DEVELOPMENT OF A WINDOWS BASED COMPUTER-AIDED DIE DESIGN SYSTEM FOR DIE CASTING

BY

Woon Yong Khai
(B. Eng. (Hons), NUS)

A THESIS SUBMITTED
FOR THE DEGREE OF MASTER OF ENGINEERING

Department of Mechanical Engineering
NATIONAL UNIVERSITY OF SINGAPORE

2003

ACKNOWLEDGEMENTS

The two year long voyage through the rough seas of research has come to an end. It is my pleasure now to have the opportunity to express my gratitude for all who made this journey smooth and enjoyable.

First and foremost, I wish to express sincere appreciation to Associate Professor Lee Kim Seng from the manufacturing division – Department of Mechanical Engineering, NUS. His invaluable guidance, continuous inspiration and enthusiasm coupled with an integral view on research have made a deep impression on me. He manages to strike the perfect balance between providing direction and encouraging independence. He has given my career in engineering a purpose and a meaningful direction.

I am also grateful to Dr. Liu Xi Lin, Chief Technologist of Manusoft Technologies Private Limited, for sharing with me his wealth of knowledge in the area of Visual C++ programming. He was instrumental in assisting me to overcome my initial difficulties in programming and in me reaching a higher level of competence in the said programming language.

I also feel privileged to be surrounded by knowledgeable and friendly colleagues who helped me daily. Many thanks to my colleagues, Sun Yifeng, Du Xiaojun, Cao Jian, Saravanakumar Mohanraj, Atiqur Rahman and Low Leng Hwa Maria.

Financial assistance in the form of research scholarship from the National University of Singapore is also sincerely acknowledged. Finally, I am forever indebted to my parents and Janice for their understanding, endless patience and encouragement when it was most required.

SUMMARY

The design of die casting dies comprises of several stages and entails a large amount of time. Moreover, recurring modifications are required due to the complexity in achieving an acceptable initial die design. As a result, die design for die casting is usually time-consuming. The die casting industry stands to gain if proper application software are developed that integrate the different die design stages and allows the editing and customization of die design. Most recently, die design for die casting has been increasingly carried out wholly or partly in solids-based Computer-Aided Design (CAD), as it enhances the visualization of complex die design and assists users in design revisions. Hence it is imperative that the proposed computer-aided die design system for die casting is solids-based.

This thesis presents the research work of a computer-aided die design system for die casting. The proposed system consists of eight distinct modules. Through these modules, die designers are able to create a complete die casting die from a product part model. It is a user-friendly system that allows die designers to easily accomplish the task of die design. The approach undertaken in this research includes (a) standardization, (b) geometric and topological information extraction, (c) feature-based and constraint-based modeling, (d) table-driven design and (e) use of reference geometry and sketch entities.

A prototype system has been developed using this approach, and the implemented system is able to aid the automation of the die casting die design process. The practical goal of this research is fourfold: To develop a system that (1) integrates the different stages of die design process for die casting, (2) facilitates the editing and customization

of die casting die design during or after the design process, (3) automates or semi-automates several die casting die design process and (4) increases standardization by providing feature libraries of predefined standard die casting features that can be loaded conveniently to the die design project. A case study was performed using all the modules of the die design system for die casting and results had shown that the duration of die design process had been reduced significantly.

TABLE OF CONTENT

NONMENCLATURE

CAD	Computer-Aided Design
CAE	Computer-Aided Engineering
CBR	Case-Based Reasoning
3D	3-Dimensional
IMOLD	Intelligent Mould Design and Assembly System
CAM	Computer-Aided Manufacturing
PDM	Product Data Management
API	Application Programming Interface
OO	Object Oriented
OLE	Object Linking and Embedding
VBA	Visual Basic Application
COM	Common Object Model / Component Object Model
DLL	Dynamic Link Library
RAM	Random Access Memory
GPF	General Page Fault
OS	Operating System

IT	Information Technology
MFC	Microsoft Foundation Classes
B-Rep	Boundary Representation
CSG	Constructive Solid Geometry

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1 : INTRODUCTION

1.1 Background

Die development is an essential process connecting product design and manufacturing activities. Die development generally comprises three tasks: design, manufacturing and try-outs. The design of die casting dies comprises of several stages, which includes the design of ejector and cover cavity inserts, gating system, die base, ejection system and cooling system. Moreover, recurring modifications are required due to the complexities in achieving an acceptable initial die design. As a result, die design usually entails a large amount of time.

The relentless pursuits for lower lead times and reduced production cycles have lead numerous die designers to design an entire die casting die wholly or partly, using solids-based Computer-Aided Design (CAD) systems. In the past, designing die casting dies in solids can be cumbersome and resource intensive. Today, with the advent of technology, these problems are easily resolved. Solids-based CAD systems offer several advantages, like the ability to enhance the visualization of complex die casting part and assembly models, manage design revisions and improving the efficiency in production designing. There are many commercial CAD packages available, such as SolidWorks, Unigraphics, ProEngineer, etc, and most of them provide integrated features for surface modeling and solid modeling.

Some new commercial software products like dieCas and DiEdiFice, which automate the most repetitive aspects of die casting die design, had also been introduced. These commercial software products are discussed in the next chapter. However, the numbers

of commercial software products available are too few as compared to plastic injection mould. Moreover these commercial software products do not integrate the entire die casting die design process.

The die casting industry will greatly benefit if more comprehensive application software is developed that integrates the different die design stages and at the same time allow the editing of die design. The focus of this research is on the development of a die design system for die casting that runs not as stand-alone packages but within the environment of a specific CAD system. More details of the die casting process, the die design for die casting and its issues are discussed in subsequent sections.

1.2 Die Casting Process

The basic pressure die casting process consists of injecting molten metal under high pressure into a steel mold called a die. Die casting machines are typically rated in clamping tons equal to the amount of pressure they can exert on the die. Machine sizes range from 400 tons to 4000 tons. Regardless of their size, the only fundamental difference in die casting machines is the method used to inject molten metal into a die. Due to the differences in the melting temperatures of various die casting alloys, two methods of injecting the molten metal into the die cavities are used. These are referred to as hot chamber and cold chamber machines.

1.2.1 Hot Chamber Machines

Hot chamber or plunger machines are used mainly for metals of low melting point and high fluidity such as tin, zinc, and lead that tend not to alloy easily with steel at their melt temperatures. Development in technology had enabled this process to be used for

some magnesium alloys. The hot chamber process is a preferred die casting method due to its high rate of productivity.

In this process, the plunger and cylinder, which constitute the injection mechanism, are submerged in the molten metal in the crucible. The operating sequence for the hot chamber die casting process as illustrated in Figure 1.1 is as follows:

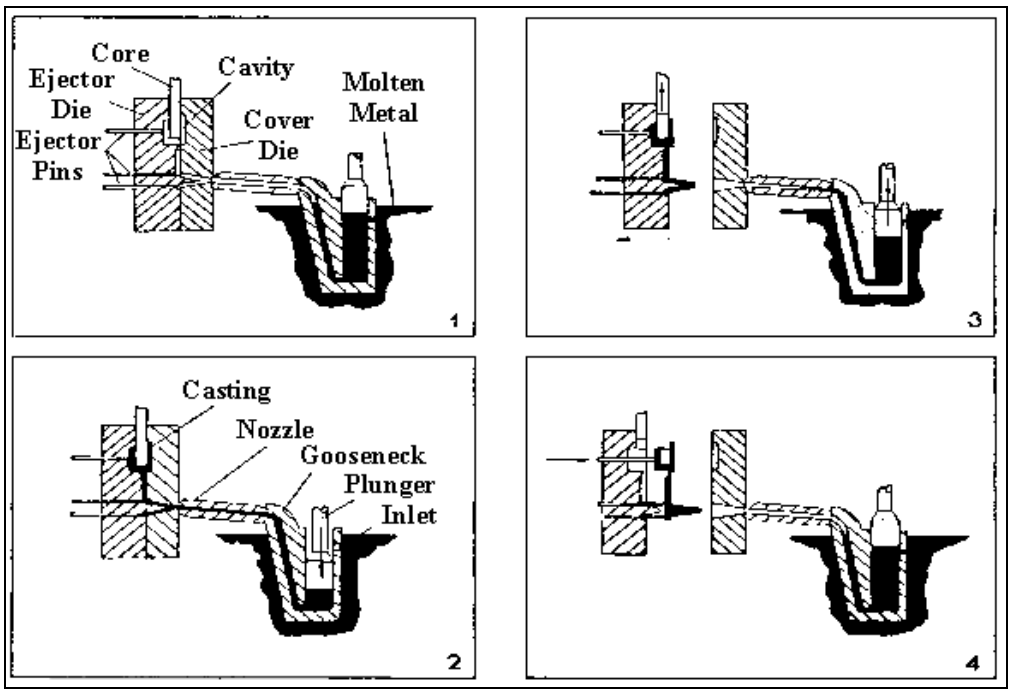


Figure 1.1: Hot Chamber Process [1]

1. The die is closed and the piston rises, opening the inlet and allowing molten metal to fill the gooseneck cylinder.

2. The plunger moves down and seals the inlet pushing the molten metal through the gooseneck passage and nozzle into the die cavity, where it is held under pressure until it solidifies.

3. The die opens and the cores, if any, retract. The casting remains in the die. The plunger returns, allowing residual molten metal to flow back through the nozzle and gooseneck.

4. Ejector pins push casting out of the ejector die. The sequence from step 1 is then repeated.

1.2.2 Cold Chamber Machines

In a cold chamber process as illustrated in Figure 1.2, the molten metal is ladled into the cold chamber for each shot. The shot chamber is not heated -- hence the term cold chamber. Cold chamber machines minimize contact between the alloy to be cast and steel machine parts, thus allowing the processing of metals such as Aluminium, Copper and their alloys at higher temperature. Its primary use is for aluminum, brass, and larger magnesium die castings.

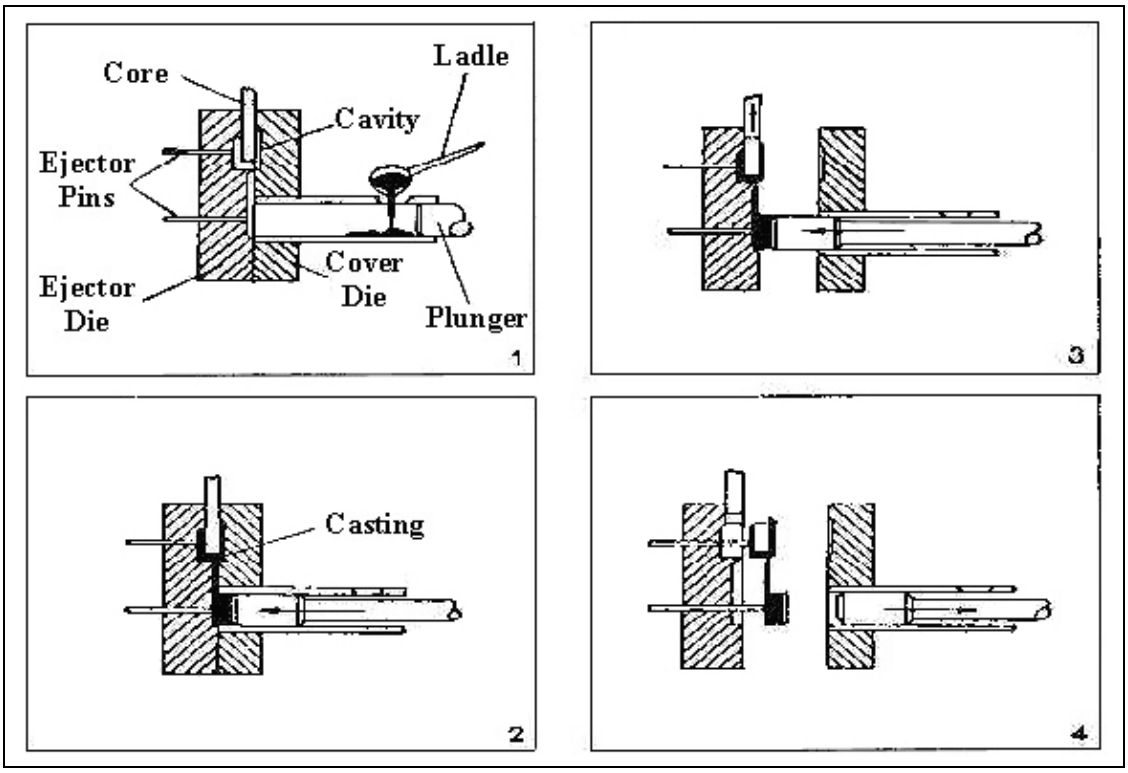


Figure 1.2: Cold Chamber Process [1]

The operating sequence for the cold chamber die casting process is as follows:

1. The die is closed and the molten metal is ladled into the shot sleeve.

2. The plunger pushes the molten metal into the die cavity where it is held under high pressure until it solidifies.

3. The die opens and the plunger advances, to ensure that the casting remains in the ejector die and to push the solidified slug from the cylinder. Any cores that are present will retract.

4. The ejector pins push the casting off from the die and the plunger returns to its original position.

1.2.3 Die Base

Die bases are made of alloy tool steels in at least two sections, the cover die half, and the movable ejector die half, to permit removal of castings. The former is attached to the molten injection system of the machine, and the latter is connected to the ejection mechanism. Die bases and their components normally come in standard sizes provided by die base vendors. The cover die half usually contains sprue holes to allow molten metal to enter the die and fill the cavity. The ejector die half generally contains the runners, gates and overflows that route molten metal to the cavity. It also contains ejector pins to help remove the casting. Modern die bases also may have moveable slides, cores or other sections to produce holes, threads and other desired shapes in the casting. Die bases also include guide pins to align the two die halves and locking pins to secure the two halves. Dies are usually cooled by circulating water or oil through various passageways in the die base.

When the die casting machine closes, the two die halves are locked and held together by the machine's hydraulic pressure. The surface where the ejector and fixed halves of the die meet and lock is referred to as the die parting line. The total projected surface

area of the part being cast, measured at the die parting line, and the pressure required of the machine to inject metal into the die cavity governs the clamping force of the machine.

1.3 Die casting die design process

Die casting die design consists of several stages and generally speaking, die design still depends on experience and know-how of experts who have the skills in die manufacture. As a result, die casting die design is often time-consuming.

The die casting die design process begins from the product model as shown in Figure 1.3 and it can be broken into nine stages as outlined below.

Stage 1: The reconstruction of the product geometry to account for material shrinkage during the processing operation. The factors determining material shrinkage include material properties, part thickness, melt temperature, die temperature and injection pressure.

Stage 2: The determination of parting lines. This is a crucial part of die design and many areas must be examined. These areas comprise of the flow pattern analysis, location of gating features, existence of slides and cores, parting line aesthetics and finishing operations.

Stage 3: The detection of undercuts and design of slides. In the removal of a cast model from the die, it is essential to note that there are no undercut sections that will lock the cast model in the die. Undercuts should be avoided but when they are necessary, movable cores or slides must be used.

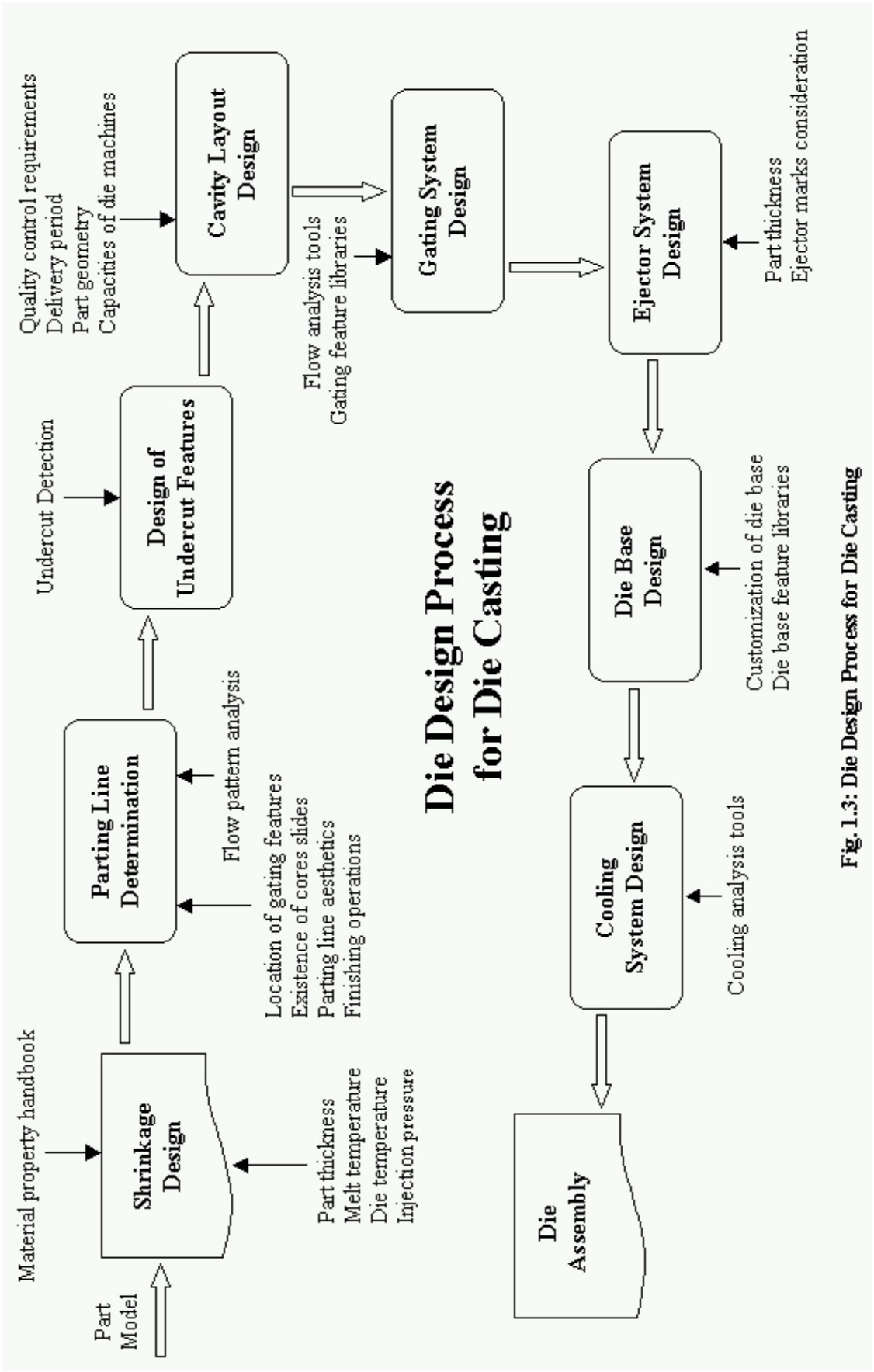


Fig. 1.3: Die Design Process for Die Casting

Stage 4: The design of the cavity layout. A single or multiple-cavity die will depend on the quality control requirements, costs, delivery period, part geometry and capacity of die machines.

Stage 5: The design of the gating system. The gating system of a die casting die consists of a series of passages through which the molten metal can flow into the die and then through the interior of the die to fill the cavity. The gating features are gate, runner, overflow and vent. The design of a proper gating system in die casting dies is very important and their control is a critical part of the die casting process. The design process encompasses several steps and involves complex computational work.

Stage 6: A selection of a suitable die base to house the die design system. Standard die bases provided by vendors are generally used. There may be small alterations like the locations of guide pin, inclusion of additional plates, etc.

Stage 7: The design of the ejector system. The placement of ejector pins is important in a successful ejection of cast model. Care must be taken to prevent ejector marks on the cast model. Ejector pins are normally fixed to the ejector plate of the die base, while the end of the ejector pins are flushed with the parting surface.

Stage 8: The design of the cooling system. The provision of suitable and adequate cooling arrangements requires special attention in die design. The cooling system should ensure rapid and uniform cooling of the die. Cooling analysis is often conducted to aid the cooling system design.

Stage 9: The assembly of the cavity layout, gating system, core slides, ejection system, cooling system into the die base. Try-outs will be conducted on the final assembly.

1.4 Research Objectives

The motivation behind this research is derived from the following observations:

a) The traditional design software products of die casting dies do not fully integrate the different stages of the die design.

b) Recurring modifications to die design are generally needed but yet cannot be easily made. As a result, die design is usually time-consuming and costly with respect to resources.

c) Part models for die casting are increasingly constructed using solids-based CAD software, but there are limited die design applications software for die casting that support solids-based CAD, especially when compared with plastic injection mould design.

Thus, a die casting die design system had been proposed and a prototype developed in this research does so as to provide solutions for the problems stated above. The objectives of the proposed system are listed as follows:

a) To be solids-based and to integrate the different stages of the die casting die design process.

b) To be equipped with the ability to update die casting die design during or after the course of the design process, based on changes to cast part model. In this way concurrent die development and re-use of existing die designs can be achieved.

c) To maintain the same look and feel to traditional commercial solids-based CAD software. This will be beneficial to the die designers as many die designers are

already familiar with the various commercial solids-based CAD software products.

1.5 Layout of Thesis

This thesis contains seven chapters and is outlined as follows:

In Chapter 1, the background of the research on the thesis is first presented, followed by the motivations and objectives of the research.

Previous works related to the use of computer-aided applications in die casting die design are reviewed and the related techniques used in the research are examined in Chapter 2.

Chapter 3 describes the developmental platform and tools used for the prototype die design system.

The design methodologies used in the die casting die design system are then presented in Chapter 4.

In Chapter 5, the system requirements are first discussed. The architecture of the computer-aided die casting die design system is then presented and the individual modules described.

