primarily done based on empirical/heuristic design guidelines. However, modification of the rib height is based primarily on volume considerations, except that the decision whether the height is to be increased or not seems to be heuristic. Hence, design of a rib could be divided into two types, viz., decreased rib height, and increased rib height. Each type of design is further discussed below, after describing a strategy for volume distribution.

Volume Distribution: As pointed out earlier, while designing a blocker, it is not enough to balance the total volume; it should also be ensured that material is located at right places in the blocker, so as to fill the finisher die (especially the ribs) adequately. This implies that more localized volume balancing could be highly advantageous. Given that a rib–web type cross section can be viewed in terms of a collection of ribs and webs, and that ribs are more difficult to fill, this localized
volume balancing requirement can be interpreted to primarily mean: how to allocate the finisher rib volume in the blocker, so as to fill the ribs adequately during finish forging.

Any extra volume of material to fill a given rib during finish forging can come from either its neighboring webs, and/or a bulge opposite to the rib under consideration. If the rib is an end rib (near the flash land), any extra material required from the web region has to come from its only one neighboring web. However, if the rib is a central rib (with webs on both sides of the rib), then the question arises as to how much material should come from which web. An intuitive answer, in the absence of any better scientific/heuristic basis, seems to be that the extra material required from the webs should come from portions of the two neighboring webs in the ratio of their relative volumes. (Strictly speaking, some material may come from web regions on the other side of the parting line. However, this can be considered later on, if sufficient data is obtained.) The actions to be taken to distribute volume for each of the ribs can be now summarized as below, given the notation for volumes shown in the Figure 39.

1. First, find the volume required in the blocker to fill the finisher rib. This is calculated by adding appropriate percentage to the finisher rib volume, to account for flash.

12. When a thin, tall rib is to be forged on one side of a web, a defect (called suck-in defect) can develop at the bottom of the rib. To avoid this defect, a bulge of metal is often added at the bottom of the blocker rib, as shown in Figure 12.
\[ V_{sw} = V_{fw} \times (1 + \frac{\text{Flash Allowance}}{100}) \]

2. This volume should be obtained from the blocker's rib, bulge (if any), and neighboring webs. Hence, decide if any bulge is required, and if so, design the bulge.

3. By subtracting this bulge volume, calculate the volume \( V_{ar} - V_{bb} \) to be obtained from the blocker rib and its neighboring webs.

4. In the blocker, this net volume \( V_{ar} - V_{bb} \) may be placed entirely in the rib, or some part of it may be distributed to the neighboring webs. If some part of this volume has to be in the webs (actual value to be calculated only after rib design), and if there are two neighboring webs, calculate the ratio in which this volume is to be assigned to the two neighboring webs. One suggestion for this is to assign based on relative volumes of the neighboring webs, as below:

\[ R_1 = \frac{V_{fw1}}{V_{fw1} + V_{fwr}} ; \quad R_r = \frac{V_{fwr}}{V_{fw1} + V_{fwr}} \]

It should be noted here that actual values for the volumes to be assigned to the neighboring webs are not calculated here; only the percentages are calculated. Actual volumes are calculated only after the rib is designed, when the exact volume required from neighboring webs is known.

Decreased Rib Height: When the height of the rib is reduced in the blocker, obviously extra material is required from either the neighboring webs (including fillets) and/or a bulge opposite to the rib, to fill the finisher rib adequately during...
finish forging. The procedure for volume distribution already specifies the volume of the bulge, if it is to be used, and also the ratio at which volume is to be assigned to the neighboring webs. Hence, once rib design is complete, this task can calculate the amount of material to be provided by each of the neighboring webs. Actions to be performed by this design plan, when the rib height is decreased, can now be summarized in the following steps:

1. Perform rib design, i.e., specify rib width, rib height, drafts, and corner radii.

2. Calculate total volume to be provided by the neighboring webs
   \[V_{ar} - V_{bb} - V_{br}\].

3. This volume is now allocated to the neighboring webs, in the ratio of their volumes as determined by the procedure for volume distribution.

   \[
   \begin{align*}
   \text{Volume from left web} &= (V_{ar} - V_{bb} - V_{br}) \times R_l \\
   \text{Volume from right web} &= (V_{ar} - V_{bb} - V_{br}) \times R_r
   \end{align*}
   \]

Increased Rib Height: The decision to increase the blocker rib height is primarily made so that filling of the finisher rib can be achieved not by backward extruding the material from the web, but by upsetting a taller blocker rib. Consequently, the amount of rib height increase is based on rib volume balancing, within certain practical limits of rib height to width ratio. This implies that in the case of increased rib height, normally there is no volume distribution to the neighboring webs, i.e., the webs in the blocker are not expected to provide extra material to fill the ribs during finish forging. For some reason, if the rib height
can not be increased sufficiently to balance the rib volume (it is not yet clear what this limit is), then additional material required for the rib can be obtained from the neighboring webs, as in the case of decreased rib height.

The rib design plan discussed as above does not specify the sequence in which the rib design parameters such as rib width, draft angles, corner radii, and rib height, are to be established. It is possible that different sequences are necessary under different conditions, necessitating a variety of design plans. Also, as pointed out earlier, a rib may need to be redesigned, if a neighboring web fails in allocating the necessary volume. This redesign of the rib could itself be accomplished in many ways, including changing rib height or corner radii. Thus, a variety of rib design plans are feasible. The current LISP/KC implementation is based on the design plan discussed above.

5.2.2.4 Web Design Plans

Design of a blocker web essentially involves specifying the following design parameters:

1. Web thickness/web shape, and
2. Fillet radii.

Note that blocker web width is never directly designed. It is indirectly determined based on associated rib/web/section-width specification. Design of the parameters listed above is based on heuristics as well as volume balancing considerations.
As pointed out earlier, blocker dies are used to fill the ribs adequately during finish forging. This implies that during finish forging, typically there is material movement from webs to ribs, and not vice-versa. Hence, webs in blocker always have more volume than those in the finisher. At the outset, the blocker web has to have a certain extra percentage of material over the finisher web, to provide for flash. In addition, extra material may have to be provided for in the blocker web, to fill the neighboring rib(s) during finish forging, the actual amount of extra material being as determined during rib design. Thus, knowing the total volume to be maintained in the blocker web, its design can be completed based on design heuristics. As for the shape of the web, one of the following can be selected, based on design guidelines applicable:

1. Straight web,
2. Tapered web, or
3. Circular arc web.

Of these options, circular arc web is relatively more difficult to obtain from the point of view of manufacturing, and it is not considered here. The actions of this design plan can be now summarized in the following steps, assuming a straight (flat) web.

1. Determine finisher web volume with a certain percentage for flash allowance:

\[ V_{aw} = V_{fw} \times \left(1 + \frac{\text{FlashAllowance}}{100}\right) \]
2. Add extra volume of material to be provided for neighboring ribs, to obtain total volume \( V_{bw} \) to be maintained in the blocker web.

3. Design the blocker web, i.e., its thickness/shape and associated fillet radii, maintaining the volume \( V_{bw} \).

As in the case of rib design, a variety of design plans are possible for web design. As an example, design plans are required to specify tapers on the webs.

In conclusion, there are a variety of design plans relevant for the blocker design problem, but only a small set of these design plans have been identified and implemented in the current version of BID. In this sense, current implementation of BID (not the blocker design task) can be classified as a weak Class 3 type design, based on the discussion in Chapter 3. However, the blocker design problem itself is clearly quite complex, with a rich variety of design plans.