The implementation of the prototype die casting die design system and some case studies are presented in Chapter 6.

Finally Chapter 7 provides a conclusion of the whole research work with an outline of the contributions. The limitations of the die casting die design system and the recommended solutions are also discussed.

CHAPTER 2 : LITERATURE REVIEW

In this chapter, previous works related to the use of computer-aided applications in die casting die design are reviewed. Some commercial die casting die design and plastic injection mould design software are also discussed. Finally, some of the related techniques, which include (a) parametric design, (b) feature-based modeling and (c) constraint-based modeling, are examined.

2.1 Background

Conventional die design has been carried out by a designer who has many years of experience and who follows a process of trial and error for designing the product and die to produce the final casting. Such processes cause the lead time to extend and increase cost. As a result, significant amount of research and development work has been conducted over the years in order to optimize the die casting process and the quality of the castings. The research and development work includes numerical simulation, knowledge-based methods and CAD/CAE design systems. Some of these researches had been commercialized. The following sections discuss these researches in details.

2.2 Numerical Simulation

The arrangement and shape of the gating system (gate, runner, sprue, pouring basin, overflow, airvent, etc) are the most important factors in the design of die casting dies. However, the design of the gating system in die casting often involves trial and error. Numerical simulation takes the guesswork out of die design by optimising the gating

system and drastically reducing the time it takes to produce production-quality dies. Hence it is considered as the most cost-effective way in optimization of the gating system [2,3]. There are currently numerous commercial numerical simulation software products in the market. These software products assist users in virtual testing, optimization through finite element analysis, thermal analysis and flow analysis.

A number of papers had also been published in the fields of numerical simulations. Tai et al. [4] explored a computer integrated system application in the design of a die casting mold, in order to enable accurate and fast determination of the optimal position for the material injection gate. Finite elements are used to find the deformation of various sizes of thin shell piece, after which a deformation learning prediction is made with an abductive network. A simulation position annealing (SA) optimization algorithm with a performance index is then applied to the neural network in order to search for the optimal position of the injection gate. Tai et al. [5, 6] also studied the optimization accuracy of a die casting product part. A model of die casting had been built using abductive network. The abductive network is composed of a number of functional nodes. Once the die casting parameters are given, this framework can predict the die casting performance accurately.

Shamsuddin et al. [7] used network analysis method with the aid of a program written in FORTRAN language to conduct a flow analysis along the gating system. Hu [8] utilized the commercial software CASTFLOW^{TM} and MAGMASOFT$^{®}$ to design and optimize the gating system for the die casting of thin-walled magnesium telecommunication parts.

CASTFLOW [9] was developed by the CSIRO Division of Manufacturing Technology and commercialised in 1991 by Castec Australia Pty Ltd. The program is an integrated

approach to die design, which can predict metal flow and temperatures inside a die at the design phase. This permits complex testing of machine performance and die design without the expensive trial and error processes of the past. However this software application is primarily used for flow analysis and limited to the design of runner and gating system. MAGMASOFT® [10] is a comprehensive simulation tool that helps users to avoid gating and feeding problems, predict casting quality, aids permanent mold design and reduces fettling costs.

2.3 Knowledge-based Methods

2.3.1 P-Q^2 technique

The first method is the P-Q^2 technique. The P-Q^2 technique, which is based on knowing the pumping rate capability of the die casting machine, can be used to predict what will occur when a die is put on the machine. Since different machines have different pumping characteristics, the gating system must be matched to the machine characteristics to achieve the correct pressure and flow rate [11].

Wu et al. [12] developed a prototype design of a gating system for a diecasting using P-Q^2 technique and feature-based parametric design. Algorithms based on the *P-Q^2* technique are proposed to carry out filling analysis and predict the process parameters. The dimensional parameters of the gating system are then determined based on the predicted result. Zhang et al. [13] also uses P-Q^2 technique in his CAD/CAE system. They designed the runner, the feed and the gate according to the gate area, flow rate, filling time and gate velocity provided by the P-Q^2 technique.

2.3.2 Case-Based Reasoning (CBR)

Another method is case-based reasoning (CBR). A CBR system adapts the solution of a previously solved case to build a solution for a new problem [14]. Lee and Luo [15] explored how CBR can be used to improve the efficiency of die casting die design. Their work include the development of a case-based reasoning environment for die casting die design, so that the system can self-learn from the previous consultation itself and become more efficient through an effective case representation and adaptation methodology.

2.3.3 Taguchi's techniques

Taguchi's techniques for quality engineering had also been used on die casting. Syros [16] studied the setting of various significant process parameters of the die casting method of $AlSi_9Cu_{13}$ aluminium alloy. The effects of the selected process parameters on the casting density and the subsequent optimal settings of the parameters have been accomplished using Taguchi's method.

2.3.4 Commercial Knowledge-Based Software

There is also a commercial software called DC-CALC [17], which enables users to calculate all the vital parameters in a die casting die within a short time. This software can quickly analyse many alternative die-machine combinations like the determination of feasibility when the number of cavities changes, show the effect on the gate velocity and cavity fill time when the depth of the gate changes, show the effect on the surface quality when the die temperature changes and display a new P-Q diagram upon any changes.

2.4 CAD/CAE Design Systems

To assist and expedite the design and manufacturing of die casting parts/dies, CAD and Computer-Aided Engineering (CAE) design systems have been used in the die casting industry in recent years. As a result, many researchers had proposed developing die casting die design system on commercial CAD software. This section discusses the research works on the automated or semi-automated design of individual die elements, entire die design system for die casting, and the integration of CAD and CAE systems. A comparison with plastic injection mould design is also included.

2.4.1 Automated or semi-automated design of individual die elements

Most of the research works concentrated on the automated or semi-automated design of individual die elements. For example, Wu et al. [12] developed a prototype design of a gating system for die casting using P-Q^2 technique and feature-based parametric design. This system is inbuilt with a gating library that contains pre-defined user-defined gating features. These gating features can be retrieved from the library and applied to the gating part with the desired parameters and locations during the design process. The prototype for the gating system of die casting dies was implemented on Unigraphics CAD system.

Lu [18] dealt with the incorporation of automatic/semi-automatic geometric modifications with drafts and rounds/fillets in a CAD system for die cast part/die designs to remove most of the detail work and to maintain consistency. The goal of the paper is to demonstrate the advantages of using a CAD system with geometric reasoning mechanisms to expedite or even automate the drafting and rounding/filleting procedure. The implementation of the proposed approach uses Pro/ENGINEER as a platform.

There is also a commercial die design application called DiEdiFice [19] that focuses on the gating system of die design. It is a 3D-design application for pressure die-casting and it is used to design a precise and efficient gating and runner system. DiEdifice performs gating system analysis for: Reynolds number, PQ^2 check, air entrapment, yield and fettling. It also possesses computational capabilities for: total net cooling load, total effective cooling length, collision check, interference check, total ejection load and the number of ejector pins. However DiEdiFice does not assist in the creation of ejector and cover cavity inserts, ejector pins and cooling lines.

An example of a commercial CAE system used in die casting industry is dieCAS [20]. dieCAS is a workstation-based CAE software product for process modeling and analysis of die casting and related processes. It is currently maintained and marketed by Technalysis, Inc. The analysis capabilities of dieCAS include cavity fill, heat transfer & solidification, casting distortion and die distortion.

2.4.2 Comprehensive die design system for die casting

There is limited published work in the areas with regard to comprehensive die design system for die casting. Choi et al. [21] developed a die design system based on the AutoCAD platform. The proposed die design system uses 3D geometry handling and achieve automation through integrating the generation process and the technology of process planning. In addition, specific rules and equations for the runner-gate system have been presented to avoid too much trial and error with expensive equipment. The system focuses on the runner-gate system and had been tested for simple shapes like Cap-shape with single impression die that have no undercuts.

2.4.3 Integration of CAD and CAE systems

In recent years, attention had also been paid to the integration of CAD and CAE systems. The integration of CAD and CAE systems is essential for the quick development of a low cost die, as well as to facilitate accuracy in simulation. Both Zhang et al. [13] and Tai et al. [5] developed a CAD/CAE system for die casting. The idea was to determine the die geometry and process parameters by using CAD and then CAE package to optimize the process design based on the simulation analysis.

2.4.4 Comparison with plastic injection moulding

Unlike die casting, several specialized software packages had been introduced for semi-automated injection mould design. One example of such a software package is IMOLD (Intelligent Mould Design and Assembly System) [22]. Although there are some similarities between die casting dies and injection moulds, the design of the former gating system is more complex [12]. Hence IMOLD and other similar specialized software packages for semi-automated injection mould design are not suitable for die design.

2.5 Parametric Design

Parametric design implies the use of parameters to define a form when what is actually in play is the use of relations [23]. Parametric design deals with variable dimensions as control parameters, and it is an efficient tool for creating models based on parameters. Parametric design not only increases the design efficiency, but also makes the updates and modifications of existing designs easier and faster, since these can be achieved by changing the parameters of the parametric model [24, 25]. Parametric design plays an important part in product modeling as it encourages standardization of product.

Tu et al [26] deployed parametric design for gating system of investment-casting mould. They pre-constructed parameterised solid models (primitives) of gating geometries and stored them in a directory (gating library). When the CAD model of the component is retrieved from the original equipment manufacturers (OEMs), the gating primitives are selected and retrieved from the gating library, specifying desired dimensions and locations. These gating pieces are then joined to the solid model of the component using a Boolean operation. Consequently, this procedure can efficiently reduce the solid-model construction time required to modify OEM model into a casting model when standard gating features are used.

2.6 Feature-Based Modeling

Feature-based modeling constructs the model of the product directly on the basis of features. Feature-based models contain not just basic geometric and topological data but also high-level information, which allows die designers to add relatively complex shapes to their designs. The high-level information comprises geometry, functionality, machining and process planning, etc, and these high level information can be extracted from the model for the purpose of calculation and evaluation. As a result, feature-based design has been extensively used in automated modeling and process planning.

Chen and Wei [27] proposed a feature-based design framework for net shape manufacturing. The proposed framework focuses on die casting and injection moulding processes, but it is general enough for other net shape processes by customising process specific features and encoding process design rules.

Lee et al [28] presented an automated process planning system for the manufacture of lifters of the injection mould. This system classifies the lifter types and extracts the

feature parameters from the model, and then the system will select a standard process plan template and correspondent machines, cutters, fixtures and cutting parameters for each process in the plan template based on the extracted information and available machining resources.

2.7 Constraint-Based Modeling

Constraint-based modeling involves constraints that are used to create a set of rules that control how changes can be made to group of geometric elements. Shimizu et al. [29] found that geometric constraints for representing 3D shapes can be categorized into the following three types: topological constraint, structural constraint and dimensional constraint. A topological constraint defines the topology of a primitive solid itself by specifying the connection between the geometric elements. A structural constraint gives the primitive solid the character of a particular feature. Dimensional constraints define the size and location of a feature.

Constraint-based modeling in a parametric system captured and solved constraint equations sequentially. As it cannot solve coupled equations, a predictable model will result. Constraint-based modeling allows the CAD system to capture the die designer's intent and relationships can be based on this intention. Furthermore, constraints-based modeling also allows changes to be propagated through the model quickly. Fudos and Hoffmann [30] constructed conic blending arcs from constraints, using a unified rational parametric representation that combines the separate cases of blending parallel and non-parallel edges.

Anderl and Mendgen [31] gave a very detailed study on modeling with constraints. They found that it is a modern approach to product modeling. Together with the

technique of feature-based modeling, constraint-based modeling has widely affected the development of new CAD system. This paper closes with a summary of the advantages and risks of designing with constraints, showing typical application fields of this modeling technique and discussing some open issues.

2.8 Project direction relative to literature review

This research focuses on implementing a windows-based computer-aided die design system for die casting. Literature review has shown that most of the research and commercially available software products on die design system for die casting concentrated on the gating system of die design or on the automated or semi-automated design of individual die elements. The only comprehensive die design system in the literature survey has been found lacking as it only caters to simple die design. The aim of the proposed die design system for die casting is to integrate the various stages of die casting die design comprising of cast design, die design, die layout, die base design, cooling system deign and ejector system design.

Recent research has proposed the integration of CAD, CAE and Computer-Aided Manufacturing (CAM) into one system as essential for the quick development of a low cost die, as well as to facilitate accuracy in simulation. However the development of a stand-alone CAD/CAE/CAM system will take huge volume of developmental work and detailed assessment of the overall integration structure. Hence this research proposes to develop a prototype die casting die design system on a commercial solids-based CAD platform.

Since die casting die design has been increasingly carried out on solids-based CAD system, developing a design system on the CAD system platform will benefit die

designer by keeping the same look and feel. Besides, with the increasing demands of capabilities to meet the requirements of complicated design activities, CAD technology had been greatly improved. For example, the commercial SolidWorks CAD software not only offers excellent modeling tools, it is also fully integrated with other software products that are also important to die casting die design. The fully integrated software products include finite element analysis software (ABAQUS), design analysis software (COSMOSWorks), CAM solution software (CAMWorks), PDM software (SmarTeam), collaborative aid (eDrawings) and fluid flow and thermal analysis software (COSMOSFloWorks).

Therefore building the die casting design system on an existing commercial CAD system also allows the die designer to exploit the diverse information resources relevant to the design decisions. For these reasons, SolidWorks was chosen for the required task in this research. In addition, since SolidWorks supports parametric design, several parameterized models of die casting features like gates, runners, overflows, ejector pins, etc can be created using feature-based and constraints-based modeling to increase standardization.

CHAPTER 3 : DEVELOPMENTAL PLATFORM & TOOL

This chapter describes the developmental platform and tools used for the prototype die design system. The developmental platform selected is the commercial SolidWorks 2001 CAD system. Through SolidWorks Application Programming Interface (API), algorithms are derived to expedite the die casting die design process. As the SolidWorks API uses an Object-Oriented approach, the concept of Object-Oriented (OO) programming will also be briefly stated. The tool used for the interface design for the die casting die design system, Microsoft Visual C++ is also discussed.

3.1 SolidWorks 2001 CAD System

SolidWorks 2001 [32] is a mid-range feature-based parametric solid modeler with surfacing capabilities that allows users to create parts, assemblies, and drawings. There are several factors behind the selection of SolidWorks 2001 as the platform for the proposed die casting die design system and they are as follows:

1. It is Windows-based: Die design and other related applications software have been increasingly carried out on Windows-based operating systems. Running on non-Windows-based operating systems will result in the dilemma of dealing with two or more different systems for the die designers. Familiar Windows functions like drag-and-drop, point-and-click, and cut-and-paste allow users to become productive in hours and proficient within weeks.

2. It is fully integrated with other CAM and CAE application software like ABAQUS, COSMOSWorks, CAMWorks, SmarTeam, eDrawings and COSMOSFloWorks.

3. It is a feature-based solid modeler: It supports feature-based design, which is essential to the proposed die casting die design system.

4. It is equipped with its own Application Programming Interface (API): API simplifies programming in Visual C++.

3.2 SolidWorks Application Programming Interface

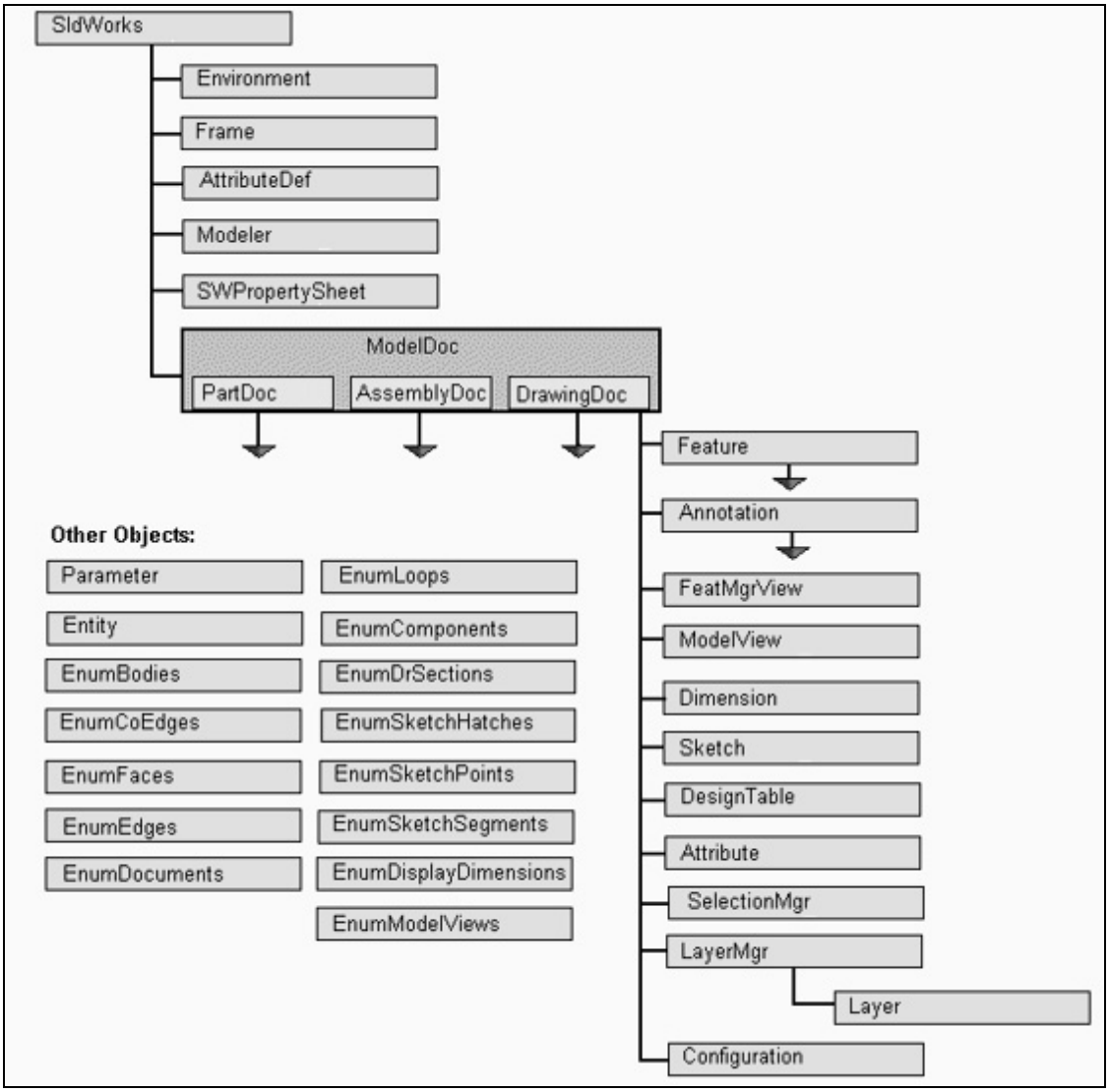


Figure 3.1: SolidWorks Application Programming Interface objects

The SolidWorks API [33] is an Object Linking and Embedding (OLE) programming interface to SolidWorks. The API contains hundreds of functions that can be called from Visual Basic, Visual Basic Application (VBA) (Excel, Access, etc.), C, C++, or SolidWorks macro files. These functions provide the programmer with direct access to SolidWorks functionality such as creating a line, extruding a boss, or verifying the parameters of a surface.

The SolidWorks API interface uses an Object-Oriented approach and it is consistent with the Windows based approach of SolidWorks. By way of reference, the entire SolidWorks user interface is based on Microsoft Foundation Classes.

SolidWorks exposes its API functionality through standard COM objects. For OLE automation, the API is exposed using IDispatch. The Dispatch interface will accept and return arguments as Variants and IDispatch pointers so languages such as Basic can handle them. The Dispatch interface should be used by all Visual Basic, VBA, or VC++ executable files (exe) implementations.

A COM implementation gives your application direct access to the underlying objects or arrays, and subsequently, increased performance. COM implementations will provide slightly more functionality, with operations such as enumeration, and will also return an HRESULT value for each API function call. The COM interface is currently only available to VC++ add-in dynamic link library (DLL) implementations. The COM interface is recommended for all VC++ add-in DLL projects.

In view of this, the proposed die casting die design system will be implemented as an add-on to SolidWorks.

3.3 Visual C++ version 6.0

Microsoft Visual C++ 6.0 [34] is the industry standard for professional C++ Windows programming. With the increasing complexity of the Windows operating system, common design elements must be used to assure platform compatibility in applications.

Visual C++ 6.0 has made creating dialog boxes and property sheets resources very easy and intuitive. The steps involved with creating a dialog resource are minimal, but the code added can be simple or complex. In any application, a graphical user interface must enable users to enter meaningful data easily. In this aspect, Visual C++ 6.0 allows developers to create common controls of Windows effortlessly. These common controls come in different shapes and styles, such as edit boxes, buttons, list boxes, combo boxes, etc.

As the proposed die casting die design system is not a stand-alone but operating on commercial CAD software, the graphical user interface should consists of modeless dialog boxes so as to facilitate die designers to use the parent CAD software. A modeless dialog box looks like a document window without a size box, zoom box, or scroll bars. The user can move a modeless dialog box, make it inactive and active again, collapse or close it like any document window. Modeless dialog boxes provide the most flexibility for users, allowing them to do any task at any time or in any order.

3.4 DLL Files

A dynamic link library (DLL) [35] is a collection of small programs, which can be called upon when needed by the executable program (exe) that is running. The DLL

lets the executable communicate with a specific device such as a printer or may contain source code to do particular functions.