5.2.3 Design Actions

Design actions differ from design plans in that they are concerned with design details such as specification of design parameters, rather than overall planning of design actions. As discussed before, design actions are not independent of each other; they are embedded within the corresponding design plans. In the current version of BID, design actions are implemented using CRL-OPS rules, as illustrated in Figures 40 and 41.
OPS RULE

\( p \) \text{ d-rib-height-3} \\
\begin{align*}
& ( \text{blackboard} \ '12\text{-goal d-rib-height} \ 'l2\text{-f-obj <f-rib>}) \\
& ( \text{forging} \ '\text{material} \ '\text{titanium} \ '\text{equipment} \ '\text{hydraulic-press}) \\
& ( \text{rib} \ '\text{schema-name} <f-rib> \ '\text{height} <h>) \\
\end{align*}

\( \text{-->} \)

\( ( \text{cschema 'blackboard ('r-rib-dh (* $<h> -0.05)))} ) \)

ENGLISH TRANSLATION

If the current goal is to design blocker rib height, the forging material is a titanium alloy, and the forging equipment is a hydraulic press, then recommended reduction in rib height is 5%.

Note: BID can use the rib height suggested here only as an initial approximation, computing the actual value based on volume considerations.

Figure 40: An Example Design Rule to Recommend Blocker Rib Height.

Figure 40 illustrates a typical design rule for recommending blocker rib height, given that the forging material is a Titanium alloy, and that the forging equipment is a hydraulic press. The rib height suggested here is only a recommended value, and it is subject to variations based on volume considerations.
OPS RULE

\[ (p \text{ r-draft-1b}) \]
\[ (\text{blackboard 'l3-goal r-draft 'l2-f-obj <f-rib> 'l3-object <be>}) \]
\[ (\text{forging 'material steel 'equipment mechanical-press}) \]
\[ (\text{rib 'schema-name <f-rib> 'height <h>}) \]

\[ \rightarrow (\text{if (equalp 'inner-draft (get-value (get-value $<be> 'f-edge) 'draft-type))}) \]
\[ (\text{cschema 'blackboard ('recommended-draft 7))}) \]
\[ (\text{if (<= $<h> 5/4)}) \]
\[ (\text{cschema 'blackboard ('recommended-draft 5))}) \]
\[ (\text{cschema 'blackboard ('recommended-draft 7))}) \]

ENGLISH TRANSLATION

If the current goal is to recommend a blocker draft, the forging material is a steel, and the forging equipment is a mechanical press, then for inner drafts, recommended draft = 7 degrees; for outer drafts, if rib height <= 5/4", then recommended draft = 5 degrees, else recommended draft = 7 degrees.

Note: The draft angle recommended here is subject to changes due to other design considerations.

Figure 41: An Example Design Rule to Recommend Blocker Draft Angle.

Similarly, Figure 41 illustrates another design rule for recommending a draft angle, given that the forging material is a steel, and that the forging equipment is
a mechanical press. The draft angle suggested by this rule is also subject to changes due to other design considerations. For example, if the draft angle recommended by this rule is significantly higher than the corresponding draft angle in the finish forging, then the draft angle actually designed for the blocker may be smaller than the recommended value, because practical considerations dictate that the corresponding blocker and finisher drafts should not differ significantly.

5.3 HUMAN-MACHINE INTERFACE

Most computer programs need to provide some kind of human-machine interface, to enable communication between the user and the computer program. User-friendliness of these interfaces is in fact a much debated topic. Ideally, human-machine interfaces should be based on some natural language (such as English), and voice input/output. While a great deal of progress has been made in natural language understanding by computers, the systems developed are in general far from robust. Similarly, with respect to voice recognition systems, higher recognition capability is currently possible only with speaker dependent systems. Given that these technologies are still evolving and given the extensive computational burden imposed by them, they have not yet had significant impact on the user–interfaces currently provided. Instead, the interfaces are usually provided by means of a keyboard/CRT, and depending on the type of interface, a formal language (developed for the particular application at hand) is also used.
The types of user interfaces commonly provided can be classified into three categories: (a) *Command-driven* interface, where the computer program accepts a sequence of English-like commands, as in an operating system language. This type of interface is popular among experienced users, as they are familiar with the vocabulary of the language. From the point of view of simplicity of parsing and generating the sentences by the computer, this formal language should preferably have a small vocabulary and restricted syntax. (b) *Prompt-driven* interfaces, where the computer queries the user leading him through specific query paths. The user is thus locked into predefined query paths, and hence this interface is attuned to the novice user. (c) *Menu-driven* interface, where a series of options are provided to the user, allowing him/her to navigate freely among predefined options. This interface permits the user to migrate to more sophisticated menus, as he becomes more familiar with the system.

BID makes provisions for combining these three types of interfaces, to enhance flexibility and ease of use. At the highest level, BID uses a prompt driven interface. However, the user interface module, activated as soon as the design task begins, is implemented using a hierarchy of menus, as illustrated in Figure 42. Under option 5 at the top level menu of the user interface module, three other interface options are also provided, as discussed later.

The user interface provided in BID is implemented as a separate module. It can be activated by any LISP module within or outside BID, including the top level
of LISP. Of course, whenever activated, the interface assumes the availability of relevant global parameters and data structures used during the interaction. Given

Figure 42: Menu Options Provided in BID.
that the user interface can be activated at different stages during the design process, only menu options relevant for that particular stage of design are displayed. For example, once the design task is completed, the top level menu will not show the Modify System Parameters option, since it is useless to change these parameters after the design is complete. Also, from the hierarchy of menus, it can be noted that option 0 on any menu refers to EXIT. Choosing this option returns control to the previous level, which may be another menu, or a LISP program (including the top level of LISP). From the remaining menu options, those that have been implemented so far are discussed below.

5.3.1 Modify System Parameters

This option permits modification of various parameters which guide the system behavior. These options can be modified even as the design task is progressing, altering the system behavior dynamically. This dynamic change in the system behavior can be best appreciated when the significance of various system parameters is well understood. Accordingly, some of the more important system parameters are described below in some degree of detail, for the purpose of illustration.

Design Interaction: BID always permits user interaction after the design task is completed. In addition, by specifying this parameter appropriately, BID can be interrupted even while the design task is in progress, for inspection/change of some aspect of the design. This option also permits specification of the predetermined stages at which the program is to be interrupted. Currently, interaction is
permitted after: (a) reading input, (b) any rib design, and (c) any web design. If none of these interaction options are specified, the program runs in a completely automated mode, permitting interaction only after the entire design is completed. However, during automated design, the program displays brief information on the tasks being performed as shown in Figure 43, for information/verification.

**Common Drafts:** This option permits specification of the commonly used draft-angles, and it is provided since angles used for standard drafts may differ from one company to another. While designing a draft angle, the design system not only uses the design heuristics, but it also ensures that the angle selected is restricted to be one of the common drafts specified under this option. Of course, natural drafts are not subject to this restriction.

**Radius Increment:** While specifying corner or fillet radii in the British system, it is common practice in the industry to specify these radii values as a multiple of some suitable fraction of an inch. For example, given 1/32" as this fraction, all radii selected will be some multiple of 1/32".

---

13. Natural (or design) draft is a draft that is inherent in sloping or curved sides of a forging, or that is obtained by select tilting of the die impression. Natural draft eliminates the need for application of an additional draft.
Find internal representation? y
The internal representation module is being loaded.
Input read and internal representation started.
Lines defined.
Features identified.
Features defined.
Volumes specified.

Do you want to activate interaction? n

Design blocker section? y
The design module is being loaded.
Design is being initiated (e.g., reading input).
The OPS module is being loaded.

Section width being designed.
Blocker rib BUR-1 being designed.
Blocker rib BUR-2 being designed.
Blocker rib BLR-1 being designed.
Blocker rib BLR-2 being designed.
Blocker rib BLR-3 being designed.
Blocker web BUW-1 being designed.
Blocker web BLU-1 being designed.
Blocker web BLU-2 being designed.
Final volume calculations etc. being performed.

Do you want to activate interaction?

Figure 43: Screen Display when BID is Executed in Automated Mode.
5.3.2 Modify Display Parameters

The objective of this option is to provide a means to modify various display parameters. The parameters included so far permit changing only the display colors.

5.3.3 Display Design

This option is provided to display the status of design at any stage during the design process. Currently, this display is accomplished by writing a display file, and activating a spawned process to execute an in-house graphics program called DIGRAF [Tang 84].

5.3.4 Inspect/Change Design

BID implementation permits the program to be run in a completely automated mode, i.e., given an input file with a valid geometry and process conditions, BID can automatically generate an initial-guess design for the blocker section. However, given the evolutionary nature of BID development (and hence the incompleteness of its knowledge-base), BID also permits user interaction with the system at some predefined stages during design, so that the user can inspect the design as it progresses, and even make changes along the way as and when he sees fit. The user can make the system pause at these predefined stages during design, using one of the system parameters (design-interaction) described earlier. Once the system pauses at some stage of design, BID makes provisions for three different interfaces,
viz., Graphics Interface, Linguistic Interface, and LISP Interface. However, only the LISP interface is implemented so far, as described below.

LISP Interface: With this interface, the user is provided access to the full power of LISP, even as the design task is progressing. Whenever the design process is interrupted and the LISP option is enabled, it is possible to:

1. Inspect or change the current design by directly accessing the corresponding LISP data structures,
2. Inspect or change the data structures or functions in the LISP modules, thereby altering the program even as it is running, and
3. Spawn a different process and perform other activities at the operating system level, thereby suspending BID and keeping it in hybernation indefinitely.

While this option is very useful during the development stage, such complete access to the program is not desirable normally. Hence, there is a provision to disable this option. Accordingly, for this option to show-up, the programming-mode should be activated by calling a special function (a kind of password) and set to programmer. Without this function invocation, programmer-mode option as well as the LISP-interface option are both disabled.
5.3.5 Output Results

This option can be used to output geometry information and other processing conditions, in different forms. The output requested is written to different files, to be used for different purposes.

5.3.6 Program Mode

As explained in Section 5.3.5, this option does not even show-up normally, unless invoked explicitly by calling a special LISP function (analogous to a password). This option takes one of two values, viz., user and programmer. If it is activated and set to programmer, then only system permits access to the LISP interface. This program mode concept could be used in general to provide other access restrictions between different classes of users.

5.4 BID DESIGN EXAMPLES

The current version of BID was used to produce some example designs, to illustrate the design capabilities of BID. The designs shown here reflect the knowledge base currently contained in BID. For example, these design rules explicitly recognize three different forging materials and three types of forging equipment. However, factors such as lot size are considered only indirectly, in the form of compiled design knowledge, as will be explained later. Before further
discussion of complete designs, brief mention is made below regarding the intermediate stages of design.

As mentioned earlier, BID performs design incrementally, interrupting itself after each feature design and passing control to the user interface module. Two of these intermediate stages are illustrated in Figures 44 and 45, for an example forging cross section. Figure 44 refers to the status of design after some ribs are designed, while Figure 45 refers to the status of design after some webs are also designed. Of course, selection of these intermediate stages is not unique; it depends on the top level design plan currently under consideration. The stages shown in Figures 44 and 45 refer to the top level design plan which is currently part of the BID knowledge base. When alternate design plans are included, these intermediate stages could be different for different design plans.

Now, to illustrate complete designs, let us first consider a simple rib–web type forging cross section, often encountered in practice with forgings such as the connecting rod. For this example forging cross section, blocker designs were generated for four different combinations of forging material and forging equipment, as shown in Figures 46, 47, 48, and 49.

These Figures show the finish forging cross section superimposed on the blocker cross section, for easy comparison. They also indicate a certain percentage increase in blocker cross section area, as appropriate flash allowance. For example,
Figure 44: Intermediate Stage After Some Ribs are Designed by BID.

Figure 45: Intermediate Stage After All the Ribs and Some Webs are Designed by BID.
Figure 46: Blocker Design for a Simple Forging Cross Section, Assuming Titanium Forging and Hydraulic Press.
Figure 47: Blocker Design for a Simple Forging Cross Section, Assuming Titanium Forging and Hammer.
Figure 48: Blocker Design for a Simple Forging Cross Section, Assuming Steel Forging and Mechanical Press.
Example-Part 1

Process Variables
Aluminum Forging
Hydraulic Press

Volumes (in**3)
Forging: 7.969117
Blocker: 8.258845
Extra: 3.635637

Figure 49: Blocker Design for a Simple Forging Cross Section, Assuming Aluminum Forging and Hydraulic Press.
as seen in Figure 46, the blocker cross section area is approximately 2.67 percent more than the forging cross section area.

Considering Figure 46, with titanium as the forging material and hydraulic press as the forging equipment, it can be seen that the blocker has shorter ribs, thicker webs, and very generous radii, with approximately 3.64 percent extra volume for flash allowance. In the case of titanium forgings, it is often preferable to have a certain reduction in web thickness in the (final) finishing operation, to achieve desired material properties. Thus, the web is made thicker, with material flowing from webs to the shorter ribs during finish forging. However, the design considerations differ from one forging machine to another. In the case of hammers which are relatively fast with less contact time, the design generated by BID (based on some existing designs) is different, as shown in Figure 47. The rib here is deeper compared to the previous design with hydraulic press, which is slow acting but can deliver much higher deformation energy than a hammer.

Figure 48 illustrates another blocker design, with steel forged in a mechanical press. Steel parts are often made (say, for the automotive industry) in very large quantities, i.e., their lot sizes are very large. Hence, to avoid having to make too many die sets, it is preferable not to allow a lot of material to backward extrude around the fillets. This will wear out the dies very quickly, reducing die life. Instead, it is preferable to start with taller ribs in the finisher, and upset them to fill the finisher cavities. Accordingly, blockers for steel forgings are typically made with deeper cavities (or taller ribs).
In fact, according to a rule of thumb occasionally followed in the case of steel forgings, a blocker rib should be made narrower and taller in such a way that it provides the necessary volume required to fill the corresponding finisher rib cavity. Obviously, this is to avoid material flow from web to rib during finishing operation. Figure 48 reflects this thinking in generating the blocker designs for steel forged on a mechanical press. Note also that radii in this design are smaller than those in previous blocker designs, since material movement is confined mostly to upsetting, with relatively less flow around the fillet radii.

In the case of Figure 49, the blocker design was generated assuming the forging material to be aluminum, and forging equipment to be hydraulic press. Aluminum being relatively easy to forge, backward extrusion is a relatively less severe problem, and some material movement from webs to ribs across the fillets is permissible. Accordingly, blocker design in this case exhibits shallower ribs, but more generous radii for easy flow of material across the fillet.

The BID program was also used to generate three other blocker designs for the forging cross section already shown in Figure 12. These designs were generated under varying assumptions for forging material and forging equipment: Figure 50 with titanium and hydraulic press, Figure 51 with steel and mechanical press, and Figure 52 with aluminum and hammer. It can be seen that these blocker designs are based on the same principles as those discussed in the previous designs. In addition, the effect of localized volume balancing is seen more clearly here, due to
Figure 50: Blocker Design for the Forging Cross Section Shown in Figure 12, Assuming Titanium Forging and Hydraulic Press.
Figure 51: Blocker Design for the Forging Cross Section Shown in Figure 12, Assuming Steel Forging and Mechanical Press.
Figure 52: Blocker Design for the Forging Cross Section Shown in Figure 12, Assuming Aluminum Forging and Hammer.
the introduction of asymmetry in the cross section. In all three designs, the first lower web (BLW-1) thickness increased more than the second lower web (BLW-2), since BLW-1 has to supply more material to its neighboring ribs than BLW-2.

In the design shown in Figure 50, height of BLW-1 can be seen to be very close to that of BLR-2. In extreme cases, the web height may even increase beyond that of a neighboring rib. In general, this may not be desirable from a practical view point, suggesting the need for alternate design plans in such situations. One such design plan could be based on the concept of redesign. The particular web design specialist that could not provide sufficient volume could indicate failure to the blocker design specialist, which can in turn instruct the rib design specialist to redesign the rib to account for additional volume. In fact, the blocker design specialist may provide for this additional volume in various other ways, such as obtaining it from the opposite side of the cross section, through a bulge or appropriate modification of the features. Thus, a variety of design plans can exist at the blocker design specialist level, each applicable under different circumstances. The same argument as above also holds good at the individual feature design level. For example, redesign of a feature can be accomplished by modifying a particular design parameter, or a combination of them. Which particular redesign plan is to be selected depends on which plan's preconditions are satisfied at that instance.
5.5 COMPARISON WITH SIMILAR WORK ON BLOCKER DESIGN

The current work here has concentrated on two closely related and yet different aspects of blocker design: (a) Compilation of blocker design knowledge, and (b) Computerization of blocker design. Both these tasks are evaluated below, in the context of relevant earlier work in the respective areas.