The advantage of DLL files is that, because they do not get loaded into random access memory (RAM) together with the main program, space is saved in RAM. When and if a DLL file is called, then it is loaded. All in all a DLL is an executable file that cannot run on its own, it can only run from inside an executable file. To do this an executable needs to declare the DLL function, then when needed the call is made with the required parameters. If a call or a declaration is made incorrectly a General Page Fault (GPF) may occur. A call to a DLL of a different version might require more or less parameters or the call may not exist. If a DLL is the wrong version for your Operating System (OS) or for a program that you have installed it will cause a GPF or lockup your machine. Generally a file that is older than your OS and is available in the Windows cabinet files is the wrong version.

3.5 Object-Oriented Approach

The concepts of Object-Oriented (OO) have been around for over forty years. OO's popularity and sophistication has increased in the past several years as businesses began to incorporate more client-server models to run their businesses and are using Information Technology (IT) as a business tool. The difference between traditional Procedural programming and Object-Oriented programming are shown in Table 3.1.

The main advantage of OO programming is its ease of modification as objects can easily be modified and added to a system. OO programming allows for more complicated and flexible interactions than procedural programming. OO's concepts are also simpler for non-technical personnel to understand because it appeals to natural

human cognition patterns. OO approach can shorten development time since many objects are standard across systems and can be reused. The OO language used to build the prototype die casting die design system is C++. C++ include features such as "class", "instance", "inheritance", and "polymorphism" that increase the power and flexibility of an object.

Table 3.1: Difference between Procedural and Object-Oriented

	Procedural	**Object-Oriented**
Handling of code and data	Code and data are kept separate. Changes made to any of the code sets and data sets can cause problems through out the system.	Code and data are merged into one indivisible item – an object The information within an object is encapsulated from the rest of the system.
Structure of system	If a function is used multiple times in a system, it is often simply cut and pasted into each program. Plenty of repeated functions.	A system is composed of multiple objects. When one object needs information from another object, a request (message) is sent asking for specific information. No repeated functions.

3.6 Microsoft Foundation Classes

Microsoft Foundation Classes (MFC) [36] is a C++ code library that simplifies the writing of Windows applications. Using standard C++ programming, one needs 80 or 90 lines of code just to get an empty window on screen. A full-fledged Windows application can be immense beyond belief. Using MFC however, one can get your first window onscreen with only a few lines of code. MFC simplified the process of writing Windows applications by hiding many of the details inside custom window classes.

MFC includes many classes that can be used to write Windows application more quickly and easily. These classes represent objects from which a Windows application is created – objects such as windows, dialog boxes, menu bars, window controls, and

many more, including some special objects such as status bars and control bars. In addition, MFC provides many general-purpose classes for handling things like strings, arrays, and linked lists.

Specifically, MFC provides the following main categories of classes: applications, Windows, menus, dialog boxes, documents and views, controls, graphics, archives and file, database, various support classes. MFC features a host of other classes that do everything from handling linked lists to enabling users to create OLE applications.

CHAPTER 4 : DESIGN METHODOLOGY

In this chapter, the design methodologies used in the die casting die design system are presented. The methodologies are outlined as follows:

- Standardization
- Geometric and topological information extraction
- Feature-based and constraint-based modeling
- Table-driven design for assembly
- Use of reference geometry and sketch entities

4.1 Standardization

The die casting design process involves several standard die casting components. By predefining these die casting components and stored them in feature libraries, the die casting design process will acquire increased standardization. A die casting feature is a feature-based parametric part or assembly model. Several techniques are used to create the various solid models of the die cast features, depending on their complexities. The die casting feature libraries constructed for the die design system for die casting as shown in Figure 4.1, are made up of individual feature libraries of core slides, gating features, ejector pins, die bases, cooling features and many standard die casting components. Additional die casting features can be added to the individual feature libraries whenever necessary.

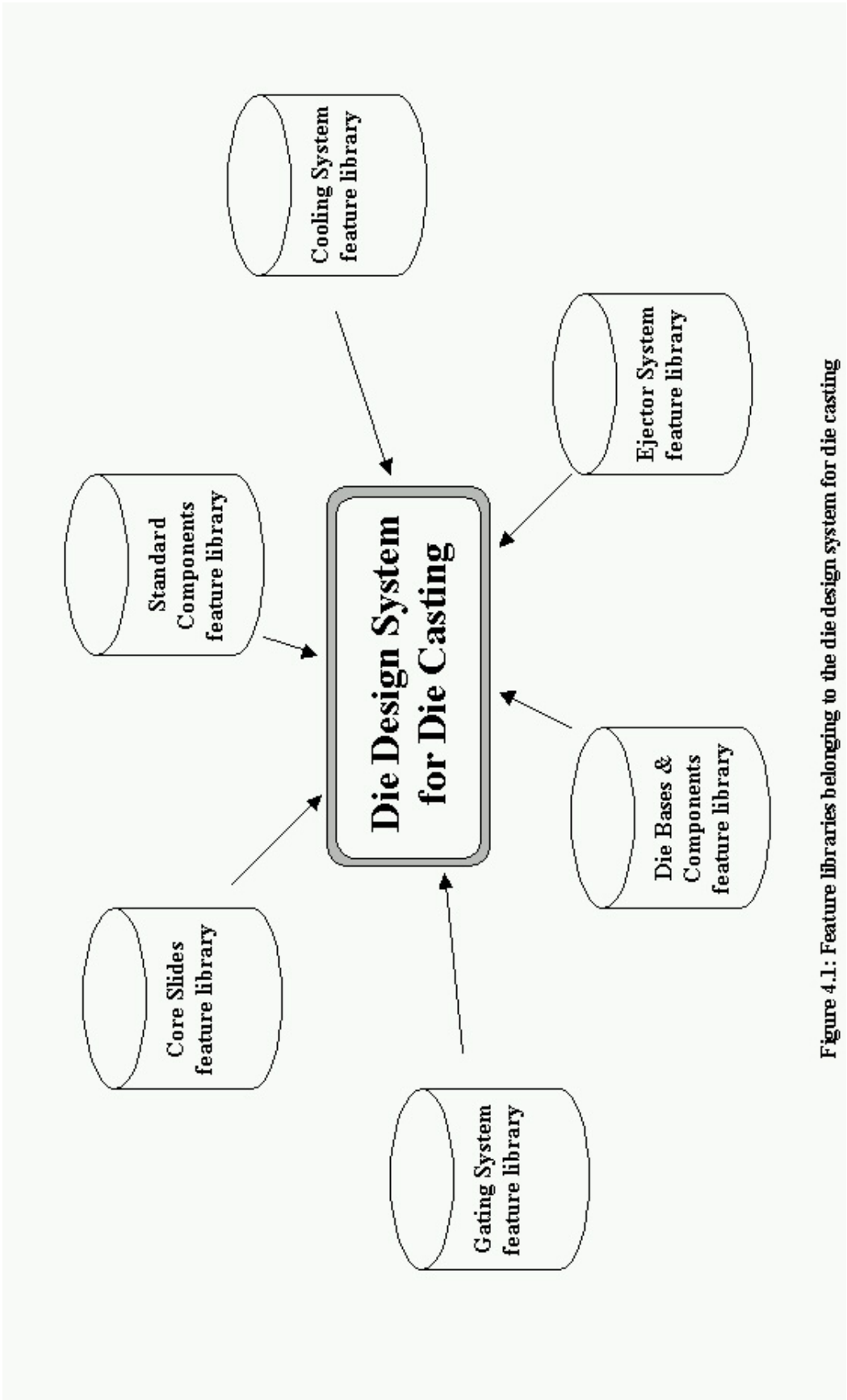


Figure 4.1: Feature libraries belonging to the die design system for die casting

4.2 Geometric and Topological information extraction

As SolidWorks is a feature modeler, base model is defined by the execution of its features. However, from the SolidWorks API point of view, part model created by the execution of the features is a boundary representation (B-Rep) model. Boundary representation is a commonly used solid modeling representation scheme. B-Rep is a more explicit representation than Constructive Solid Geometry (CSG) and it is a more natural representation for display and for applications such as NC path generation, because up-to-date edge and face information are always readily available. However, the data structure of B-rep model is complex and requires a large memory space. In this scheme solids are represented in terms of individual surfaces, edges, and vertices. B-Rep models can be divided into two parts – topology and geometry.

Topology records the connectivity of the faces, edges and vertices by means of pointers in the data structure while geometry defines the exact shape and position of each of the edges, faces and vertices. It is important to understand the B-Rep model because it allows the die casting die design system to extract geometric and topological information from the model.

Examples of some of the information that can be extracted are as follows:

- Finding the list of edges adjacent to a given edge.
- Finding the list of faces adjacent to a given face.
- Checking whether a face has a hole or a boss in it?

This information is essential to automate some of the die casting die design process like the search for the parting line, the search for the parting face and the patching of

holes. The algorithms for 'Parting Line Search', 'Parting Face Search' and 'Hole Patching' will be discussed in the following sections.

4.2.1 Algorithm for 'Parting Line Search'

"Parting Line Search" is the technique for searching the edges of the part model that determines the joint between the cover and ejector cavity inserts. For a complex die casting part model, the parting line may contains more than a hundred edges. Even though this algorithm does not provide an automatic detection of parting line, it allows the die designer to select the parting line in a fast and intuitive way.

The algorithm for the function *CompareEdge* is as follows:

CompareEdge(edge):

```
for (int m=0; m<total_edgeCount; m++)
{
        edge->IGetStartVertex(&s_vtx);
        edge->IGetEndVertex(&e_vtx);
        s_vtx->IGetPoint(s_ptr); //Get Coordinates of end point 1
        e_vtx->IGetPoint(e_ptr); //Get Coordinates of end point 2

        result1 = FALSE;
        result2 = FALSE;
        result3 = FALSE;
        result4 = FALSE;

        if (nEdgeList[m] != NULL)
        {
            nEdgeList[m]->IGetStartVertex(&s_vtx1);
            nEdgeList[m]->IGetEndVertex(&e_vtx1);

            if (s_vtx1 != NULL && e_vtx1 != NULL)
            {
                s_vtx1->IGetPoint(s1_ptr);
                e_vtx1->IGetPoint(e1_ptr);

                if (s_pt[0] = s_pt1[0] && s_pt[1] = s_pt1[1] && s_pt[2] =  s_pt1[2])
                    result1 = TRUE;

                if (e_pt[0] = e_pt1[0] && e_pt[1] = e_pt1[1] && e_pt[2] = e_pt1[2])
                    result2 = TRUE;
```

```
                if (s_pt[0] = e_pt1[0] && s_pt[1] = e_pt1[1] && s_pt[2] = e_pt1[2])
                    result3 = TRUE;

                if (e_pt[0] = s_pt1[0] && e_pt[1] = s_pt1[1] && e_pt[2] = s_pt1[2])
                    result4 = TRUE;

                if ( result1 = TRUE || result2 = TRUE || result3 = TRUE || result4
                =TRUE )
                {
                    //Select Next Edge
                    nEdgeList[m]->QueryInterface(IID_IEntity, (LPVOID *)&swEntity)

                    swEntity->SelectByMark(TRUE, 1, &chk)

                    CompareEdge(edge);
                }
            }
        }
}
```

The ‘Parting Line Search’ algorithm can be divided into 6 steps.

Step 1: The die designer is required to specify a list of ‘Boundary Faces’. ‘Boundary Faces’ are faces of the product model that contains the edges that define the ‘Parting Line’. Figure 4.2a shows the part model that only needs one boundary face.

Step 2: Then the die designer needs to select any of the single edge that makes up the ‘Parting Line’. Figure 4.2b displayed the selected edge that will be used to generate the ‘Parting Line’. This edge will be termed the ‘First Edge’.

Step 3: The algorithm will then store the common edges belonging to all the boundary faces specified by the user into an array. The product model shown in Figure 4.2a has only one boundary face. Hence there are no common edges and all the edges belonging to the single boundary face will be stored. Figure 4.3a shows all the edges belonging to the boundary face that are stored.

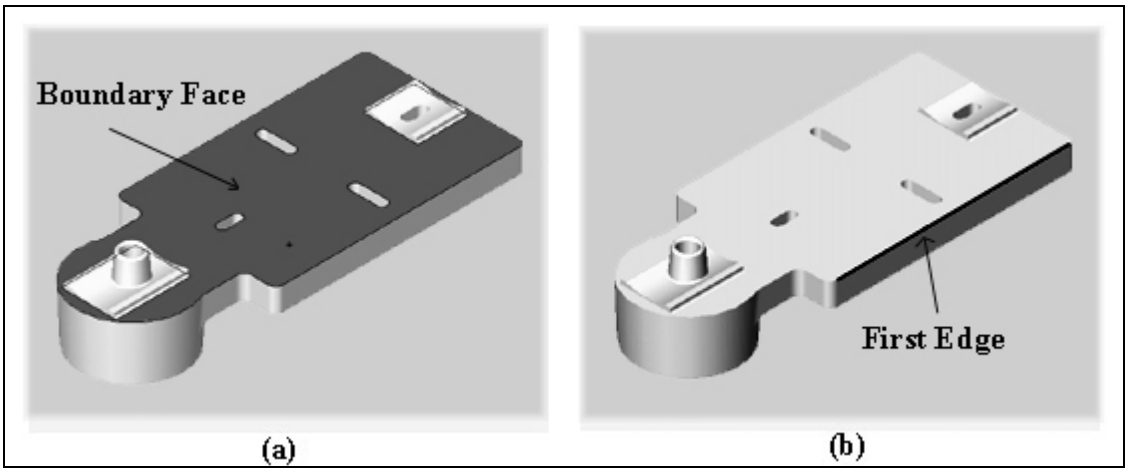


Figure 4.2: (a) 1 Boundary face (b) The 'First Edge'

Step 4: The next procedure is to obtain the coordinates of the two end points from the 'First Edge'. The algorithm will follow on to compare the two end points of the 'First Edge' with the two end points of all the common edges stored earlier, using the function *CompareEdge*. Those common edges that share one common end point with the 'First Edge' will be selected. Hence the two edges adjacent to the selected edge will be selected first as shown in Figure 4.3b.

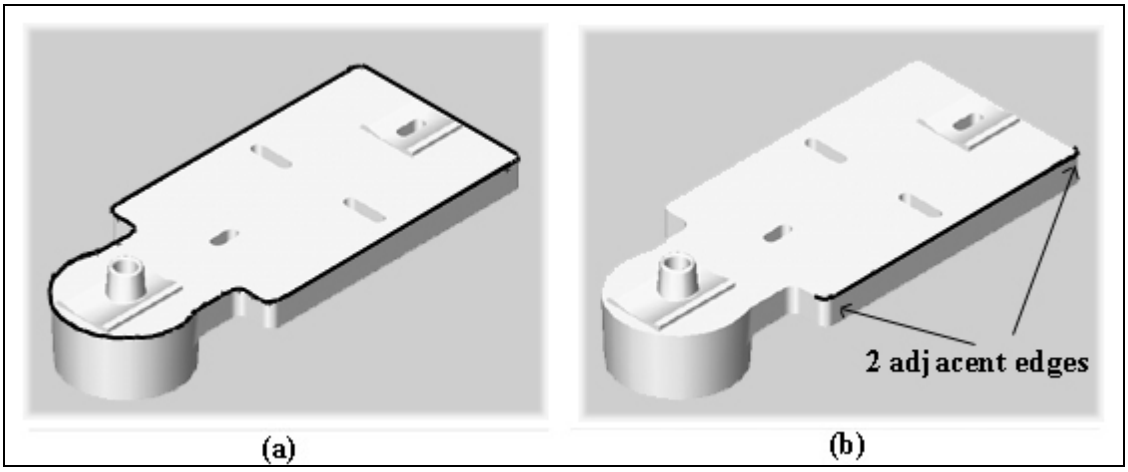


Figure 4.3: (a) All the edges in boundary face (b) 2 adjacent edges selected

Step 5: The algorithm will then repeat using the newly selected edges to select two more edges adjacent to the previous two selected edges as shown in Figure 4.4a.

Step 6: This algorithm will continue until all the edges that made up the 'Parting Line' are selected as shown in Figure 4.4b.

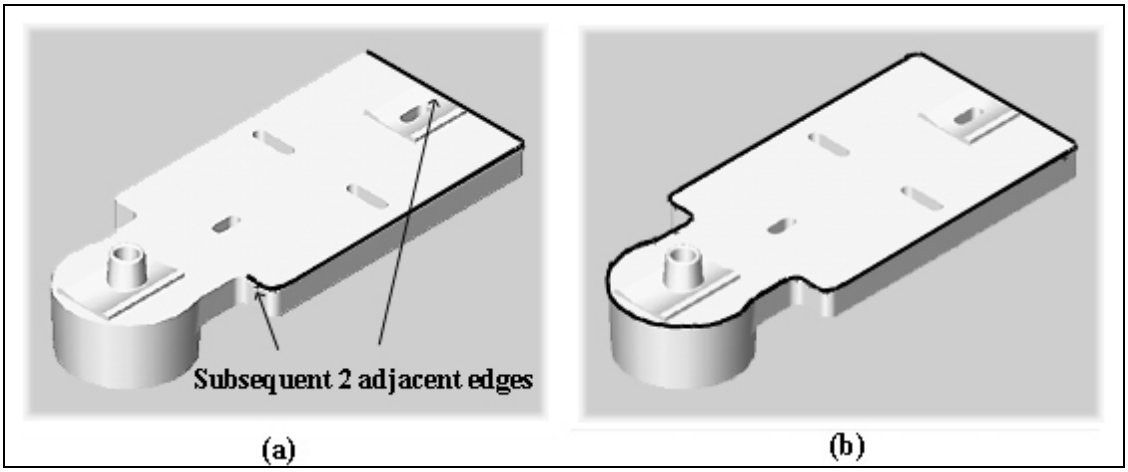


Figure 4.4: (a) Subsequent 2 adjacent edges selected (b) The 'Parting Line'

4.2.2 Algorithm for 'Parting Face Search'

'Parting Face' is defined as the faces of the part model that are used for parting the bolster into ejector and cover cavity. Similar to parting line, the parting face may contain hundreds of faces for a complex die casting part model. This algorithm allows the die designer to effortlessly select the parting face.

The algorithm for the function *SelectTangentFaces* is as follows:

```
SelectTangentFaces(swFace):

swFace->IGetFirstLoop(&swThisLoop)

int b=0;

//Repeat for this loop
while (swThisLoop != NULL)
{
        swFirstCoEdge = NULL;
        //Get first co-edge
        swThisLoop->IGetFirstCoEdge(&swFirstCoEdge)

        //Get 1st CoEdge params
        swFirstCoEdge->IGetCurveParams(FirstCoEdge_ptr);
        swThisCoEdge = swFirstCoEdge;
```

```
//repeat while subsequent co-edge is not the same as first edge
do
{
	int e = 0;

	//Get partner co-edge
	swThisCoEdge->IGetPartner(&swPartnerCoEdge)

	//Get face from partner co-edge
	swPartnerCoEdge->IGetLoop(&swPartnerLoop);
	swPartnerLoop->IGetFace(&swPartnerFace);

	//Compare with selected faces
	for (int a=1; a<count+1; a++)
	{
		smgr->IGetSelectedObject3(a, &ent_unk_chk);
		ent_unk_chk->QueryInterface(IID_IFace,(LPVOID
		*)&face_chk);

		swPartnerFace->IIsSame(face_chk, &chk);

		if (chk == TRUE)
		{
			e = 1;
		}
	}

	//Compare with boundary faces
	if ( e != 1 )
	{
		//Compare with boundary faces
		for (int c=0; c<num_list; c++)
		{
			m_lBoundaryFace.GetText(c, rString);
			rString = rString.Right(2);
			int d = atoi(rString);
			swPartnerFace->IIsSame(BFace[d], &chk1);
			if (chk1 == TRUE)
			{
				e = 1;
			}
		}
	}

	//Select the face procedure
	if (e !=1)
	{
		//num++;
```

```
                    swPartnerFace->QueryInterface(IID_IEntity,(LPVOID
                    *)&swEntity)

                    //Select the new face
                    swEntity->SelectByMark(TRUE, 1, &chk)
                }

                // Move on to the next co-edge
                swThisCoEdge->IGetNext(&swThisCoEdge)

                //Test that the coedge does not repeat
                cond = FALSE;
                swThisCoEdge->IGetCurveParams(NextCoEdge_ptr);
                for (int f=0; f<5; f++)
                {
                    if (FirstCoEdge[f] != NextCoEdge[f])
                    {
                        //coedge is not repeated
                        cond = TRUE;
                        break;
                    }
                }

        }
        while (cond == TRUE);

        swThisLoop->IGetNext(&swThisLoop);
}
```

The ‘Parting Face Search’ algorithm can be divided into 6 steps.

Step 1: The die designer is required to specify a list of ‘Boundary Faces’. ‘Boundary Faces’ as shown in Figure 4.5a are used to indicate the border of the ‘Parting Face’.

Step 2: Then the die designer needs to select a single face that makes up the ‘Parting Face’. Figure 4.5b displayed the selected face that will be used to generate the ‘Parting Face’. This face is termed the ‘First Face’.

Step 3: The algorithm first stores all the ‘Boundary Faces’ specified by the user into an array. The next procedure is to gather information from the ‘First Face’ and extracts the ‘First Loop’ belonging to that face. The ‘First Loop’ as shown in Figure

4.5c is defined as the external loop (edges) surrounding the selected face. The algorithm will then obtain the 'First Coedge' belonging to the 'First Loop'. 'Coedges' refer to the edges that made up a loop. As an edge is the intersection of two faces, an edge consists of two 'coedges'. One 'coedge' belongs to one face. Each face will utilize its own 'coedge' to refer to the common edge.