5.5.1 Compilation of Blocker Design Knowledge

By its very definition, a knowledge-based system requires some sort of effort in acquiring the domain knowledge. However, as pointed out earlier, this is especially critical in the case of blocker design, since most earlier efforts in forging die design concentrated primarily on finisher die design [Haller 82, Thomas 80, Aluminum Association 80]. Any discussion found in the literature on blocker design is essentially qualitative, as the authors attempt to cover the topic in all its rich variety. No serious effort has been made to compile quantitative blocker design knowledge, especially from the point of view of computerizing the same. In this sense, the knowledge acquisition effort attempted here is relatively unique in its goal.
5.5.2 Computerization of Blocker Design

While discussing the state of the art in blocker design, it was pointed out that attempts to computerize blocker design have taken two distinct approaches: (a) interactive design, and (b) automatic design. The objective of the current computerization effort being generation of designs automatically, it is meaningful to compare BID only with those efforts which attempted automatic design. In fact, as suggested earlier, interactive design (graphic as well as linguistic) can be provided as an option within the BID system.

In the computer implementations for automatic design of blocker cross sections, the overriding limitation has been adherence to a fixed set of procedures, right down to the modification of design parameters. For example, the DIE FORGE system [Subramanian 77] uses fixed multiplication factors to modify the L shapes. Similarly, in their respective implementations, for designing fillet radii, Yu and Dean [Yu 85] uniformly applied the expression suggested by Bruchanov and Rebelski [Bruchanov 55], while Biswas and Knight [Biswas 76] used the exponential curves suggested by Chamouard [Chamouard 64]. Obviously, adoption of such fixed procedures is not desirable, since modification of the design parameters depends primarily on the prevailing process conditions i.e., the forging factors. In contrast, the BID system at least attempts to replace these fixed procedures with appropriate design heuristics. This is perhaps the most important feature distinguishing BID from similar attempts in automatic design of blocker sections.
In the case of the DIE FORGE system, apart from the fact that it may not be always feasible to divide a forging cross section into L shapes, mismatch between neighboring L shapes can be expected to occur in most designs, as shown in Figure S3. In these situations, the program goes through a series of steps to eliminate mismatch, by adjusting the heights of L shapes etc. On the other hand, the die designer never seems to encounter such matching problems. In addition, when the neighboring L shapes are modified in order to match them, the multiplication factors (which are supposed to implicitly reflect the design heuristics) are also indirectly being changed, thereby violating even the design heuristics. This is obviously not desirable, particularly since mismatch can be expected to occur with most neighboring L shapes. Thus, while the division of forging cross section into L shapes provides an excellent engineering solution to the complex problem of forging die design, it also leads to some unnatural problems in implementation. The BID system avoids this mismatch problem by dividing the forging cross section into ribs and webs, which seems to better reflect the designers thinking.

As discussed in Chapter 2, the proposal to NSF [Altan 84] assumes the same division into L shapes, and suggests using Design Weighting Factors (DWFs). While this seems to alleviate the problem of fixed multiplication factors, obtaining such detailed DWFs is a very difficult task. More importantly, this approach neglects the effect of interplay among process variables in designing blocker sections. To illustrate the point, let us say that modification of fillet radius is affected, among others, by the combination of forging material and forging equipment. However,
the DWFs approach suggests two independent DWFs, one specifying how the forging material modifies fillet the radius, and the other specifying how the forging equipment modifies the fillet radius. Many such cases can be expected with even
more interdependence among the forging factors, and these are undermined by the 
DWF approach. The BID system, by not laying down any uniform procedure, 
permits a variety of design heuristics to be included in the system. The relative 
priorities of different chunks of knowledge can be set a priori in the system, to 
ensure that appropriate piece of knowledge is used depending on the forging 
factors at that instance.

Thus, the BID system enables automatic design of blocker cross sections, 
taking process variations into consideration. Even though BID makes these 
provisions for incorporating the process dependent design knowledge into the 
system, how well it can design blocker cross sections still depends on how much of 
the blocker design knowledge can be understood and incorporated into its 
knowledge base.
CHAPTER 6
CONCLUSIONS AND RECOMMENDATIONS

6.1 SUMMARY OF RESEARCH

As set out in the research objective, geometric design was studied in the context of the blocker design problem, and a model of the design process applicable for a class of geometric design problems was developed. An architecture reflecting this model was then used to implement a computer system for designing blocker forgings, using the knowledge-based approach. In parallel, effort was also made to compile blocker design knowledge, to help develop this computer system. The computer system, called Blocker Initial-guess Design (BID), is currently implemented in LISP/Knowledge-Craft, using FORTRAN for volume calculations and graphics display. This dissertation summarized the development effort so far, including the design model, BID architecture, and to some extent, the blocker design knowledge.

As suggested earlier, the blocker design problem (and BID) can be viewed as a test-bed for the more generic problem of geometric design. Accordingly, the
concepts developed in this dissertation in terms of geometry representation, problem solving in geometric design, and the implementation architecture, are relevant for a class of geometric design problems where designing a geometry can be viewed in terms of designing a set of individual features (such as ribs and webs).

6.2 SUMMARY OF CONTRIBUTIONS

The major contributions of this research are as follows.

(1) Geometry Representation: The principal contribution of this research is in demonstrating the usefulness of a topological description based representation of geometry in the computer, to support computer based automation of design and planning activities. It was suggested that a hierarchic representation, where the geometric details (at the lower levels of the hierarchy) are embedded within the higher level topological description of the object, provides a powerful framework for representing geometry in the computer. As discussed in Chapter 4, such a representation, besides being transparent, can (a) support high level topological queries about the geometry, (b) provide a linguistic means for human–machine interaction, which could be useful for interactive design modification as well as specification of design instructions as part of the input itself, and (c) facilitate extension to three dimensional design and planning problems.
Architecture for Geometric Design: The geometric design problems require relatively complex design primitives, which involve complex spatial reasoning and accurate geometric manipulations, instead of purely linguistic reasoning. This renders application of the knowledge-based approach to geometric design problems harder compared to application to some other domains. The architecture suggested here for geometric design manages this complexity by effectively combining the advantages of both symbolic processing (LISP) and numeric processing (FORTRAN), as well as integrating the use of relevant KBS paradigms. The architecture is based on distribution of knowledge and associated control. The design decisions are classified into design plans and design actions, where a design action typically refers to atomic actions such as changing a particular design parameter, while a design plan refers to overall planning of the sequence of design actions to be performed under a specific design situation. It was suggested that complex design actions may require further design goals. In the context of this particular architecture, a qualitative measure for the complexity of a design problem was suggested, as discussed in Chapter 3. Also, the architecture suggested makes it relatively easy to modify the knowledge base.

Blocker Design Knowledge Acquisition: Given that very little information is available in the literature on the design of blockers, and whatever information is available is in the form of a few qualitative design guidelines, trying to acquire quantitative blocker design knowledge is in itself an unique effort.
(4) The BID System: Major advantages offered by the BID system in computerizing blocker design can be summarized as below.

1. Design Heuristics Instead of Fixed Procedures: Most computer applications related to blocker design have concentrated on interactive graphic design. Given that very little quantitative blocker design knowledge is available in the literature, the few systems that attempted automatic design of blocker cross sections [Yu 85, Subramanian 77] resorted to using some fixed procedures, rendering them inflexible. The BID system is based on the recognition of the need to include the rich variety of blocker design knowledge, and hence makes explicit provision to incorporate this variety of design heuristics, as far as they can be identified and codified into the program.

2. Human-Aided Machine Design: The BID system implementation permits it to be run either in a fully automated mode, or in a semi-automated mode (allowing interruption after each feature design, for possible interactive modification or verification). However, it is expected to be used more in a semi-automated mode, (say) as a design assistant, where the designer watches what BID does automatically and intervenes only when it is necessary to make some modifications. Thus, computer-based design can move from machine-aided design paradigm (CAD) to human-aided design paradigm. The system's usefulness lies in the fact that it will tirelessly use all the design knowledge incorporated into it, without forgetting the details.

3. Blocker Design Shell: The BID system architecture makes it comparatively easy to modify the knowledge base, i.e., to add or modify design plans and design actions. Of course, it is easier to modify design actions compared to design plans, since design plans are more comprehensive. Given this flexibility in modifying the knowledge base, since design heuristics used by designers from different companies differ from each other, each company could modify the knowledge base to reflect its own design practices. Thus, the BID system can serve as a shell for blocker design.
4. Automatic Localized Volume Balancing: From a practical viewpoint, one of the most useful aspects of BID is its capability to perform localized volume balancing automatically, since this is a comparatively tedious task to perform using an interactive design system, let alone doing it manually.

5. Study of Generic Issues of Geometric Design: An important generic contribution of this research is that the BID system provides a framework to study the issues in developing such geometric design systems. For example, when sufficiently developed, the knowledge organization and control structures used in BID can be used as a kind of shell for use in similar geometric design problems.

6.3 DIRECTIONS FOR FUTURE RESEARCH

In the current development effort, it is planned to develop a complete LISP version of BID, replacing the use of Knowledge Craft (KC), as well as extend the knowledge base. As part of this conversion/extension, Schemata from KC will be replaced with appropriate implementation for frames, and CRL-OPS will be replaced using frames with procedural attachment. Besides this implementation plan, possible directions for future research are discussed below, in the context of the contributions mentioned earlier.

(1) Geometry Representation: The geometry recognition and representation used here are for specific use with rib-web type cross sections, even though they would work with any complex variations to the basic rib-web shapes (including
some which may not even look like rib-web shapes). It is an interesting research topic, although very complex, to study what kinds of feature extraction and geometry representation methodologies are suitable to support a variety of computer-based automation applications. There is in fact work being done dealing with related issues [Pratt 84, Weiler 85, Henderson 84b].

(2) Architecture for Geometric Design: Knowledge-based approach to Mechanical Design is a relatively new, although quite active, research topic. An important research direction would be to investigate the kinds of generic knowledge organization and control constructs required to facilitate application of the KBS approach in domains such as geometric design, which involve complex spatial reasoning and geometric manipulations as part of the problem solving.

(3) Blocker Design Knowledge Acquisition: In the context of blocker design, the design knowledge being compiled under the NSF project will be summarized in the final report to NSF, due in June 1987. However, more research is needed in this knowledge acquisition area. Knowledge-based systems, as in the case of BID, are typically developed by explicit encoding of domain knowledge into the system. This kind of painstaking hand-crafting is obviously not desirable, if it can be avoided. Instead, the system should be able to by itself acquire new knowledge as well as modify the existing knowledge. This is one of the most important research topics from the perspective of developing knowledge-based systems. In order for the system to possess this kind of capability, it should have an epistemological
understanding of the domain, although providing the system with this understanding is obviously a very difficult task.

(4) The BID System: Some of the useful enhancements to the BID system are discussed below.

1. Enhance Knowledge-Base: Enhancement of the knowledge base includes (a) incorporating more design plans and design actions, (b) providing deeper knowledge, to the extent possible, and (c) separate knowledge bases for forging materials and equipment.

2. Design Modification: During the intermediate stages when design is interrupted, or after a completed design, it is useful to be able to interactively modify the design. This capability could be provided in two different ways:

   a. Graphics Interface: This interface represents the commonly used interactive graphics, and it could be provided using a simple graphics program (perhaps preferable), or a CAD/CAM system.

   b. Linguistic Interface: This interface could be provided by defining a formal language based on the internal representation and the associated naming convention, as discussed in Chapter 4.

For maximum flexibility, these two interfaces could complement each other in their usage, instead of being used independently.

3. Explanations: Given the evolutionary nature of knowledge-based systems development, it is imperative to provide them with some sort of explanation capability. This is useful not only in gaining confidence in a system's response, but also in further refinement of its knowledge base. However, given that the current LISP/KC version of BID is to be converted to a complete LISP implementation, no attempt has been made towards providing any explanation capability in the current CRL-OPS based knowledge base. The explanations provided by the KBS are often
based on canned expressions explicitly stored in the computer. Instead of such What kind of explanations, it would be clearly more useful if the systems can provide the how explanations, even though it is a harder task. In providing this kind of capability, causal models are very helpful.

4. Backtracking the Design Steps: In some design situations, it could be useful to backtrack and undo certain design actions, to pursue an alternate design plan. While this capability has not been contemplated in the implementation of BID, it would be clearly useful to investigate this possibility.

5. Extension to 3D Design: Given the complexity of the blocker design problem, automating 3D blocker design is an extremely difficult task. However, reflecting the common practice of designing blockers section by section, BID could be extended to at least simple 3D geometries, by designing the sections one by one and then assembling them. One of the requirements for such an effort would be the ability to say how material moves across the sections. The high level description provided by the internal representation could facilitate at least this aspect of the 3D design problem.

In conclusion, from the perspective of geometric design, the most useful research direction would be to abstract out the knowledge organization and control constructs from the BID system, in order to develop a shell which could be applicable for similar geometric design problems.
BIBLIOGRAPHY

[Akgerman 73] Akgerman, N., and Altan, T.
Computer-Aided Design and Manufacturing of Forging Dies for Structural Parts.


[Altan 83] Altan, T., Oh, S. I. and Gegel, H.
Metal Forming Fundamentals and Applications.
American Society for Metals, 1983.

[Altan 84] Altan, T., and Badawy, A.
Handbook of Rules and Guidelines for Designing Blocker / Preform in Closed Die Forging With and Without Flash.
Technical Report, Battelle Columbus Division, Columbus, Ohio, Mar, 1984.
Proposal to National Science Foundation.

[Aluminum Association 80]
Aluminum Forging Design Manual

[Aquesbi 83] Aquesbi, A. et. al.
An Expert System for Computer Aided Mechanical Design.
Badawy, A., Akgerman, N. and Altan, T.  
*Computer Aided Design of Blocker Sections Using a FORTRAN-Based Interactive Graphics Program.*  
Technical Report 14, Battelle Columbus Division, Columbus, Ohio, Jan, 1982.  
Prepared for Member Companies – Group Program on Close Tolerance Forging Process Technology.

Balzer, R.  
Transformation Implementation: An Example.  
*IEEE Transactions on Software Engineering* SE-7:3-14, 1981.

Barr, Avron, and Feigenbaum, Edward A. (editors).  
*The Handbook of Artificial Intelligence.*  

Begg, Vivienne.  
*Developing Expert CAD Systems.*  
Subject covered is CAD in electronics.

Biswas, S. K. and Knight, W. A.  

Biswas, S. K. and Knight, W.  
Towards an Integrated Design and Production System for Hot Forging Dies.  

Biswas, S. K. and Rao, K. M.  
Flow of Metal In Constrained Plane-Strain Extrusion Forging: Part I.  

Brachman, R.  
What is in a Concept: Structural Foundations for Semantic Networks.  
[Brown 82] Brown, C.M.

[Brown 83] Brown, D.C. and Chandrasekaran, B.
An Approach to Expert systems for Mechanical Design.

[Brown 86] Brown, David C. and Chandrasekaran, B.
Knowledge and Control for a Mechanical Design Expert System.

[Brownston 85]
Brownston, L., Farrell, R., Kant, E., and Martin, N.
*Programming Expert Systems in OPS5: An Introduction to Rule-Based Programming*.