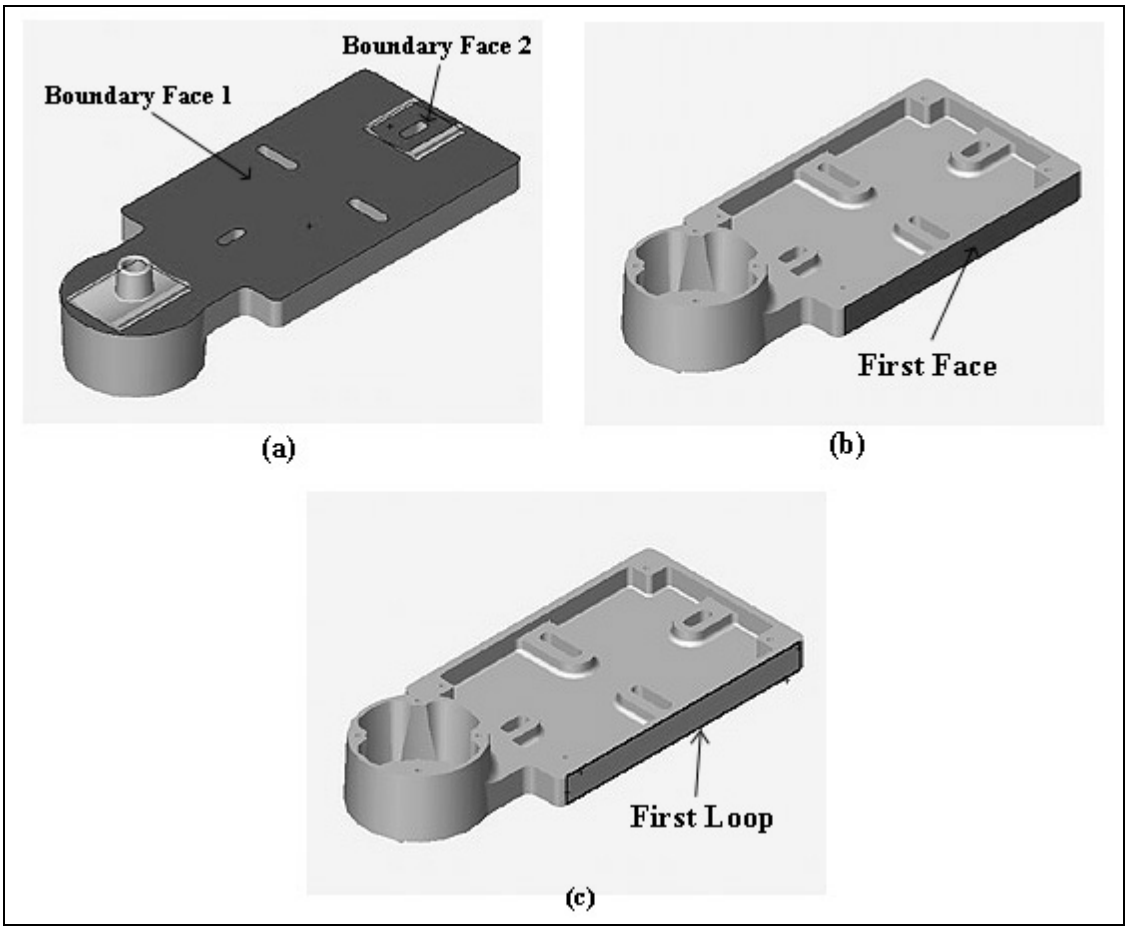


Figure 4.5: (a) Boundary faces (b) 'First Face' (c) 'First Loop'

Step 4: The algorithm will next get the 'partner edge' of the 'coedge'. 'Partner edge' of a 'coedge' refers to the adjacent 'coedge' that makes up the common edge. From the 'partner edge', the 'partner face' can be obtained as shown in Figure 4.6a. The 'partner face', the common edge and the two 'coedges' are as shown in Figure 4.6b. The 'partner faces' will be compared with all the 'Boundary Faces'. If the

'partner faces' do not correspond to any of the 'Boundary Faces', the 'partner faces' will be selected.

Step 5: The algorithm will then move on to the next 'coedge' belonging to the 'First Face' and repeat Step 4. Hence the first group of faces selected will be those adjacent to the 'First Face' as shown in Figure 4.7a. The bottom face not shown in Figure 4.7a is also selected.

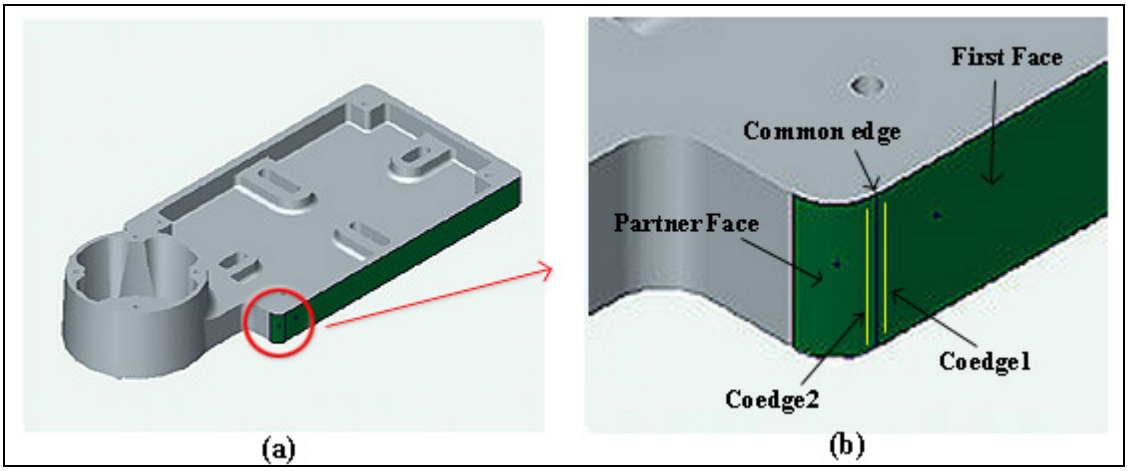


Figure 4.6: (a) 'Partner Face' selected (b) Enlarged view

Step 6: Continue with Step 5 until all the faces that made up the 'Parting Face' is selected as shown in Figure 4.7b. Figure 4.7c displays the underside, which shows the faces that do not made up the 'Parting Face'.

4.2.3 Algorithm for 'Hole Patching'

'Hole patching' is the technique for filling up holes with a planar surface. If the holes identified are non-planar, the system will prompt the die designer that the holes are non-planar and will need other techniques to fill the holes. The system provides die designers three options for the 'Hole Patching' function. The three options are as follows:

- **Automatic:** The system patched all the holes in a selected face.

- **With prompting:** The system prompts the die designer whether to patch the hole, at every hole; Useful when not all the holes are to be patched.

- **Manual:** The system patched the hole that the die designer manually selected.

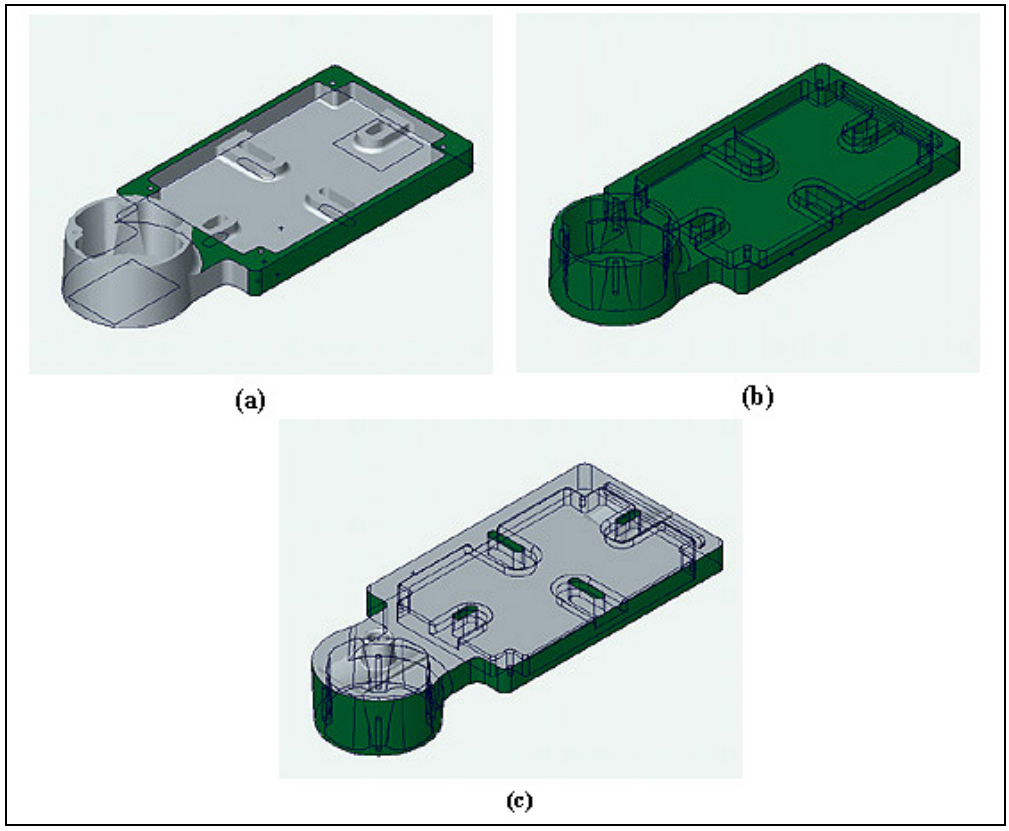


Figure 4.7: (a) First group of selected faces (b) 'Parting Face' (c) Unselected faces

The algorithm for "Hole patching" is as follows:

```
//Get the first loop of edges in the selected face
Loop = SelectedFace->GetFirstLoop

While (Loop != NULL)
{
        If (Loop->IsOuter == TRUE)    //Check for inner or outer loop
        Loop =  Loop->GetNext    //Get next loop if loop is outer

        Else   // If loop is inner
        {
                Loop->GetEdgeCount(num_edges);    //Find number of edges

                //Allocate array to store edges
                LPEDGE *edgelist = new LPEDGE[num_edge];

                Loop->IgetEdges(&edgelist);    //Store edges in array
```

```
                for (i=0; i<num_edge; i++)
                {
                        edgelist[i]->SelectByMark;
                        //Patch the holes
                        if (PartDoc->InsertPlanarRefSurface() == FALSE)

                                //Inform die designer that holes cannot be patched
                                AfxMessageBox("The selected hole cannot be patched");
                }
        }
}
```

The 'Hole Patching' algorithm can be divided into 5 steps.

Step 1: The die designer is required to select a face where there are holes to be patched. Figure 4.8a shows an example of a selected face.

Step 2: The algorithm begins by getting the 'First Loop' on the selected face. The 'First Loop' obtained is normally the biggest among all the loops present in the face. Figure 4.8b depicts the 'First Loop' obtained.

Step 3: The next step is to check whether the 'First Loop' obtained is an outer loop of the selected face. An outer loop is shown in Figure 4.8b. If the 'First Loop' is an outer loop, the system is to proceed to the next loop and repeat the step until the 'Next Loop' is an inner loop. The 'Next Loop' refers to the loop that is subsequently selected after the 'First Loop'

Step 4: When the 'Next Loop' is an inner loop like those shown in Figure 4.8c and Figure 4.8d, the algorithm will extract all the edges in the loop and store it in an array. These edges are then selected and the algorithm will attempt to fill these edges using a planar surface. If the edges do not form a hole as shown in Figure 4.8c, the planar surface will not be created. Instead the die designer will be prompted with a message stating that the selected edges cannot be patched. Similarly, the die designer will also be informed if the selected edges form a non-planar hole.

Step 5: The specific functions from the SolidWorks API can be call upon to cover the identified planar holes. The "patched" holes are shown in Figure 4.8e.

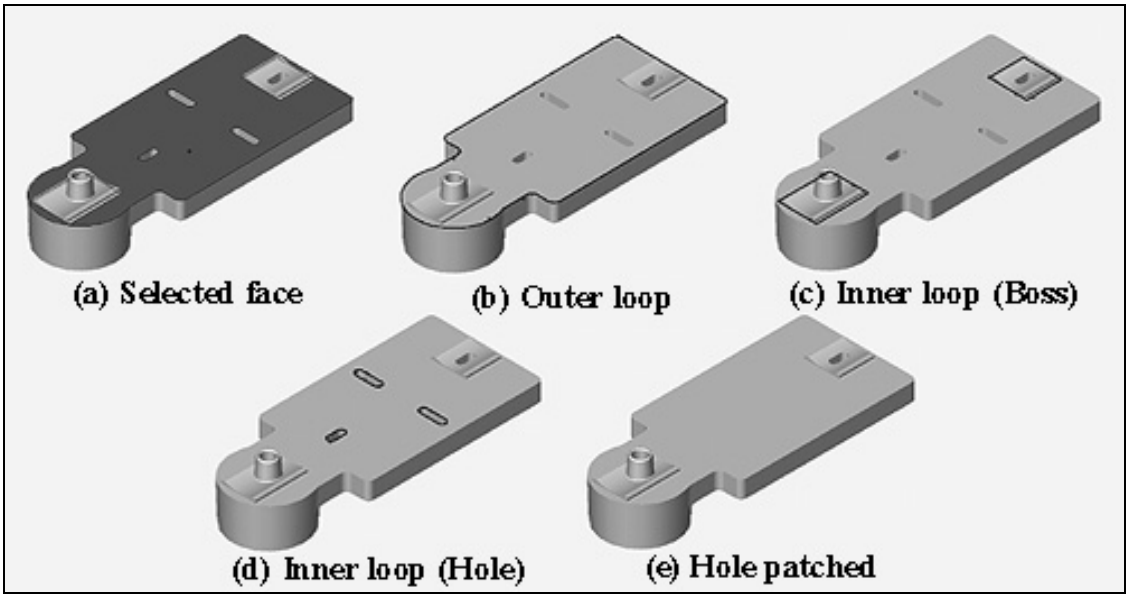


Figure 4.8: Graphical illustration of 'Hole Patching' algorithm

4.3 Feature-based and constraint-based modeling

In this study, various predefined die casting features are created. These features are standard components that are used extensively in die design for die casting, like die bases and their components. Several techniques are used to create the various solid models of the die casting features, depending on their complexities. The creation of the various solid models of the die casting features was by sketching the topology of the feature's shape and then applying linear and angular dimensions and dependencies (or relations) such as horizontality or parallelism to the topology-objects. The dimensions and dependencies are referred to as the constraints. These constraints can be applied through the following functions available in SolidWorks:

- **Mating -** It is a mating relationship between two geometric objects on two different components in an assembly. Some of the common types of mating methods are Angle, Coincident, Concentric and Distance.

- **Add Relations –** It is the creation of geometric relations (like tangent or perpendicular) between sketch entities, or between sketch entities and planes, axes, edges, or vertices.

- **Equations -** It involves the addition of mathematical relations (equations) between model dimensions, using dimension names as variables. The equations can be set between parts, between a part and a sub-assembly, with mating dimensions, and so forth.

4.3.1 Constraint through Mating

An example of a constraint through mating can be seen in the modeling of the die base. The top face of the head of the leader pin was given a coincident mate with the top face of the top plate. This will keep the top face of the head of the leader pin align with the face of the top plate as illustrated in Figure 4.9.

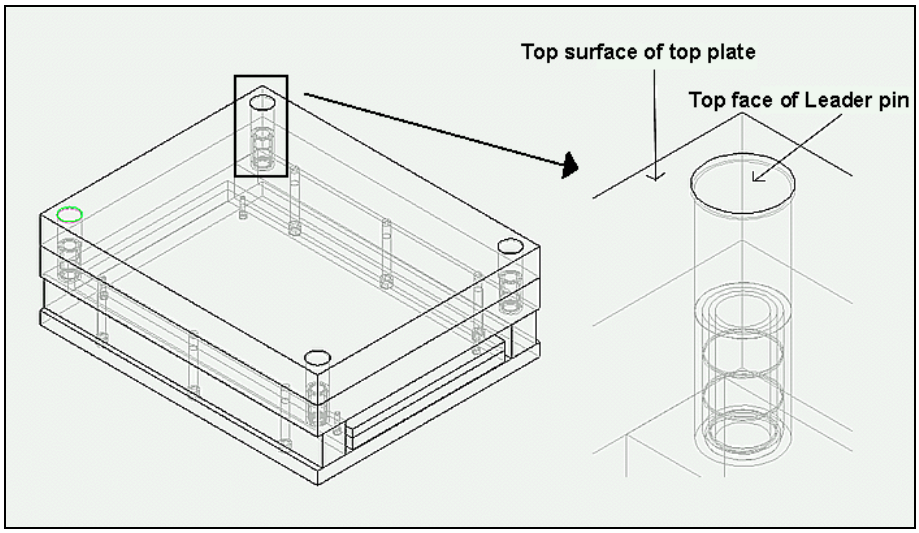


Figure 4.9: Top face of leader pin align with top face of top plate

4.3.2 Constraint through Add Relations

Gating component such as trapezoidal runner shown in Figure 4.10 is an example of a die casting feature that is governed by constraints through adding relations. This is

necessary to ensure that the runner's cross-section remains trapezoidal regardless of any changes to the runner's parameters.

The first step is to create an initial sketch as shown in Figure 4.10a. Next symmetrical relation about the horizontal axis is given to the two horizontal sides as shown in Figure 4.10b. This relation will ensure that the two horizontal edges are equidistant from the horizontal axis. The other two sides are also given symmetrical relation about the vertical axis as shown in Figure 4.10c. After these two relations are added, dimensions can be specified as shown in Figure 4.10d. The relations will remain regardless of the dimensions specified.

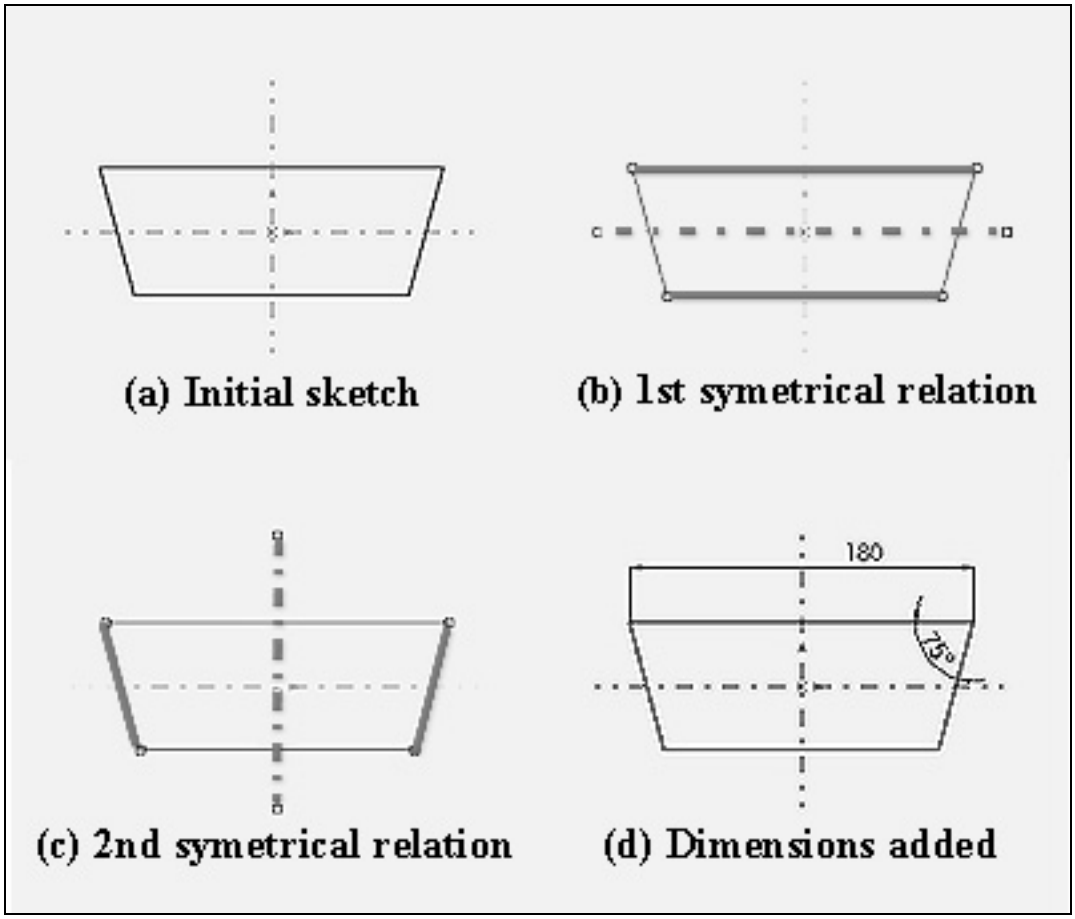


Figure 4.10: Relations added for trapezoidal runner

4.3.3 Constraint through Equations

Constraints through equations are often applied in assemblies of die casting features. It aids the editing of the various parameter values of the different parts that made up an assembly. For example, the slider assembly as shown in Figure 4.11 is made up of two parts - angle pin and slider body. A sketch named ‘Dimension’ consisting of lines and circles is created in the assembly document. The dimensions of these lines and circles are linked to the various parameters of the two parts through applying equations as shown in the Equations dialog box depicted in Figure 4.11. In this way, parameter values of the two parts are controlled by parameter values in a single sketch. Hence die designers can alter the parameters of the two parts by altering the dimensions of the sketch entities in the ‘Dimension’ sketch.