[Bruchanov 55]
Bruchanov, A. N. and Rebelski, A. W.
*Gesenkschmieden und Warmpressen, (in German, translated from Russian), Die Forging and Warm Forging*.

[Buchanan 78] Buchanan, B. G., and Feigenbaum, E. A.
DENDRAL and Meta-DENDRAL: Their Applications Dimension.

[Buchanan 84] Buchanan, B., and Shortliffe, E. H.
The MYCIN Experiments of the Stanford Heuristics Programming Project.

[Bullers 80] Bullers, W. I., Nof, S. H. and Whinston, A. B.
Artificial Intelligence in Manufacturing Planning and Control.

*Forging Handbook*.
[CAM-I 81] CAM-I Illustrated Glossary of Workpiece Form Features
Computer Aided Manufacturing International, Inc. (CAM-I),
Arlington, Texas, 1981.

[Chamouard 64]
Chamouard, A.
Estampage et Forge (in French) Closed-Die Forging.

[Chandrasekaran 82]
Chandrasekaran, B. and Mittal, S.
Deep Versus Compiled Knowledge Approaches to Diagnostic
Problem-solving.

[Chandrasekaran 83a]
Chandrasekaran, B.
Towards a Taxonomy of Problem Solving.

[Chandrasekaran 83b]
Chandrasekaran, B.
Expert Systems: Matching Techniques to Tasks.
Technical Report, Air Force Office of Scientific Research, Bolling
Annual Report for Research on Distributed Knowledge Base Systems
for Diagnosis and Information Retrieval.

[Chandrasekaran 86]
Chandrasekaran, B.
Generic Tasks in Knowledge-Based Reasoning: High Level Building
Blocks for Expert System Design.

[Chang 82]
Chang, T. C.
TIPPS – A Totally Integrated Process Planning System.
PhD thesis, Virginia Polytechnic Institute, Blacksburg, Virginia, Nov,
1982.

[Charniak 80] Charniak, Eugene, Riesbeck, C. K. and McDermott, D.V.
Artificial Intelligence Programming.
[Charniak 85]
Charniak, E., and McDermott, D.
*Introduction to Artificial Intelligence.*

[Choi 82]
Choi, B. K.
*CAD/CAM Compatible Tool-Oriented Process Planning System.*
PhD thesis, School of Industrial Engineering, Purdue University, West Lafayette, Indiana, Dec, 1982.

[Clancey 83]
Clancey, William J.

[Clocksin 81]
Clocksin, W.F. and Mellish, C.S.
*Programming in Prolog.*
Springer Verlag, 1981.

[Cullingford 82]
Cullingford, R. E., et. al.
Automated Explanations as a Component of a Computer-Aided Design System.

[Davis 76]
Davis, R.
*Applications of Metalevel Knowledge to the Construction, Maintenance, and Use of Large Knowledge Bases.*

[Davis 83]
Davis, R.
Expert Systems: Where are We? And Where Do We Go from Here?

[DeKleer 81]
DeKleer, J. and Brown, J. S.
Mental models of physical mechanisms and their acquisition.
In Anderson (editor), *Cognitive Skills and Their Acquisitions.*
[Dixon 83] Dixon, J. R. and Simmons, M. K.
*Computers in Mechanical Engineering* :10–18, November, 1983.

[Dixon 84] Dixon, J. R. and Simmons, M. K.
Expert Systems for Engineering Design: Standard V-belt Drive
Design as an Example of the Design–Evaluate–Redesign
Architecture.
In *Proceedings of Computers in Engineering Conference,*

[Drabing 80] Drabing, L. G.
Practical Forging Die Design.

[Dreyfus 72] Dreyfus, H. L.
*What Computers Can't Do.*

[Duda 79] Duda, R. O., Gascbnig, J. and Hart, P. E.
Model Design in the Prospector Consultant System for Mineral
Exploration.
Electronic Age,* pages 153–167. Edinburgh University Press,
1979.

[Feigenbaum 77]
Feigenbaum, E. A.
The Art of Artificial Intelligence: Themes and Case Studies of
Knowledge Engineering.

Use of Electronic Geometry Transfer and FEM Simulation in the
Design of Hot Forging Dies for a Gear Blank.

[Forgy 77] Forgy, C and McDermott, J.
OPS: A Domain–independent Production System.
[Fox 82] Fox, M. F.
Job Shop Scheduling: An Investigation in Constraint-Directed Reasoning.

[Goldstein 79] Goldstein, I. P. and Roberts, B.
Using Frames in Scheduling.

[Gomez 81] Gomez, F. und Chandrasekaran, B.
Knowledge organization and distribution for medical diagnosis.

[Haller 71] Haller, H. W.
Handbuch des Schmiedens (in German), Forging Handbook.
Carl Hanser Verlag, Munich, 1971.

[Haller 82] Haller, H. W.
Praxis des Gesenkschmiedens (in German), Closed-die Forging Practice.
Carl Hanser Verlag, Munchen, 1982.

[Harmon85 85] Harmon, P., and King, D.
Artificial Intelligence in Business.

Building Expert Systems.

[Henderson 84a] Henderson, Mark R.
Feature Recognition in Geometric Modeling.
In 13th Annual Meeting and Technical Conference on Man or Machine – A Choice of Intelligence, pages 5-1 to 5-12.
[Henderson 84b]
Henderson, Mark R.
*Extraction of Feature Information from Three Dimensional CAD Data.*

[Hillyard 82] Hillyard, Robin.
The Build Group of Solid Modelers.

[Hofstadter 79]
Hofstadter, D.
*Gödel, Escher, Bach: An Eternal Golden Braid.*

*Forging Industry Handbook.*


[Knowledge Craft 85]
*Knowledge Craft Manual Guide, Version 3.0*

[Kyprianou 80]
Kyprianou, L. K.
*Shape Classification in Computer–Aided Design.*

[Lange 58] Lange, K.
*Gesenkschmieden von Stahl (in German), Closed–Die Forging of Steel.*

[Lange 77] Lange, K., and Meyer–Nolkemper, H.
*Closed–Die Forging (in German).*
[Lange 85] Lange, Kurt, et. al. (editors).
Handbook of Metal Forming.

[Latombe 79] Latombe, J. C.
Failure Processing in a System for Designing Complex Assemblies.

[Lyman 70] Lyman, Taylor (editor).

R1: A Rule-Based Configurer of Computer Systems.

[Medland 86] Medland, A. J.
Springer-Verlag, 1986.

[Minsky 75] Minsky, M.
A Framework for Representing Knowledge.
In P. H. Winston (editor), The Psychology of Computer Vision,

A Knowledge-based Approach to Design.

[Mittal 80] Mittal, S.
Design of a Distributed Medical Diagnosis and Data Base System.


[Nau 82] Nau, D. S.
Expert Systems and Their Applicability to Automated Manufacturing.

[Newell 69] Newell, A.
Heuristic Programming: Ill-structured Problems.

[Nii 86] Nii, Penny H.

ALPID - A General Purpose FEM Program for Metal Forming.

[Oh 86] Oh, S. I., Malas, J., Gegel, H., and Altan, T.
Practical Experience and Future Developments in Computer Applications in Forging.

[Pearl 84] Pearl, Judea.

[Pratt 84] Pratt, M.J.
Solid Modeling and the Interface Between Design and Manufacture.

[Rasdor 85] Rasdorf, W. J.
Perspectives on Knowledge in Engineering Design.
[Requicha 80] Requicha, A.A.G.

[Requicha 82] Requicha, A.A.G. and Voelcker, H.B.
Solid Modeling: A Historical Summary and Contemporary
Assessment.

[Requicha 83] Requicha, A.A.G. and Voelcker, H.B.
Solid Modeling: Current Status and Research Directions.

[Rosen 67] Rosen, Saul.
Programming Systems and Languages.

[Rychener 83] Rychener, M. D.
Expert Systems for Engineering Design: Experiments with Basic
Techniques.

[Sabroff 68] Sabroff, A. M., Boulger, F. W. and Henning, H. J.
Forging Materials and Practices.
Reinhold Book Corporation, 1968.

[Schank 84] Schank, Roger C., and Childers, Peter.
The Cognitive Computer: On Language, Learning and
Artificial Intelligence.

[Scherlis 83] Scherlis, W. and Scott, D.
First Steps Towards Inferential Programming.
In Proceedings of IFIP Congress 83 – Invited paper. North
Holland, 1983.

[Shneiderman 86] Shneiderman, Ben.
Designing the User Interface: Strategies for Effective
Human–Computer Interaction.
[Shortliffe 76] Shortliffe, E. H.

*Computer-Based Medical Consultations: MYCIN.*

[Simon 69] Simon, Herbert A.

*The Sciences of the Artificial.*

[Sowa 84] Sowa, John F.

*Conceptual Structures: Information Processing in Mind and Machines.*

[Spies 59] Spies, K.

*Preforming in Forging and the Preparation of Reducer Rolling (in German).*


Using Syntactic Pattern Recognition to Extract Feature Information from a Solid Geometric Data Base.


[Steel 84] Steele, Guy L. Jr.

*Common LISP: The Language.*

[Stefik 81] Stefik, M.

Planning with Constraints (MOLGEN: Part I).


[Subramanian 77] Subramanian, T. L., Akgerman, N. and Altan, T.

*Application of Computer-Aided Design and Manufacturing to Precision Isothermal Forging of Titanium Alloys: Volumes 1, 2 and 3.*


[Watermann 86]
Watermann, Donald A.

[Weiler 85]  
Weiler, Kevin.
Edge-Based Data Structures for Solid Modeling in Curved-Surface Environments.

[Weiss 84]  
Weiss, S. M., and Kulikowski, C. A.

[Winston 79]  
Winston, Patrick H.
*Artificial Intelligence.*
Addison-Wesley, 1979.

[Winston 81]  
Winston, P. H. and Horn, B.K.P.
*LISP.*
Addison-Wesley, 1981.

[Woo 84]  
Woo, Tony C.
Interfacing Solid Modeling to CAD and CAM: Data Structures and Algorithms for Decomposing a Solid.

[Yu 85]  
Yu, G. B. and Dean, T. A.
APPENDIX A.

OVERVIEW OF THE DESIGN PROCESS IN FORGING

A.1 THE FORGING PROCESS IN MANUFACTURING

Manufacturing can be in general characterized as the process of transforming a given material into some useful product. The processes for manufacturing metal parts can be classified into five general areas: (a) Primary shaping, (b) Metal Forming, (c) Metal Cutting, (d) Metal Treatment, and (e) Joining. Among these manufacturing processes, metal forming represents a highly significant group of processes. It is especially attractive in cases where (a) the part geometry is of moderate complexity and the production volumes are large, so that tooling costs per unit product can be kept low — for example, in automotive applications; and (b) the part properties and metallurgical integrity are extremely important, in examples such as load-carrying aircraft and jet engine and turbine components.

14. This appendix is compiled based on the literature on forging technology, especially from [Altan 83]
Metal forming processes can be classified into massive forming and sheet forming processes. In both cases, the surfaces of the deforming material and of the tools are in contact, and friction between them has a major influence on the process. In massive forming, the input material is in billet, rod or slab form, and a considerable increase in the surface-to-volume ratio occurs in the formed part. Processes which fall under this category have the following distinguishing features:

- The workpiece undergoes large plastic deformation, resulting in appreciable change in shape or cross section.
- The portion of workpiece undergoing permanent (plastic) deformation is generally much larger than the portion undergoing elastic deformation; therefore, elastic recovery after deformation is negligible.

Examples of massive forming processes are extrusion, forging, rolling and drawing. Among these processes, the forging process in turn has many variations such as open-die forging, closed-die forging without flash, closed-die forging with flash, forward extrusion forging, and backward extrusion forging. This is in addition to the variations due to differences in temperatures of the forging materials.
By convention, closed-die forging with flash is considered to be a hot working operation. This process is described in brief in Figure 54. As shown in this figure, two or more dies move toward each other to form a metal billet at a suitable temperature, into a shape determined by the die impressions. The process is capable of producing components of high quality at moderate cost. They offer a high strength-to-weight ratio, toughness, and resistance to impact and fatigue. In terms of tonnage, more than half of the forgings produced are used in automobiles. Approximately one-fourth of the total output of the forging industry is used in producing trucks, tractors and off-highway equipment; the remainder is used in the manufacture of aircraft, railroad and mining equipment and in other general mechanical and energy related engineering production.

In closed-die forging, a material must satisfy two basic requirements: (a) the material strength (or flow stress) must be low so that die pressures can be kept within the capabilities of practical die materials and constructions, and (b) the capability of the material to deform without failure (forgeability) must be sufficient to allow the desired amount of deformation.

15. This is the forging process of interest in this dissertation
**Definition:** In this process, a billet is formed (hot) in dies (usually with two halves) such that the flow of the metal from the die cavity is restricted. The excess material is extruded through a restrictive narrow gap and appears as flash around the forging at the die parting line.

**Equipment:** Anvil and counter-blow hammers, hydraulic, mechanical and screw presses

**Materials:** Aluminum alloys, magnesium alloys, beryllium, copper alloys, carbon and alloy steels, stainless steels, nickel alloys, titanium and titanium alloys, iron, cobalt and nickel-base superalloys, columbium and columbium alloys, tantalum and tantalum alloys, molybdenum and molybdenum alloys, tungsten alloys (all forgeable materials)

**Process Variations:** Closed-die forging with lateral flash, closed-die forging with longitudinal flash, closed-die forging without flash

**Application:** Production of forgings for automobiles, trucks, tractors, off-highway equipment, aircraft, railroad and mining equipment; general mechanical industry and energy-related engineering production

**Figure 54: Closed-die Forging with Flash.**
A.3 DIE DESIGN IN CLOSED-DIE FORGING WITH FLASH

The main objective of forging-process design is to ensure adequate flow of metal in the dies so that desired finish part geometry can be obtained without any external or internal defects. Metal flow is greatly influenced by part or die geometry. Often, several operations are needed to achieve gradual flow of metal from an initially simple shape (cylinder or round-cornered square billet) into a more complex shape of the final forging.

Figure 55: Suggested Preform Cross Sections for Various H-Shape Forgings in Steel. [Lange 77]
For example, Figure 55 illustrates the intermediate stages (or preforming operations) typically suggested for various H-shape forgings in steel. It can be seen that more preforming operations are recommended as the rib height to width ratio increases, i.e., as the complexity of the forging increases. For instance, Figure 56 illustrates the various preforming operations necessary to forge a relatively complex part. The round bar from rolled stock is (a) rolled in a reducer roller for volume distribution, (b) bent in a die to provide the appropriate shape, (c) blocked in a blocker die cavity, and (d) finish forged.

By convention, parts produced by the initial stages of metal distribution, using operations such as rolling and bending, are called *preforms*, while those produced by the stage just before the finishing operation, using the closed-die forging process, are called *blockers*. If any parts are produced before the blocking operation using the closed-die forging process, they are called *pre-blockers*. Occasionally, the word preform is used in a more generic sense, to refer to parts produced by all stages prior to the finish forging operation, i.e., including blockers and pre-blockers.