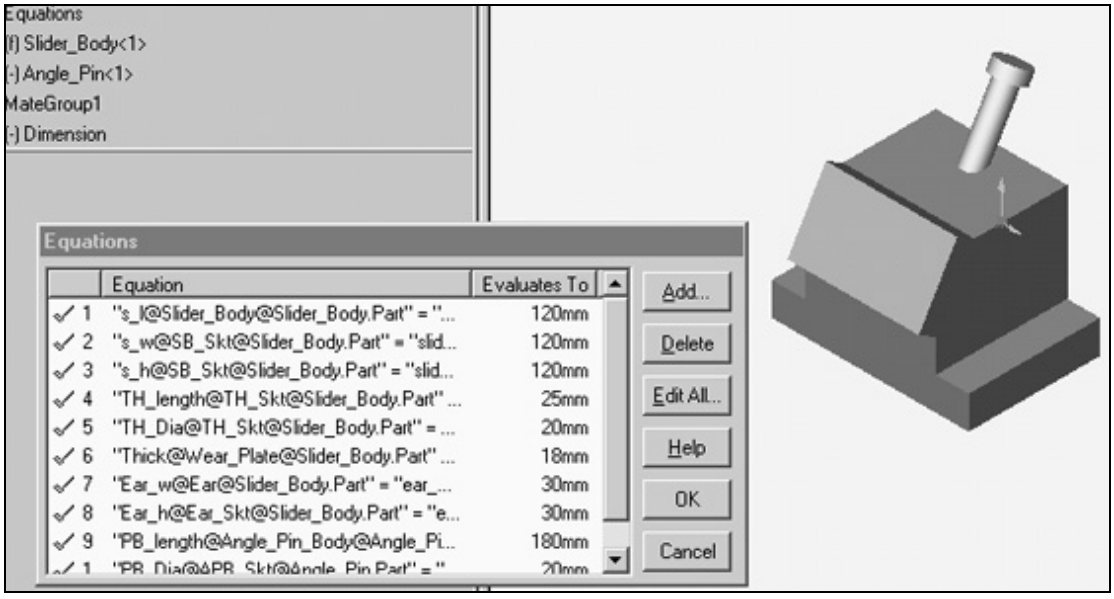


Figure 4.11: Slider assembly with the Equations dialog box displayed

This method greatly facilitates the editing of the parameter values of slider body and the angle pin. Examples of equations added are as follows:

1. "s_l@Slider_Body@Slider_Body.Part" = "slide_l@Dimension"

2. "PB_length@Angle_Pin_Body@Angle_Pin.Part" = “anglepin_l@Dimension”

The first equation links a parameter value ‘s_l’ of the slider body to a parameter value ‘slide_l’ in the ‘Dimension’ sketch while the second equation links a parameter PB_length’of the angle pin to a parameter value ‘anglepin_l’ in the ‘Dimension’ sketch.

4.4 Table-driven design for assembly

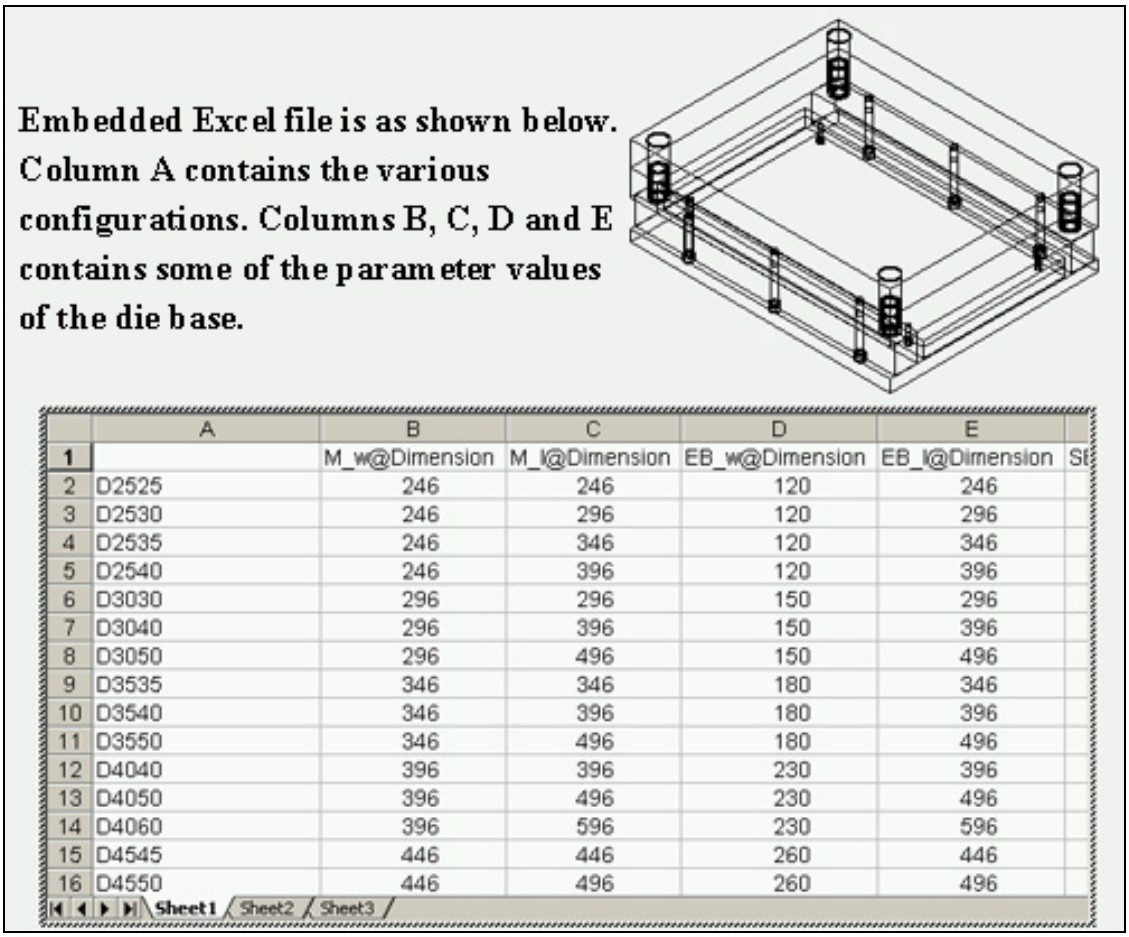
As mentioned in section 4.1, the die casting design process involves several standard die casting components. Vendors with their own catalogue parts often supply these standard components. In the proposed die design system for die casting, the DME series D die base is modeled where only some predefined sets of dimension values are addressed according to the catalogue supplied by DME.

SolidWorks CAD system allows the storage of these sets of predefined parameter values in tables. In SolidWorks, a set of predefined parameter values is called a configuration and the table where configurations are stored is called a design table.

Configurations allow the creation of multiple variations of a part or assembly model within a single document. It provides a convenient way to develop and manage families of models with different dimensions, components, or other parameters.

Design tables provide a convenient way to create and manage configurations in an easy-to-use worksheet. A design table can control several parameters like the dimensions and suppression state of features, configuration properties, comments, etc. It can be used in both part and assembly documents, and it can be displayed in drawings. A design table allows for multiple configurations of parts or assemblies by specifying parameters in an embedded Microsoft Excel worksheet. The design table is

saved in the model document and is not linked to the original Excel file. Therefore the changes made to the model will not alter the original configurations in the embedded Excel file. As a result additional security is provided for the die design system for die casting. The DME series D die base, with a design table embedded in their assembly files where the different configurations are stored is shown in Figure 4.12.



	A	B	C	D	E
1		M_w@Dimension	M_l@Dimension	EB_w@Dimension	EB_l@Dimension
2	D2525	246	246	120	246
3	D2530	246	296	120	296
4	D2535	246	346	120	346
5	D2540	246	396	120	396
6	D3030	296	296	150	296
7	D3040	296	396	150	396
8	D3050	296	496	150	496
9	D3535	346	346	180	346
10	D3540	346	396	180	396
11	D3550	346	496	180	496
12	D4040	396	396	230	396
13	D4050	396	496	230	496
14	D4060	396	596	230	596
15	D4545	446	446	260	446
16	D4550	446	496	260	496

Sheet1 Sheet2 Sheet3

Figure 4.12: DME series D die base with embedded Excel file

4.5 Use of Reference Geometry and Sketch Entities

This is the methodology for the placement of die casting components. In the proposed die casting die design system, die designers often need to load in predefined die components. In order to position the die components in the intended locations, these

die components are sometimes constructed together with sketch entities that act as position markers. The sketch entities include sketch point and centerline.

Another common method used for the purpose of placement of die casting components is the use of reference geometries. All SolidWorks files contain a standard set of reference geometries. These reference geometries include a coordinate system and three reference planes – Top, Right and Front planes. Therefore no additional reference geometries need to be created for the purpose of placement of die casting components.

The placement of the die casting components generally uses either of the two methods. However, both methods may be needed in order to place a more complicated die casting component.

4.5.1 Use of Sketch Entity

Examples of the usage of sketch entity in the placement of die casting components include runner, gate, overflow, ejector pin and cooling components. This section will show how the system uses sketch entities to place a runner. The principles are similar to the placement of the other die casting components.

A standard overflow is stored in the gating feature library and it can be added to a die casting project as and when it is needed. The standard overflow is constructed together with two sketch points as shown in Figure 4.13. The sketch points marked the end points of the standard overflow and they will be used as the mating points during the placement of the overflow.

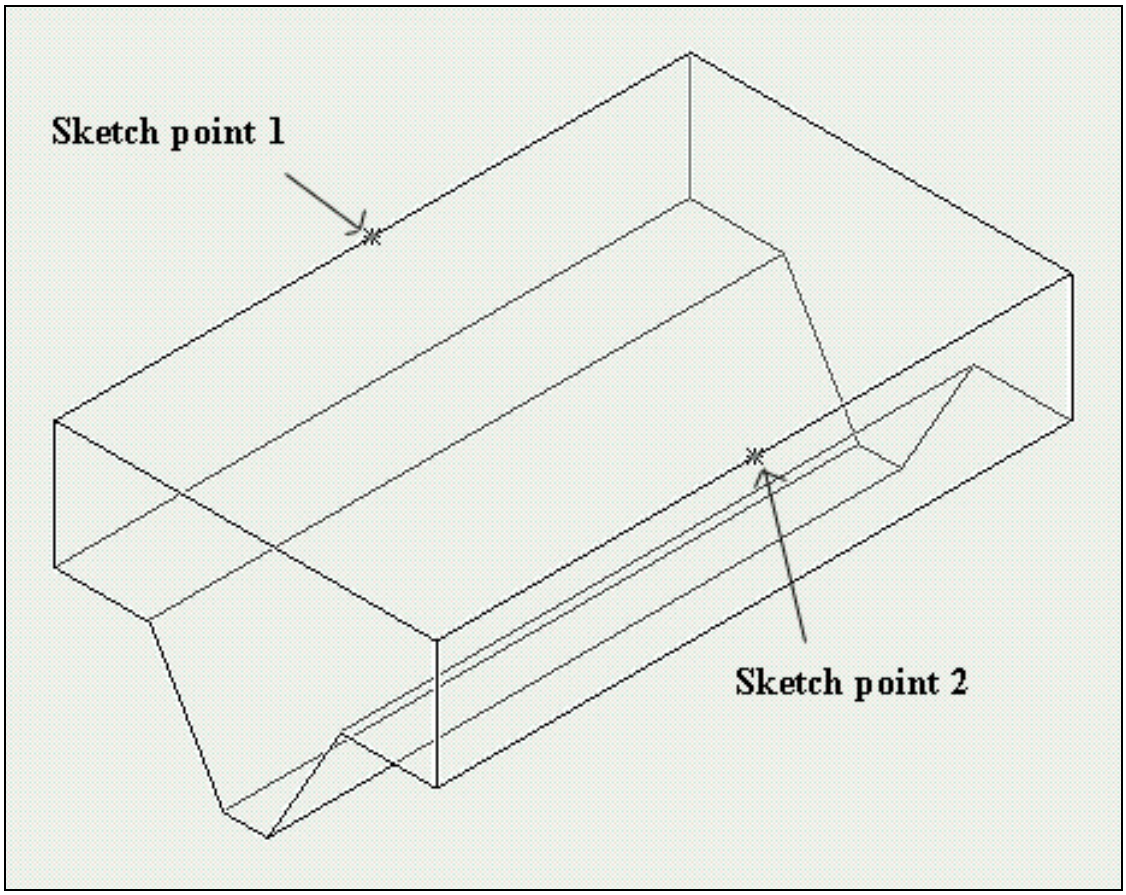


Figure 4.13: Enlarged view of the overflow model with two sketch points

When placing the overflow, the die designer needs to sketch a line on the position where the overflow is to be added as shown in Figure 4.14. The length of the line will indicate the length of the overflow to be added. The die casting die design system will then extract information like the end points and the length of the sketched line. The length of the overflow to be added will then be set accordingly and placed on the sketched line by mating the two end points with the pre-defined sketch points of the overflow. An illustration of the placement for the overflow is depicted in Figure 4.15.

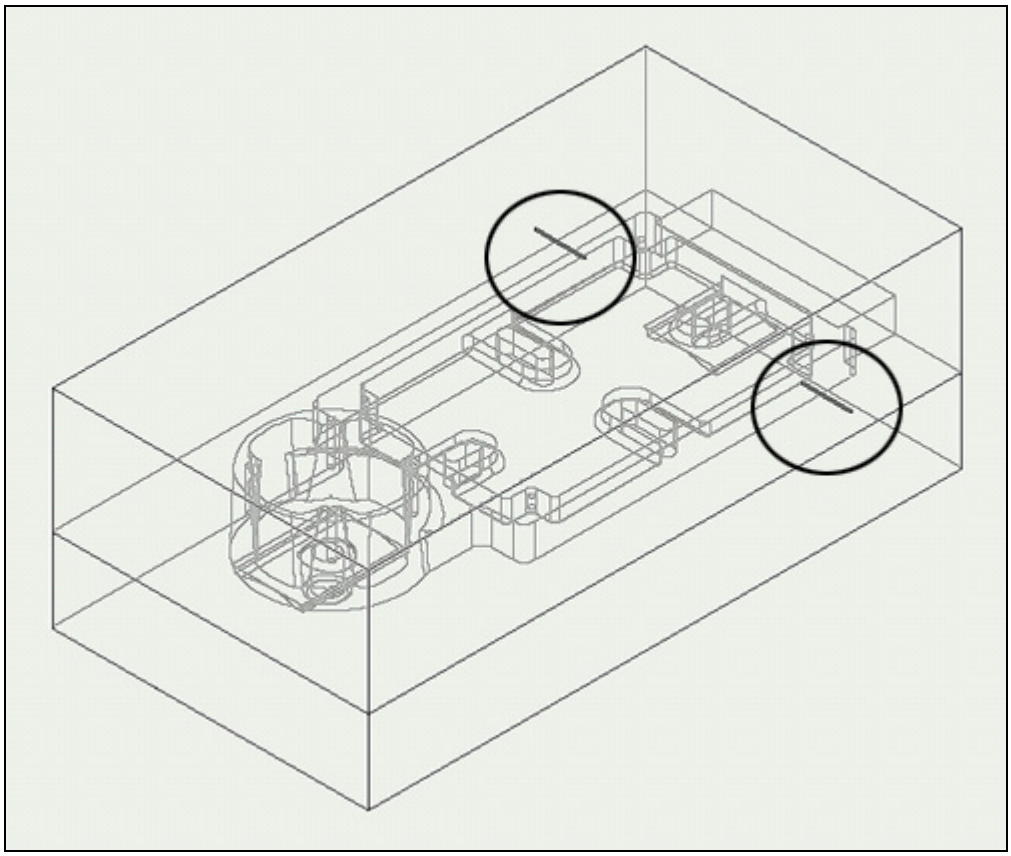


Figure 4.14: Two sketch lines indicating the position and length of the overflows

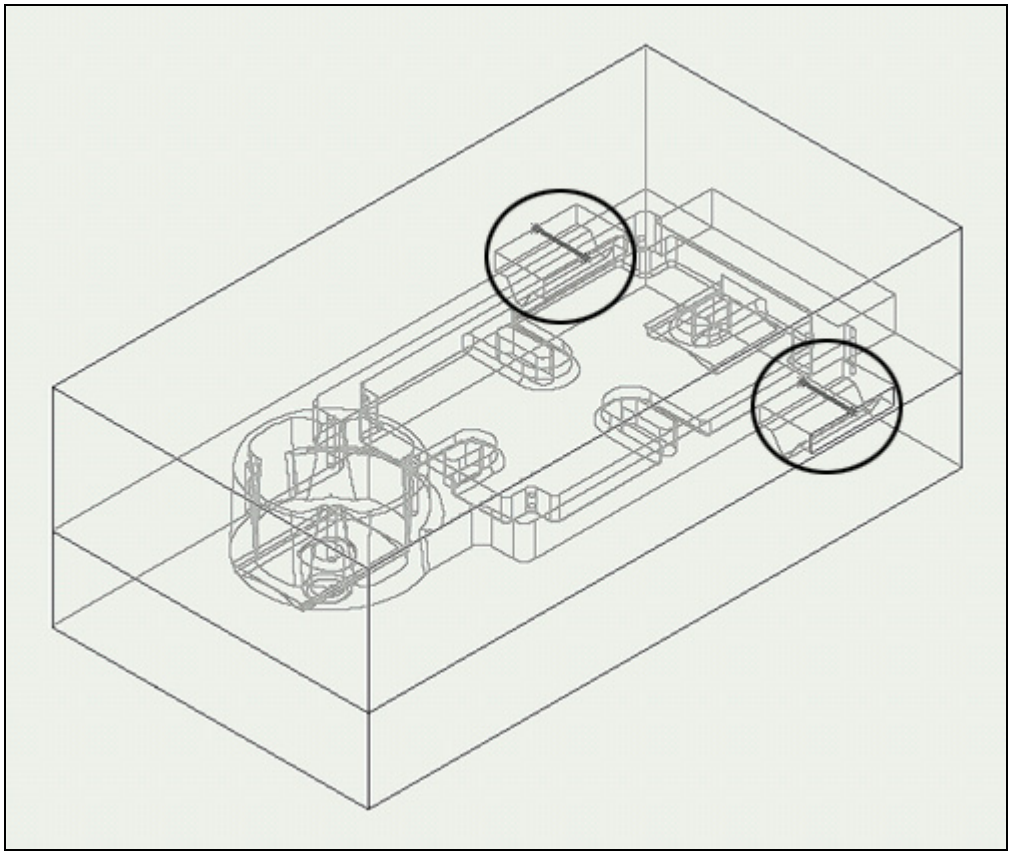


Figure 4.15: The placement of two overflows in the positions indicated

4.5.2 Use of Reference Geometry

The coordinate system is the most common reference geometry used for the purpose of placement. A good example is the placement of the containing box. The predefined containing box is a rectangular block. As the name suggests, the containing box is used to contain the product model. In order to contain the product model, the containing box is constructed with the coordinate system at the centroid of the block as shown in Figure 4.16.

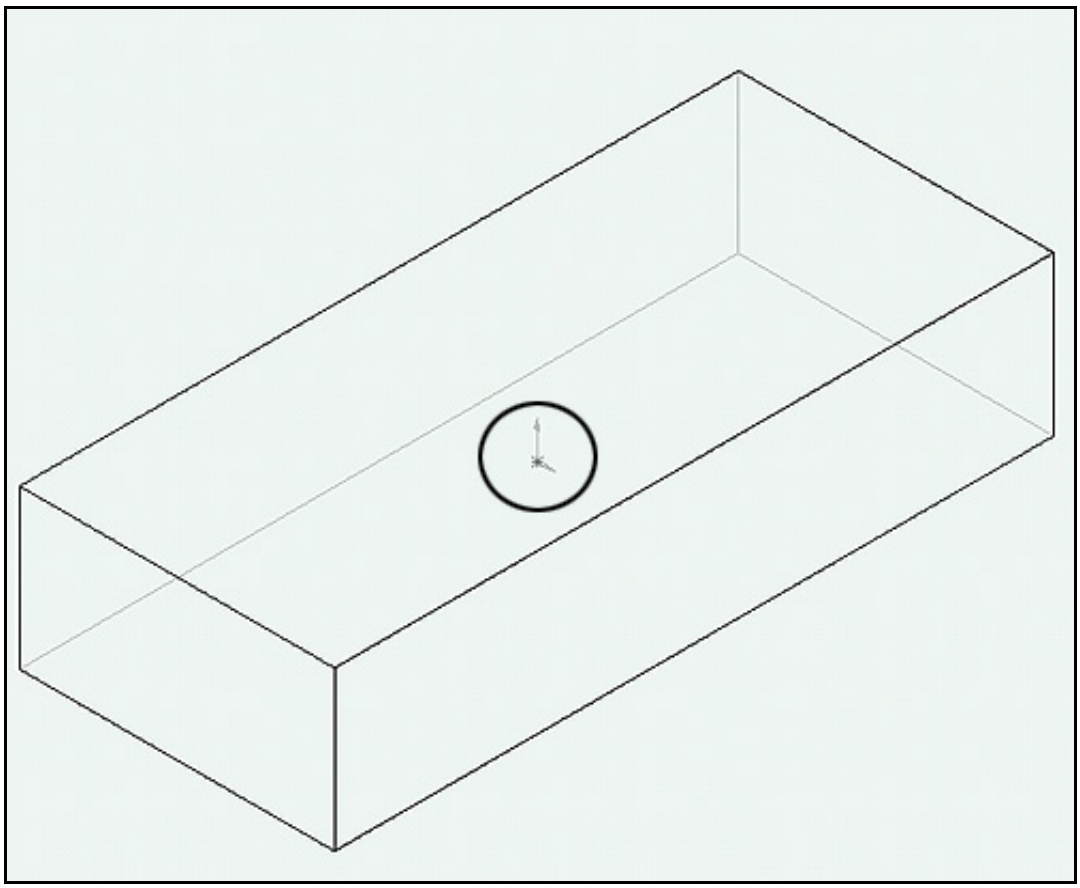


Figure 4.16: The containing box with coordinate system at the centroid position

Another coordinate system is also created at the centroid of the product model as shown in Figure 4.17. The containing box is then positioned by coinciding the coordinate system with the centroid of the product model. An illustration of the mating of the containing box with the product model is shown in Figure 4.18.

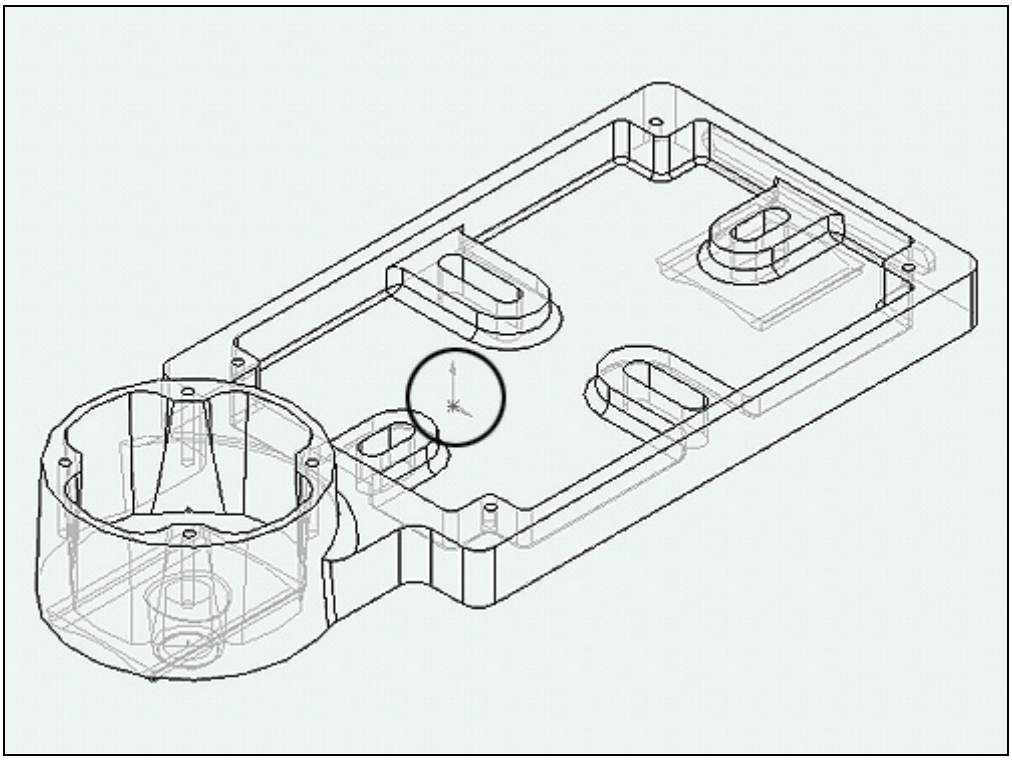


Figure 4.17: Product model with coordinate system at the centroid position

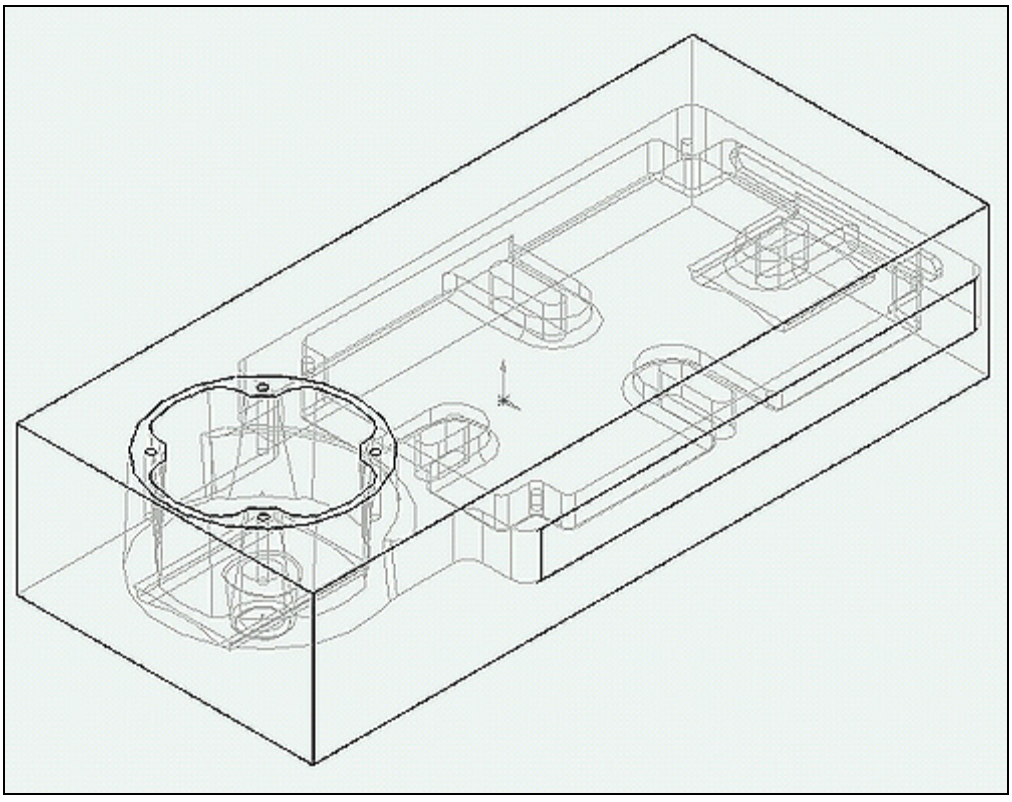


Figure 4.18: Product model mated with the containing box

CHAPTER 5 : SYSTEM ARCHITECTURE & DESIGN

In this chapter, the system requirements are first discussed. The architecture of the computer-aided die casting die design system is then presented and the individual modules described.

5.1 System Requirements

Research had shown that there is a need for a die casting die design system that is more comprehensive than previous works. The proposed system design should adhere to the following guidelines:

a) Fully integrated with SolidWorks – the chosen platform.

b) Enable die designers to alternate between the different stages of die casting design process.

c) Enable die designers to edit die components as and when needed.

d) Enable die designers to stop at any stage of the die casting design process and carry on from that stage on a later date.

5.2 System Overview

Figure 5.1 illustrates the algorithm of the proposed die casting die design system. The algorithm follows closely to the die design process for die casting as shown earlier in Figure 1.3. The only difference is that the algorithm allows continuation of existing die casting die design projects, and most importantly the editing of the projects.

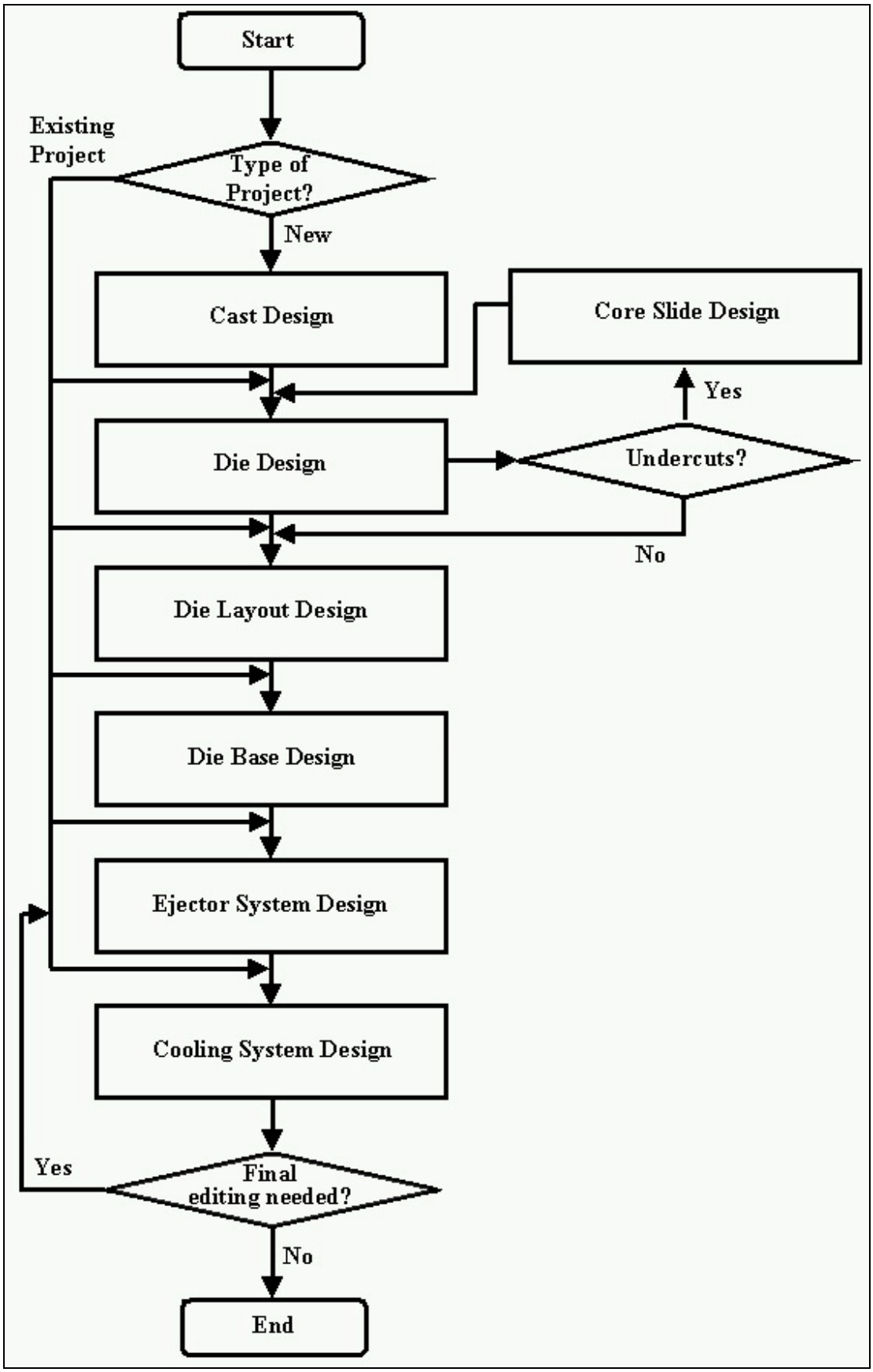


Figure 5.1: Algorithm of Die Design System for Die Casting

The system will begin by identifying whether the die casting die design project is an existing or new project. A new project will proceed on to the next stage while an existing one will join the stage that the project last ended. For a new project, the next stage is cast design, which include the determination of shrinkage factors, measuring units, etc. It is essential that the shrinkage design be properly completed before proceeding to the other stages.

The subsequent stage will be the die design. In die design, the most important task is to determine the parting line. The cavity inserts can only be created after the parting line had been finalized. Before the system proceeds to the next stage, it may need to go through an optional stage for undercut features. This stage is useful if the die casting die design project involves undercut features, as it assists in the design of a core slide.

In die layout, the system aids the die designer in the design of the gating system of the project. The gating system begins with the design of the cavities layout, followed by the addition of gates, runners, overflows and vents. This is then followed by the die base design stage where the system assists in the inclusion of the die base. The die base is designed at later stage because with the completion of the gating system, the system will be able to better estimate the size of the die base needed.

The final two stages are the ejector system design stage and the cooling system design stage. The ejector system is added after the die base had been included because the system needs to retrieve information regarding the location of the ejector plate. The location of the ejector plate is vital to the placement of ejector pins as they are normally placed on the ejector plate of the die base. The cooling channels are generally added at the end as the system can better reflect on the amount of free space available in the die base to place the cooling channels. An important area to note is that the

system allows editing to be made to any of the stages even after the die designer had completed the entire design.

The proposed die casting die design system consists of eight design modules. A design module handles each stage, while one additional module is catered to the addition of standard components to the die casting die design project. Some of the design modules that are more complicated or require several steps are divided into further sub-modules. Standard die casting component feature libraries are also built and added to some of the design modules. The system architecture is illustrated in Figure 5.2. The subsequent sections will describe in details the design of the individual design modules in the die casting die design system and their applications.

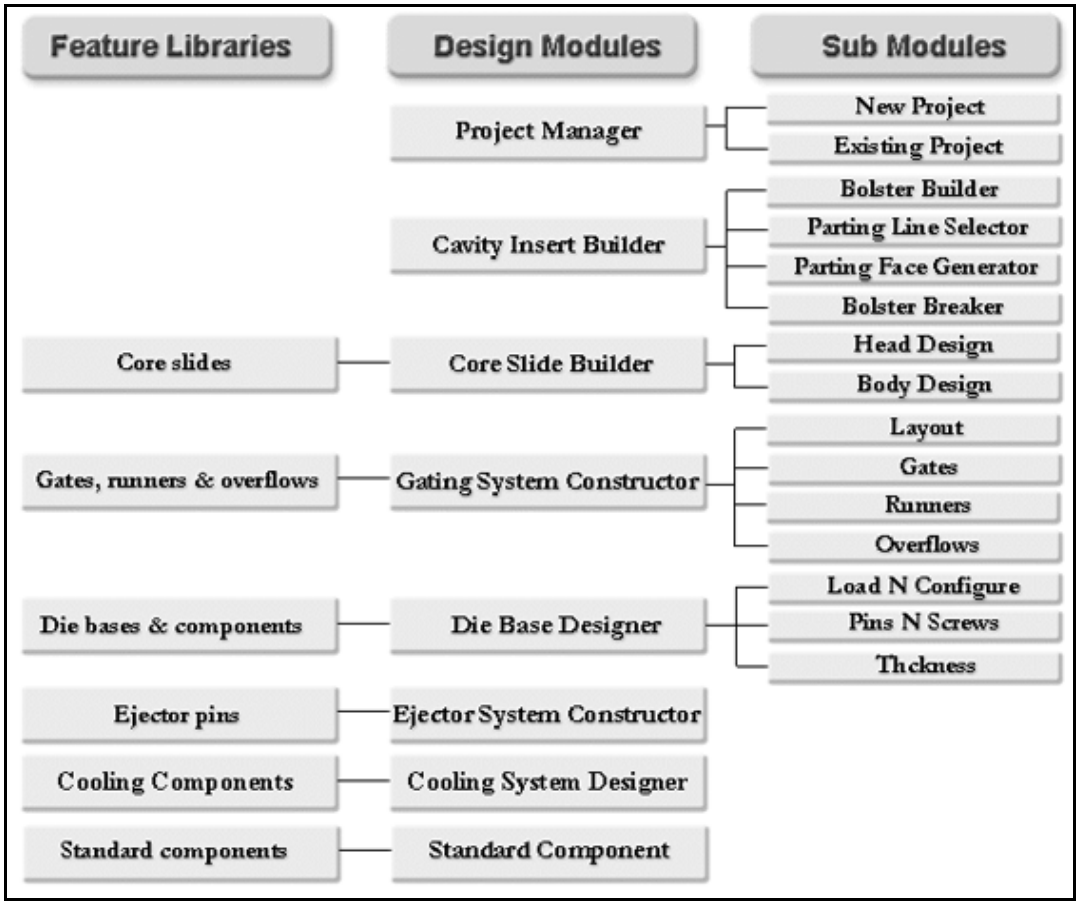


Figure 5.2: System Architecture of the Die Casting Die Design System

5.3 Design of 'Project Manager' Module

The module 'Project Manager' is designed to handle the cast design stage. This design module is also designed to play a secondary role of administrating all the die projects under the die casting die design system. All die projects must be launched through this module. This is important as the die casting die design system, been an add-on to a commercial CAD software, needs to provide a design module that can open its own project. Using the commercial CAD software 'New' or 'Open File' functions will complicate matters as information pertaining specifically to the die casting die design cannot be retrieved or recorded easily.

For that reason, 'Project Manager' is equipped with its own sub-modules that works just like any Windows 'New' and 'Open File' functions and they are named 'New Project' and 'Existing Project' respectively.

The sub-module 'New Project' allows the die designer to launch a new project (Interface shown in Figure 5.3) while the sub-module 'Existing Project' launches an existing one. An existing project refers to a previous project that the die designer did not manage to complete earlier. This is useful as some complicated die projects may take a few hours or even days. The die casting die design system provides die designers with the option of saving the current work and continue on another date. Hence when a die designer launches an existing project, the project will automatically return to the stage it was last saved.

The design of the sub-module 'New Project' needs to keep to the following criteria:

1. Provides a standard function for die designers to browse for the product model

2. Predefines some of the files needed to automate or semi-automate the die

casting die design process.

3. Provides die designer the function to set shrinkage factors, measuring units, etc.

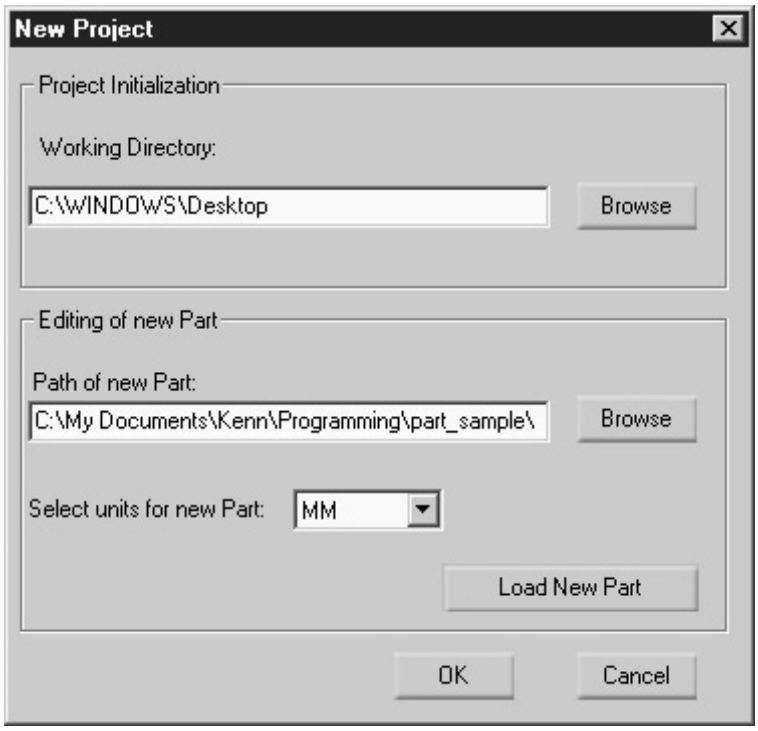


Figure 5.3: Interface of New Project

Designing a 'Path of product model' edit box and a 'Browse' button on the sub-module's interface as shown in Figure 5.3 can easily satisfy criterion 1. The 'Browse' button allows the die designer to browse for the product model from the various directories of the operating system. It works just like the Windows 'File Open' function, except that the product model will not be loaded into the system. It will only locate the file and displayed the path location in the 'Path of product model' edit box. The loading function is executing through another button 'Load New Part' so as to ensure that the die project had been properly initiated before the product model is loaded into the system.

To fulfill criterion 2, a 'Working Directory' is proposed. The working directory indicates the location in the operating system where all the files belonging to the die

project will be saved. In this way, additional files can be created and stored in this 'Working Directory'. For example, a new assembly file can be created at the start of each project. This assembly file is the main assembly that is used to contain all the individual part files of die components. The first part file that it will contain will be that of the product model. Subsequently, other part files of the other die components like cavity inserts, die base, etc will be added to the main assembly file.

In order to conform to criterion 3, the interface of 'New Project' is designed with a combo box for die designers to select the project's measuring units. The system will then set the measuring units accordingly. Moreover, an option to provide a function to scale the product model is designed after the product model is loaded. The Scale dialog box as shown in Figure 5.4 is displayed to execute this function, which allows the die designer to select whether to scale the product model base on the centroid, origin or coordinate system. However this function only scales the solid geometry, for use in data export, cavities, and so on. It does not scale dimensions, sketches, or reference geometry.

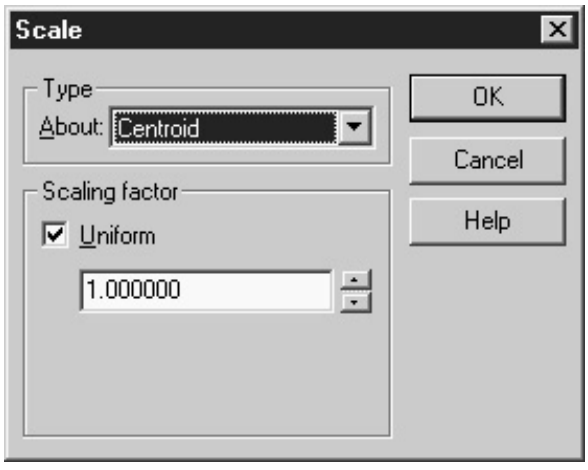


Figure 5.4: Scale dialog box

The design of the sub-module 'Existing Project' is less tedious. The design only needs to satisfy the criterion of retrieving information from the existing die project and then

launch the design module or sub-module, which the existing die project last used. As a result, a dialog interface is not needed.