Proper design of dies for preforming (including blocking) operations is one of the most important aspects of closed-die forging, to achieve adequate metal distribution. With proper preform/blocker die design, defect free metal flow and complete die filling can be achieved during the finish forging operation, and metal losses into the flash can be minimized. In designing finisher and blocker dies for
hot forging, common practice is to consider some critical cross sections of a forging, where metal flow is plane strain or axisymmetric. These cross sections are often called planes of metal flow, as illustrated schematically in Figures 57 and 58. Such sectioning approximates, in two dimensions, the complex three-dimensional metal flow in a practical forging, so that design of blocker and preform dies can be achieved by designing them along these planes of metal flow.
Figure 57: Planes and Directions of Metal Flow During Forging of Two Simple Shapes: (a) Planes of Flow, (b) Finish Forged Shapes, and (c) Directions of Flow. [Altan 73]

Figure 58: Planes and Directions of Metal Flow During Forging of a Relatively Complex Shape: (a) Planes of Flow, (b) Finish Forged Shapes, and (c) Directions of Flow. [Altan 73]
Design of dies for forging will have to consider many factors of the associated forging process. While not exhaustive, some of the important factors that need to be considered are briefly discussed below.

Material Characteristics: Workpiece material is probably the single most important factor affecting economical forging design. The choice of workpiece material depends on many factors, such as intended usage, desired properties, corrosion resistance, density, availability, material cost, and elevated-temperature strength. It also has a definite bearing on the blocker design. For example, the design limit for a nickel-base superalloy requires more stock and subsequently more machining than producing the same part from aluminum. This is because the superalloy is a more difficult material to forge than aluminum.

Forging Equipment: Forging equipment influences the forging process, since it affects deformation rate and temperature conditions, and it also determines the rate of production. A sound and fundamental understanding of the basic interactions between process and equipment variables is required, before successful blocker design can be achieved. The principal process and equipment variables and their interactions in hot forging under presses are schematically illustrated in Figure 59, where a line between two blocks indicates that one variable influences the other. The behavior of metals during forging is influenced by the time necessary to complete the plastic shaping. Thus, it is important to recognize that the basic difference between the types of equipment lies in their forging velocities or rates.
of deformation. Forging hammers, for instance, deform metals at rates of deformation on the order of 100 times the rate of hydraulic presses.

Finished Part Shape Difficulty: One of the main factors affecting the flash dimensions, the stock volume, and the blocker geometry, is the geometric complexity of the part. Therefore, it is important to develop a simple method for measuring the complexity of a given finisher cross section.

Figure 59: Relationships Between Process and Machine Variables in Hot Forming Processes Conducted in Presses. [Altan 73]
In a general sense, spherical and blocklike shapes are the easiest to forge in impression or closed dies. Parts with long, thin sections or projections (webs and ribs) are more difficult to forge because they have more surface area per unit volume. Such variations in shape maximize the effects of friction and temperature changes, and hence influence the final pressure required to fill the die cavities. There is a direct relationship between the surface-to-volume ratio of a forging and the difficulty of producing that forging. The ease of forging more complex shapes depends on the relative proportions of vertical and horizontal projections on the part.

As shown in Figure 60, majority of the forgings can be classified into three basic groups [Spies 59]: compact shapes, disk shapes, and long shapes. These three groups are further divided into subgroups, depending on the presence and type of elements subsidiary to the basic shape. This shape classification can be useful for practical purposes, such as for estimating costs and for predicting preforming steps. However, this method is not entirely quantitative and requires some subjective evaluation based on past experience.

A quantitative value called the shape difficulty factor, based on longitudinal and lateral shape factors, has been suggested in the literature [Teterin 68] for expressing geometrical complexities of round (axisymmetric) forgings. This factor expresses the complexity of a half cross section of a round forging with respect to that of the circumscribing cylinder. In round forgings, during the forging
SHAPE CLASS 1
COMPACT SHAPE

SHAPE CLASS 2
DISC SHAPE

SHAPE CLASS 3
OBLONG SHAPE

<table>
<thead>
<tr>
<th>SUB-GROUP</th>
<th>NO SUBSIDIARY ELEMENTS</th>
<th>WITH HUB</th>
<th>WITH HUB AND HOLE</th>
<th>WITH RIM</th>
<th>WITH RIM AND HUB</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>102</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>103</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>104</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SUB-GROUP</th>
<th>NO SUBSIDIARY ELEMENTS</th>
<th>WITH HUB</th>
<th>WITH HUB AND HOLE</th>
<th>WITH RIM</th>
<th>WITH RIM AND HUB</th>
</tr>
</thead>
<tbody>
<tr>
<td>21</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SUB-GROUP</th>
<th>NO SUBSIDIARY ELEMENTS</th>
<th>WITH HUB</th>
<th>WITH HUB AND HOLE</th>
<th>WITH RIM</th>
<th>WITH RIM AND HUB</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SUB-GROUP</th>
<th>NO SUBSIDIARY ELEMENTS</th>
<th>SUBSIDIARY ELEMENTS PARALLEL TO AXIS OF PRINCIPAL SHAPE</th>
<th>WITH OPEN OR CLOSED FORE ELEMENT</th>
<th>WITH TWO OR MORE SUBSIDIARY ELEMENTS OF SIMILAR SIZE</th>
</tr>
</thead>
<tbody>
<tr>
<td>311</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>312</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>313</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>314</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 60: Classification of Forging Shapes. [Spies 59]
operation, the material is moved laterally (toward the ends of the cylinder) from the center, which is considered to be at the neutral axis. In a nonsymmetric forging, the material is still moved out laterally from a neutral surface. Thus, once this neutral surface is defined, a shape difficulty factor can be calculated even for nonsymmetric forgings.

Forging Tolerances: Tolerances in forgings are affected by numerous variables. Based on the accuracies obtainable as a result of the cumulative effect of all these variables, forgings can be classified as below.

- **Precision/Net Forgings**: In this type of forgings, the functional surfaces (e.g., the surface of a gear tooth) do not require any finishing operation.
- **Close Tolerance/Near Net Forgings**: These forgings require one finishing operation (machining or grinding) on the functional surfaces.
- **Conventional Forgings**: Most surfaces would be finished, usually by machining.

**Effect of Lot Size**: Lot size influences accuracy of forgings and die life. As the lot size increases, blockers should have generous radii, and material movements (especially sliding movements) should be minimized as much as possible.
A.4 COMPUTER APPLICATIONS IN CLOSED-DIE FORGING WITH FLASH

Traditionally, the process of forging die design is carried out manually, using empirical design guidelines, experience, and intuition. The steps involved in this design process can be summarized as below.

- Conversion of the available machined part geometry into a forging geometry by using guidelines associated with design of forgings and limitations of the forging process.

- Design of finisher dies, including determination of flash dimensions, forging stresses and forging load. In some cases, it may even be appropriate to calculate die stresses and modify the die geometry in critically stressed areas of the die to reduce the probability of premature die failure.

- Design of blocker or preblocker dies: This includes calculation of forging volume, including flash allowance, and the estimation of blocker and preblocker die geometry (including web thicknesses, rib heights, and fillet and corner radii).

- Design of preform and estimation of stock size: This includes prediction of desired metal distribution in the stock (by preforming or busting operations) prior to forging in the blocker die.

Once these design steps are completed, the forging dies are conventionally manufactured by (a) directly machining from a die block, (b) making a solid model and copy milling, or (c) making a graphite electrode and electrodischarge machining (EDM) the dies. The graphite electrodes, in turn, are manufactured by (a) copy
milling (b) abrading using a special abrading machine, or (c) numerical control (NC) machining.

Of late, computers are being increasingly used for forging die design applications. Initial developments concentrated on NC machining and computer-aided drafting. However, the current trend is to develop software for forging process analysis, including volume calculations, predicting stresses and forging loads for given die geometry, and in some cases simulating the metal flow [Oh 81]. These recent applications and developments of new methods for simulating forging operations indicate that CAD/CAM can significantly augment productivity and skill of the die designer. The next step in computerizing the forging die design process is to combine these islands of individual computer applications, towards an integrated CAD/CAM approach to hot forging, as shown in Figure 61.

As this computerization trend continues, it can be expected that application of CAD/CAM in the forging area will continue to increase, starting with drafting and NC machining, and progressing to design and analysis. As a result, the need for expensive die tryout trials on the shop floor can be expected to reduce, leading to improved material utilization and higher overall productivity gains.
Figure 61: Outline of an Integrated CAD/CAM Approach for Hot Forging. [Altan 83]