5.4 Design of 'Cavity Insert Builder' Module

The 'Cavity Insert Builder' module is designed to handle the die design stage. As this stage is rather complicated, the module is designed with four sub-modules – 'Bolster Builder', 'Parting Line Selector', 'Part Face Generator' and 'Bolster Breaker', to aid the die designer in the creation of the ejector and cover cavity inserts.

5.4.1 Design of 'Bolster Builder' Sub-module

The design of the sub-module 'Bolster Builder' (Interface shown in Figure 5.5) needs to keep to the following criteria:

1. Semi-automate the creation of the 'containing box'.

2. Allows editing of the 'containing box' part.

To semi-automate the creation of the 'containing box', a predefined rectangular block part (containing box) is created and stored. When this part is added, the system will automatically obtain the product model's maximum length, breadth and height, and then configure the 'containing box' accordingly. Hence the default 'containing box' that is added will just 'fit' the product model. The 'containing box' will then be mated with the product model as discussed in section 4.5.2. The system is also designed to display the dimensions of the default 'containing box' for reference.

To allow editing of the 'containing box' part, the interface of the sub-module 'Bolster Builder' is designed with controls that allow the die designer to enter the increments in length, breadth and height. The length, breadth and height of the containing box are

then extended accordingly.

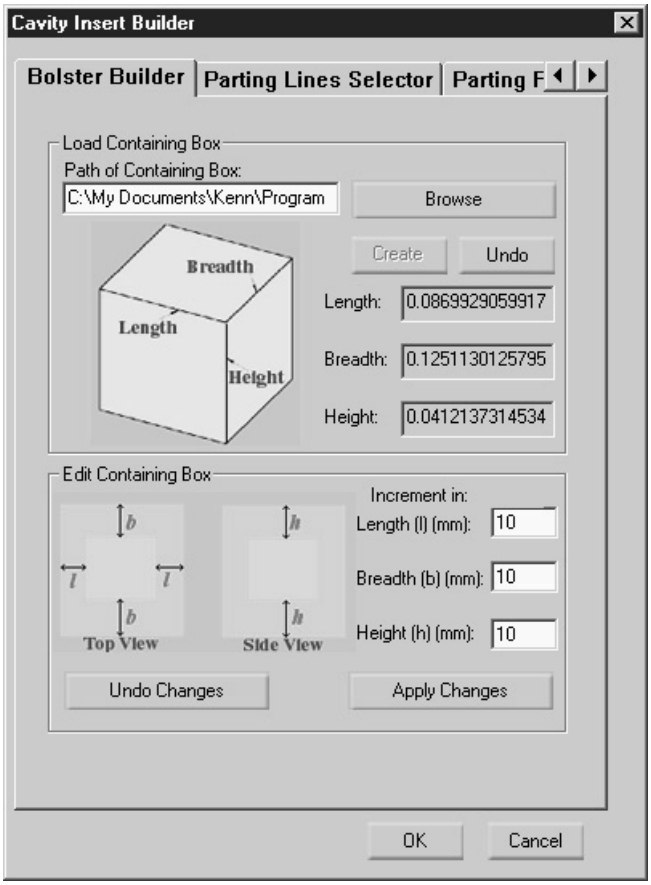


Figure 5.5: Interface of 'Bolster Builder'

5.4.2 Design of 'Parting Line Selector' Sub-module

The design of the sub-module 'Parting Line Selector' (Interface shown in Figure 5.6) needs to keep to the following criteria:

1. Semi-automate the selection of the 'Parting Line'.

2. Semi-automate the creation of a surface that divides the 'containing box' into two.

The algorithm to semi-automate the selection of the 'Parting Line' is discussed in section 4.2.1. The interface of the sub-module 'Parting Line Selector' is designed with controls that allow the die designer to add, delete and select the boundary faces and the

parting lines generated.

The creation of the surface that divides the 'containing box' into two is accomplished through radiating a surface from the 'Parting Line'. The system is designed such that the die designer only needs to specify the distance of the radiated surface to be extended and the reference plane to be based on. The reference plane refers to the plane that is parallel to the direction in which the surface is to be radiated. However, the die designer needs to note that the radiated surface must be extended beyond the containing box so as to distinctly divide the bolster into two parts.

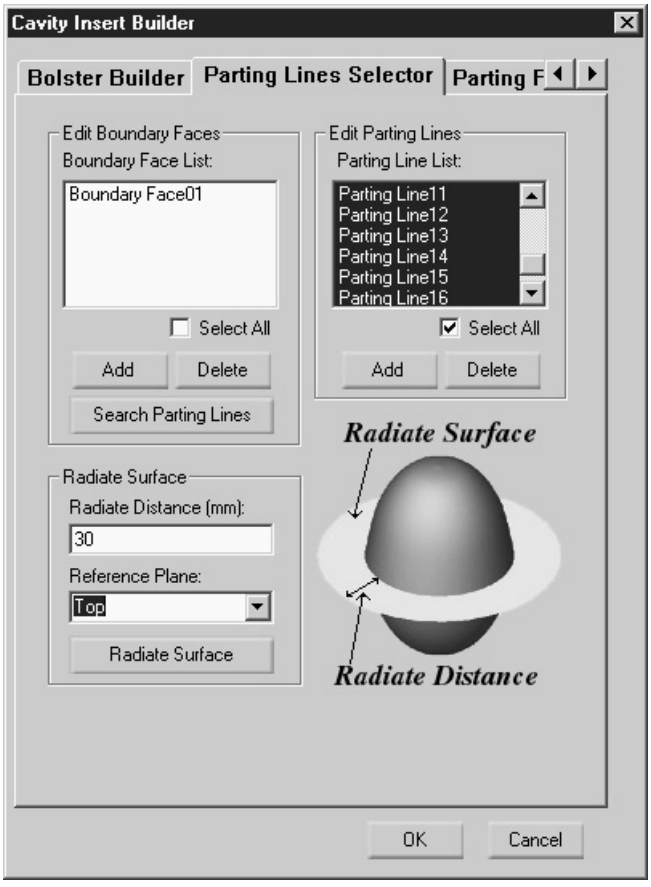


Figure 5.6: Interface of 'Parting Line Selector'

5.4.3 Design of 'Parting Face Generator' Sub-module

The design of the sub-module 'Parting Face Generator' (Interface shown in Figure 5.7) needs to keep to the following criteria:

1. Semi-automate the selection and the formation of the ‘Parting Face’.

2. Semi-automate the process of covering holes – ‘Hole Patching’

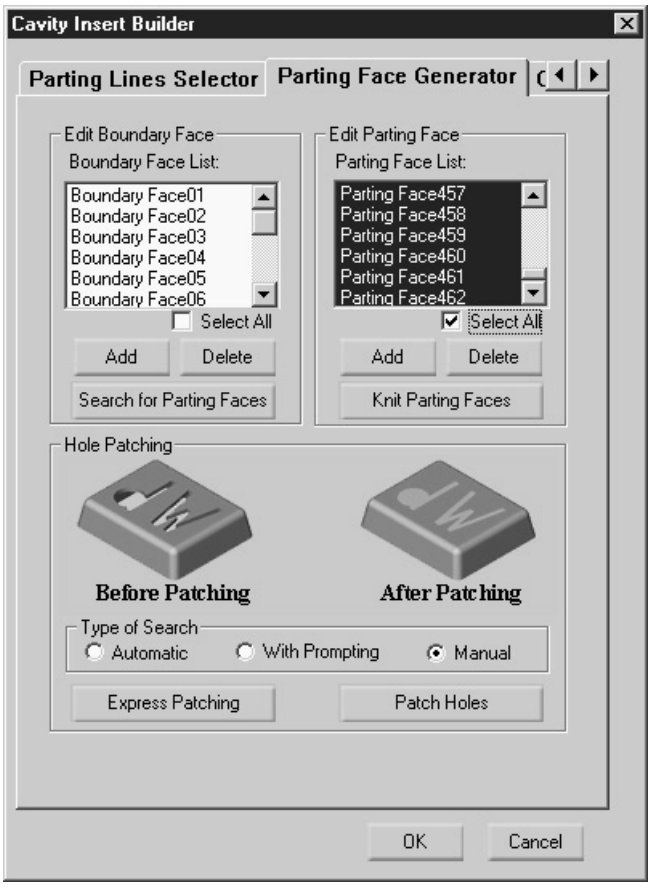


Figure 5.7: Interface of ‘Parting Face Generator’

The technique of searching for faces to form the ‘Parting Face’ is discussed in section 4.2.2. The interface of the sub-module ‘Parting Face Generator’ is designed with controls that allow the die designer to add, delete and select boundary faces and parting faces generated. Knitting all the selected faces together completes the formation of the ‘Parting Face’. The knitting function is a SolidWorks inherent function that combines two or more faces and surfaces into one. This surface body is termed ‘Parting Face’.

The interface of the sub-module ‘Parting Face Generator’ is also designed with controls that allow the die designer to cover existing holes in the knitted ‘Parting Face’ automatically or manually. The ‘Hole Patching’ algorithm is discussed in section 4.2.3.

5.4.4 Design of 'Bolster Breaker' Sub-module

The design of the sub-module 'Bolster Breaker' (Interface shown in Figure 5.8) needs to keep to the following criteria:

1. Automatically searches for surfaces to form the 'Parting Surface'.

2. Semi-automates the creation of the ejector and cover cavity inserts.

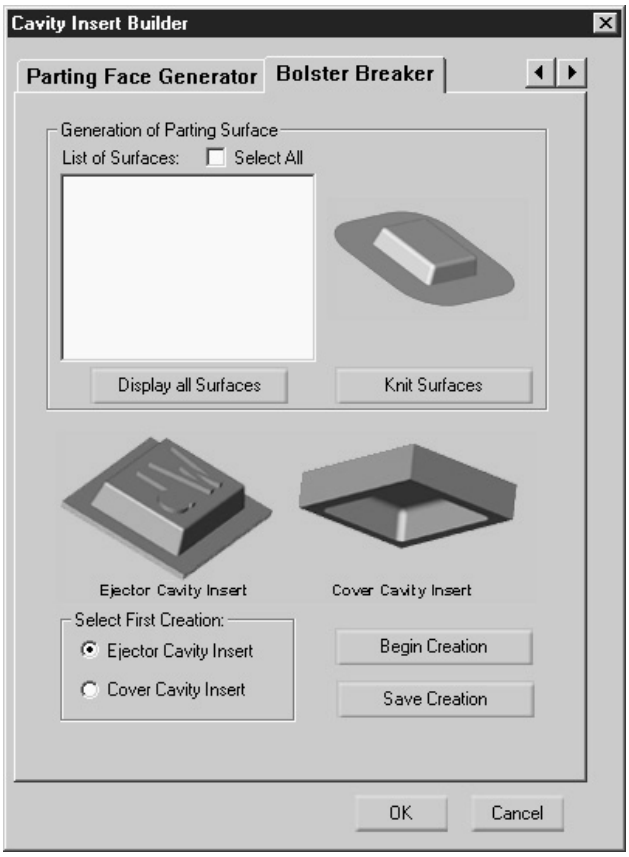


Figure 5.8: Interface of 'Bolster Breaker'

The system is designed to automatically search and then list out all the surfaces generated earlier for the die designer. These surfaces include the previous generated parting face, patched surfaces and other manually generated surfaces. The interface of the sub-module 'Bolster Breaker' is designed with controls that allow the die designer to select the surfaces that are crucial to the parting of the bolster and knit them together to form the final surface termed 'Parting Surface'.

With the generation of the 'Parting Surface', the interface of the sub-module 'Bolster Breaker' is also designed with controls that enable the die designer to easily separate the bolster and derive the ejector and cover cavity inserts. The system is designed such that the cavity inserts are created one at a time and then automatically exported to the main assembly when saved. The cavity inserts will also be mated together automatically in the main assembly.

5.5 Design of 'Core Slide Builder' Module

The 'Core Slide Builder' module is designed to handle the core slide design stage. It is intended to assist the die designer to overcome external undercuts problems by generating core slides. 'Core Slide Designer' is made up of 2 sub-modules – 'Head Design' and 'Body Design'.

5.5.1 Design of 'Head Design' Sub-module

'Head Design' sub-module (Interface shown in Figure 5.9) is designed to enable the die designer to semi-automatically construct the core head. The interface of the sub-module 'Head Design' is designed with controls that allow the die designer to indicate whether the location of the undercut feature is on the cover or ejector cavity insert. Upon selection, the system will activate the part file of the respective cavity insert. The die designer is then required to perform an extruded cut manually to remove the protruding undercut area. The system will then automatically retrieve the 'cut' feature and use it to generate the core head. An instance of the generated core head will be added to the Slide Core listbox. The interface of the sub-module 'Head Design' also contains controls that allow the die designer to delete the core head from the listbox whenever needed. A control to extend the length of the core head is also provided.

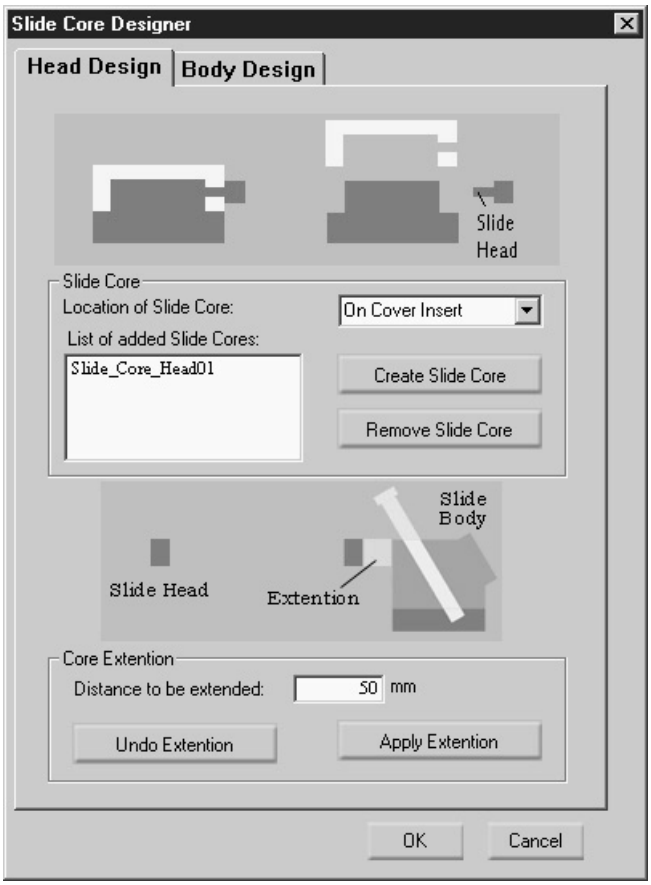


Figure 5.9: Interface of 'Head Design'

5.5.2 Design of 'Body Design' Sub-module

The design of the sub-module 'Body Design' (Interface shown in Figure 5.10) needs to fulfill the following criteria:

1. Allow access to the pre-defined core slide feature library.

2. Enable editing of the core slide before and after adding it to the project.

The interface of the sub-module 'Body Design' includes controls that allow the die designer to freely retrieve slide bodies from the pre-defined core slide feature library and add it to the die project. Once a slide body is added, an instance of the slide body will be added to the core slide body listbox for easy selection. Controls are also designed to delete unwanted slide bodies.

The interface also includes controls that allow the die designer to freely alter the various default parameters of the slide body according to his own preferences before adding the slide body. The same controls can be used even after the slide body had been added.

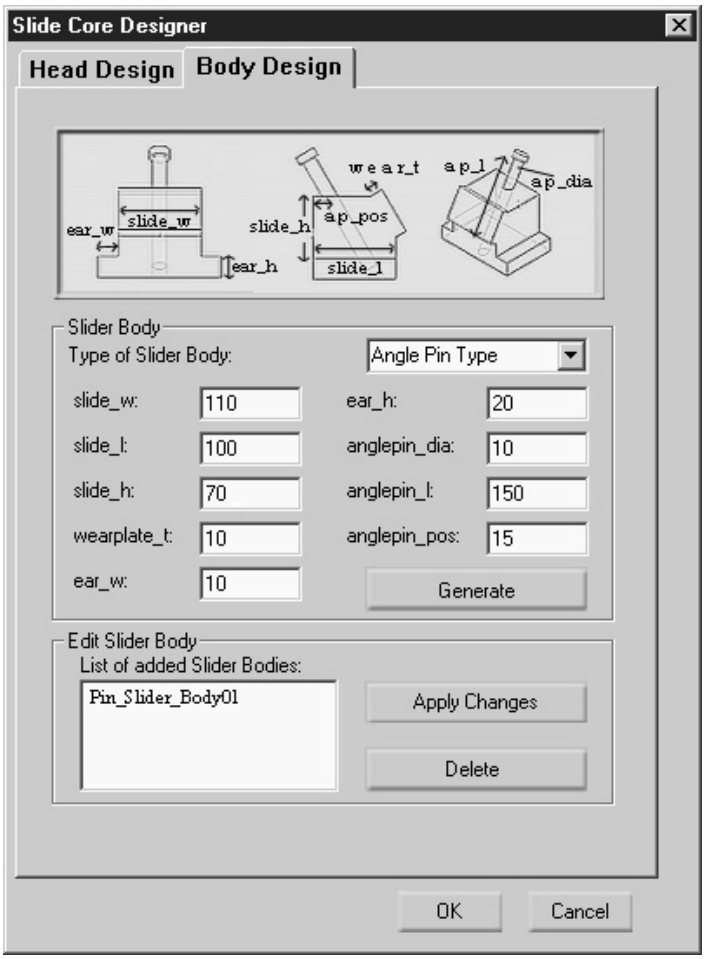


Figure 5.10: Interface of 'Body Design'

5.6 Design of 'Gating System Constructor' Module

'Gating System Constructor' module is designed to handle the layout design stage. As this stage is rather complex, the module is designed with four sub-modules – 'Layout', 'Gates', 'Runners' and 'Overflows', to assist the die designer in planning the layout of cavities, insertion of gates, designing of runners and placement of overflows.

5.6.1 Design of 'Layout' Sub-module

The design of the sub-module 'Layout' (Interface shown in Figure 5.11) needs to fulfill the following criteria:

1. Aids the creation of cavities layout

2. Allows the translation and rotation of individual cavity insert.

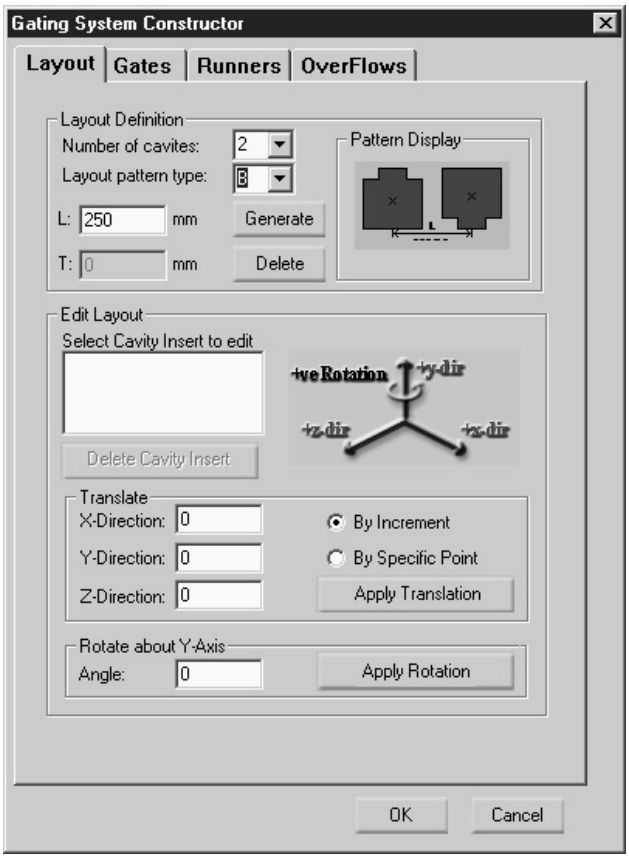


Figure 5.11: Interface of 'Layout'

In the creation of cavities layout, the interface of the sub-module 'Layout' includes controls that allow the die designer to select the number of cavities and their layout pattern types. The different types of cavity layouts provided by the system are shown in Figure 5.12. Once a type of cavity layout is selected and generated, an instance of each cavity inserts will be added to the cavity insert listbox for easy selection.

The interface also includes controls that provide the die designer the option of editing and deleting the individual cavity insert that was added. Through these controls, the die designer can alter the orientation of the cavity inserts through translation and rotation. The system is designed to provide the die designer with two options for the translation of the cavity insert. One option is translating by increments in the X, Y and/or Z direction and the other option is to move the cavity insert to a specific location according to the X, Y, Z coordinates. With regard to cavity insert rotation, the system is designed to only provide rotation about the Y-axis. This is because the default direction of the moving mechanism of die opening in the proposed design system is the Y-axis. Cavity inserts are generally only rotated about the direction of the moving mechanism of die opening.

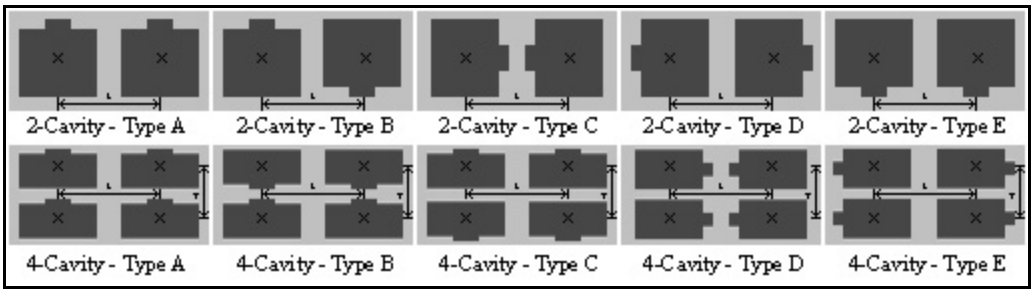


Figure 5.12: Types of cavity layouts

5.6.2 Design of 'Gates' Sub-module

The design of the sub-module 'Gates' (Interface shown in Figure 5.13) needs to fulfill the following criteria:

1. Allow access to the pre-defined gating library.

2. Enable editing of the gate before and after adding it to the project.

3. Allows the translation and rotation of the added gates.

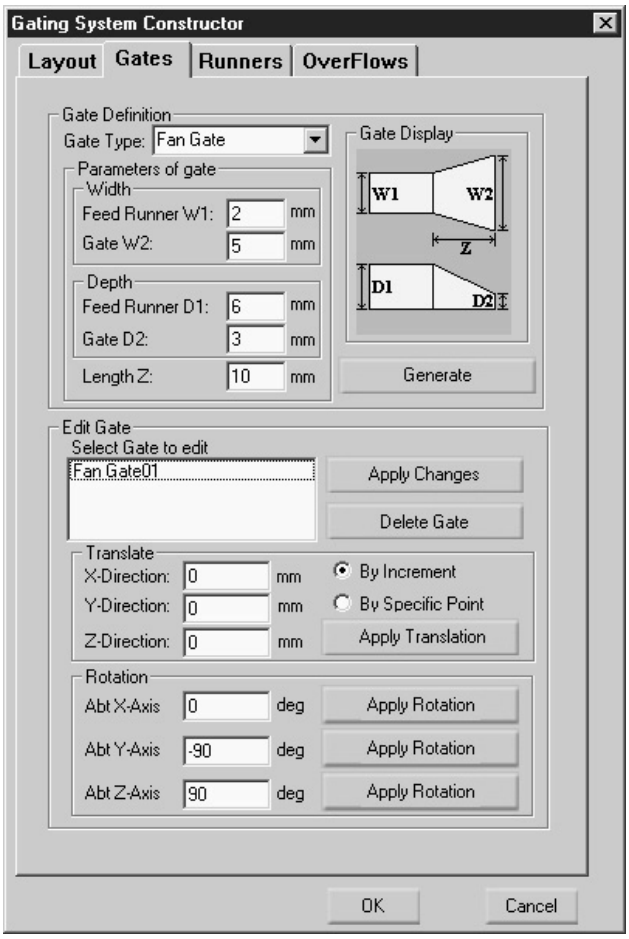


Figure 5.13: Interface of 'Gates'

The interface of the sub-module 'Gates' includes controls that allow the die designer to freely retrieve gate features from the pre-defined gating feature library and add it to the die project. The different types of gates are shown in Figure 5.14. Before the gate is added, the die designer needs to specify its location by sketching a sketch line on one of the cavity insert. The placement of the gate is similar to the placement of overflows as discussed in section 4.5.1. For multiple cavity-layout, the system is designed to automatically copy the newly added gate to the same location on all the other cavity inserts. Once a gate is generated, an instance of the gate will be added to the gate listbox for easy selection. Controls are also designed to delete unwanted gates.

The interface also includes controls that allow the die designer to freely alter the various default parameters of the gate according to his own preferences before adding

the gate. The same controls can be used even after the gate had been added.

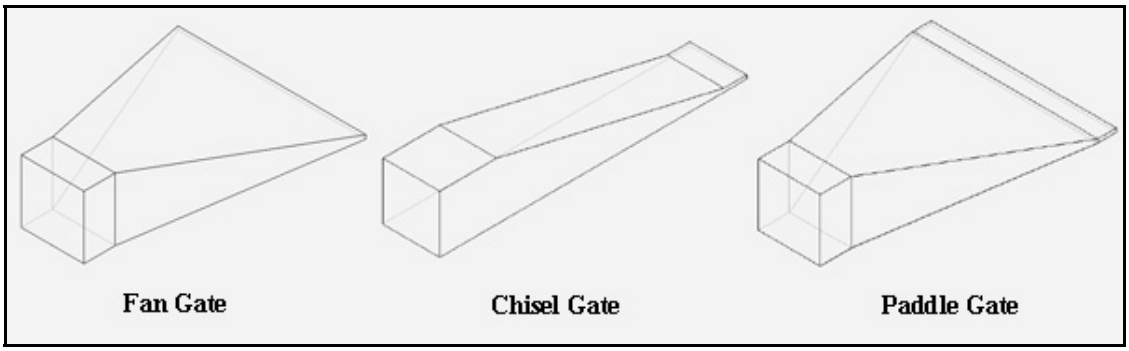


Figure 5.14: Types of gates

Similar to cavity insert, the system is also designed to allow the die designer to alter the orientation of the gate through translation and rotation. The translation of gate is similar to the translation of cavity insert. However unlike the rotation of cavity insert, the system provides rotation about the X-axis, Y-axis and Z-Axis for the rotation of gate. This is because the rotation of gate is generally independent of the direction of the moving mechanism of die opening.

5.6.3 Design of 'Runners' Sub-module

The design of the sub-module 'Runners' (Interface shown in Figure 5.15) aims to aid the die designer to plan the route of the runners linking the different cavities and needs to fulfill the following criteria:

1. Allow access to the pre-defined gating library.

2. Enable editing of the runner before and after adding it to the project.

3. Allows the translation and rotation of the added runners.

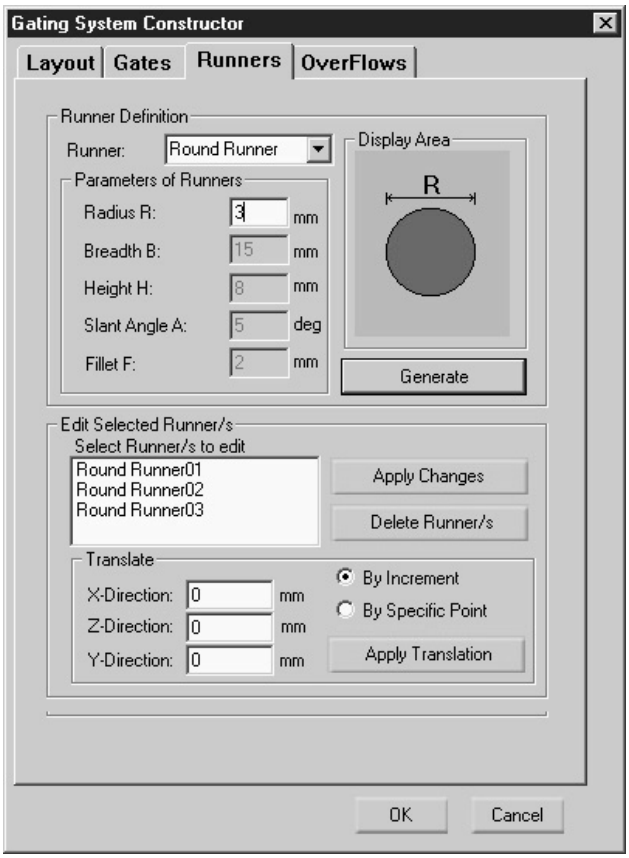


Figure 5.15: Interface of 'Runners'

The interface of the sub-module 'Runners' includes controls that allow the die designer to freely retrieve runner features from the pre-defined gating feature library and add it to the die project. The different types of runners are shown in Figure 5.16. The system is designed in such a way that upon selection of a runner type, the cross-sectional parameters of the selected runner type are displayed. The system also allows the die designer to alter the various cross-sectional parameters according to his own preferences before adding the runner. The same controls can be used even after the runner had been added. The placement of the runner is also by specifying a sketch line. The length of the runner is the same as the specified sketch line. Once a runner is generated, an instance of each runner will be added to the runner listbox. Controls are also designed to delete unwanted runners.

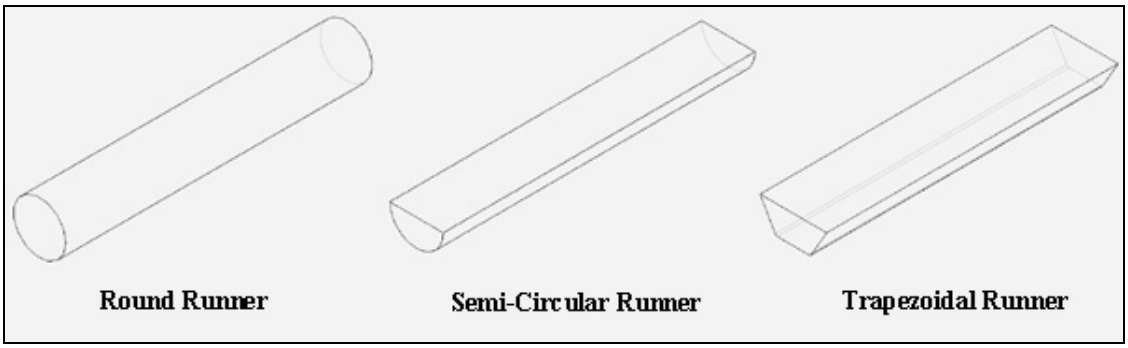


Figure 5.16: Examples of runners

Unlike the design of the cavity insert and the gate, the system is designed to only allow the die designer to alter the orientation of the runner through translation. When translating a runner, the die designer also has two options of translating - by increments in the X, Y and/or Z direction and by specific location according to the X, Y, Z coordinates. The runner cannot be rotated because runners are generally inter-connected and rotating individual runner will affect the entire runner system.

5.6.4 Design of 'Overflows' Sub-module

The design of the sub-module 'Overflows' (Interface shown in Figure 5.17) aims to aid the die designer to place overflows for the cavities and needs to fulfill the following criteria:

1. Allow access to the pre-defined gating library.

2. Enable editing of the overflow before and after adding it to the project.

3. Allows the translation and rotation of the added overflows.

The interface of the sub-module 'Overflows' includes controls that allow the die designer to freely retrieve overflow features from the pre-defined gating feature library and add it to the die project. The system presently provides one standard type of overflow as shown in Figure 5.18. The placement of overflow is illustrated in section

4.5.1. Similar to the addition of gate, the system first adds an overflow to one of the cavity insert. The system then ensures that the other cavity inserts in the cavity layout (if there are more than one cavity inserts) will automatically have a similar overflow added to the same location. Once an overflow is generated, an instance of each overflow will be added to the overflow listbox for easy selection. Controls are also designed to delete any unwanted overflows.

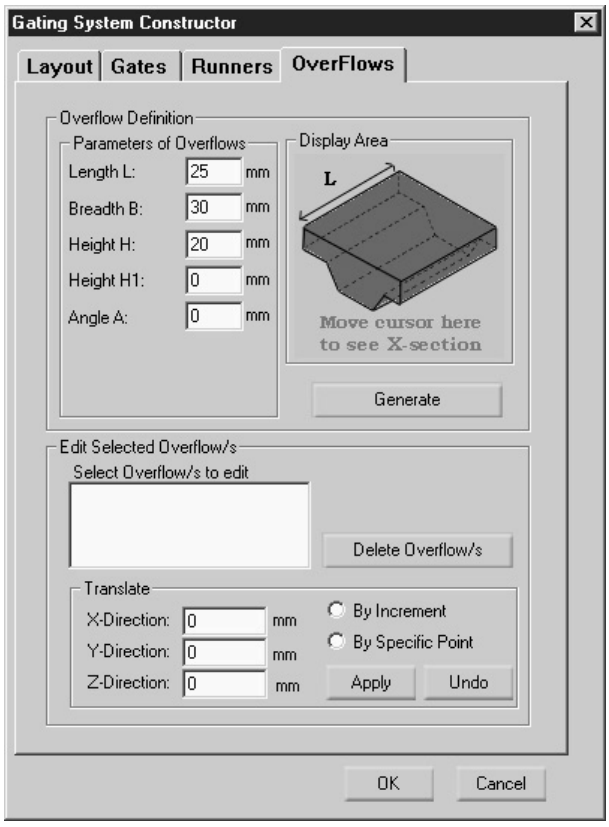


Figure 5.17: Interface of 'Overflows'

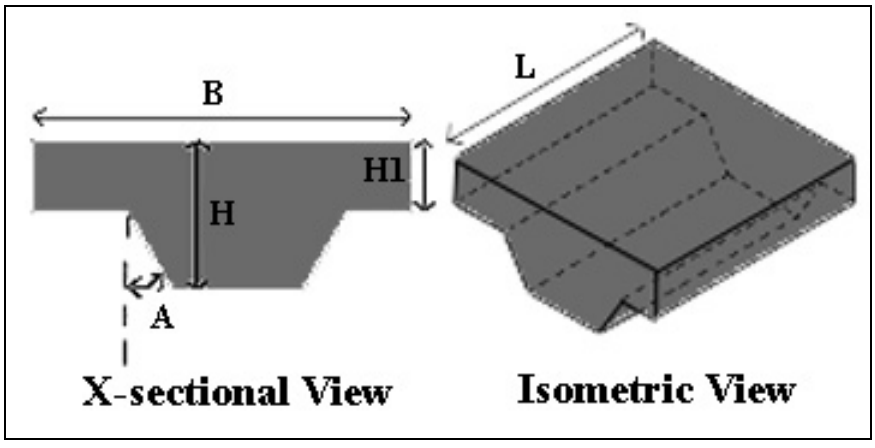


Figure 5.18: Standard overflow

The interface also includes controls that allow the die designer to customise the various default parameters of the overflow according to his own preferences before adding the overflow. The same controls can be used even after the overflow had been added.

The system is also designed to translate the overflow in the same way as the cavity insert, gate and runner. There is no rotation function for the overflow as it is generally not necessary.

5.7 Design of 'Die Base Designer' Module

'Die Base Designer' module is designed to handle the die base design stage. This module is designed with three sub-modules – 'Load N Configure', 'Pins N Screws' and 'Thickness', to assist the die designer in the addition of a die base to the die project.

5.7.1 Design of 'Load N Configure' Sub-module

The design of the sub-module 'Load N Configure' (Figure 5.19) needs to fulfill the following criteria:

1. Allow access to the pre-defined die base library.

2. Provides selection of the different configurations of the die base.

The interface of the sub-module 'Load N Configure' includes controls that allow the die designer to retrieve standard die bases from the pre-defined die base feature library. The die bases are modeled according to commercial vendor's catalogue and contain the standard components. An exploded view of the DME series D die base is shown in Figure 5.20.

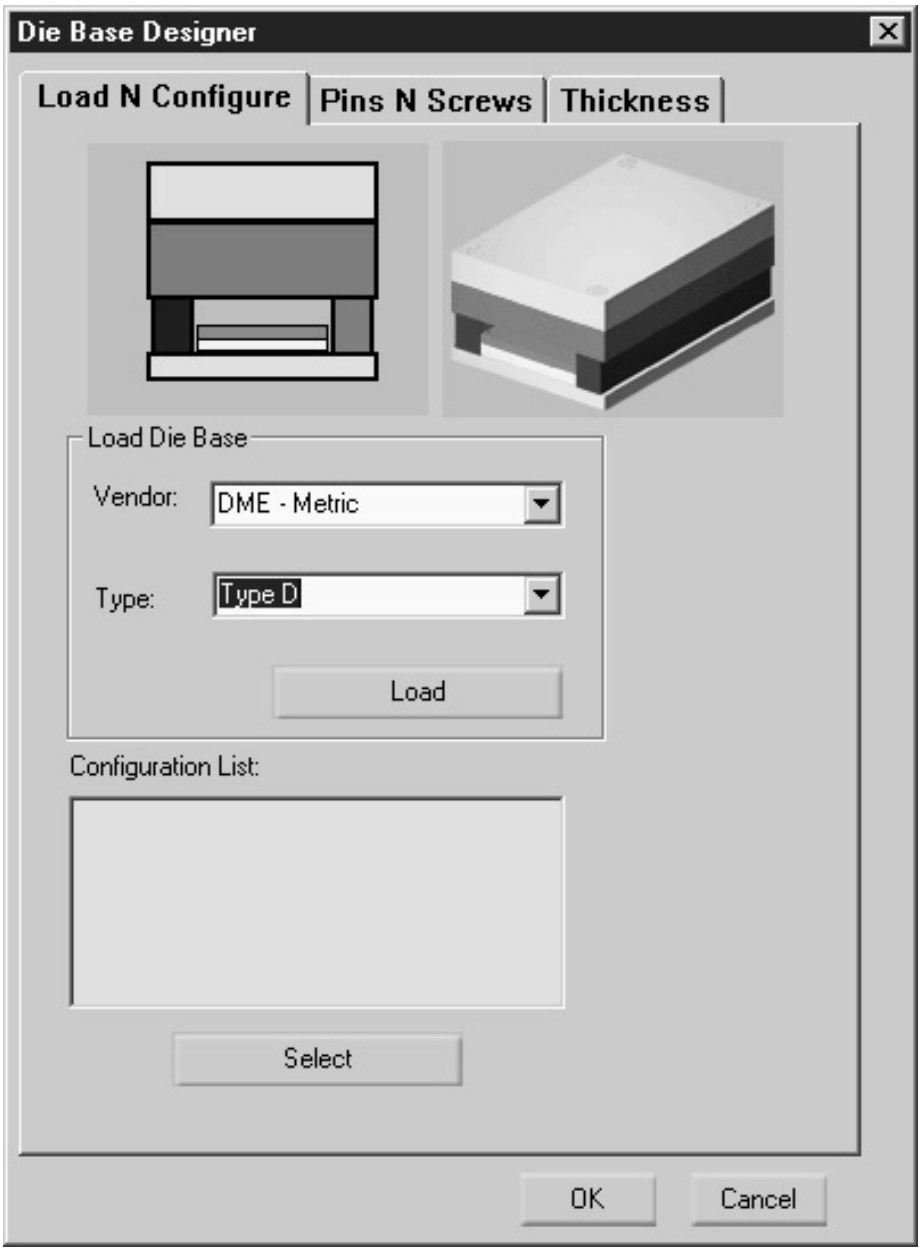


Figure 5.19: Interface of 'Load N Configure'

The system is designed in such a way that the die base is loaded initially with the default configuration and a complete list of configurations available for the die base is displayed in the configuration listbox for the die designer. If the standard configurations of the available die bases did not meet the die designer's requirements, the system then provides the other two sub-modules that assist the die designer to customize a die base to meet his specific requirements.

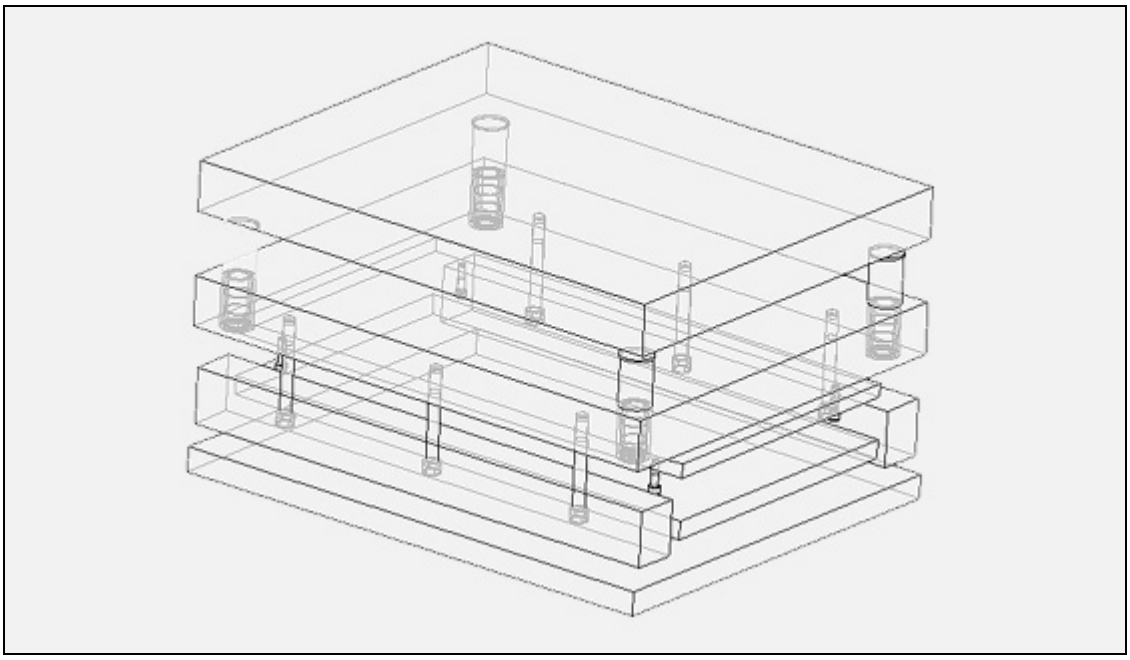


Figure 5.20: Exploded view of DME die base series D

5.7.2 Design of 'Pins N Screws' Sub-module

In the 'Pins N Screws' sub-module (Interface shown in Figure 5.21), controls are designed to help the die designer to edit the position of the guide pins, ejector, top and bottom screws in the die base.

The positions of these pins and screws are referenced to the central axis. The controls allow the die designer to adjust the horizontal or the vertical distance of these pins and screws from the central axis. As the pins and screws are arranged in a pattern, any modifications to one screw/pin will alter the other screws/pins in the pattern. For example as shown in Figure 5.22, the original positions of the guide pins are shown in the diagram on the left. When the die designer changes the distance l_1 to l_2, all the four guide pins will move accordingly as shown in the diagram on the right.

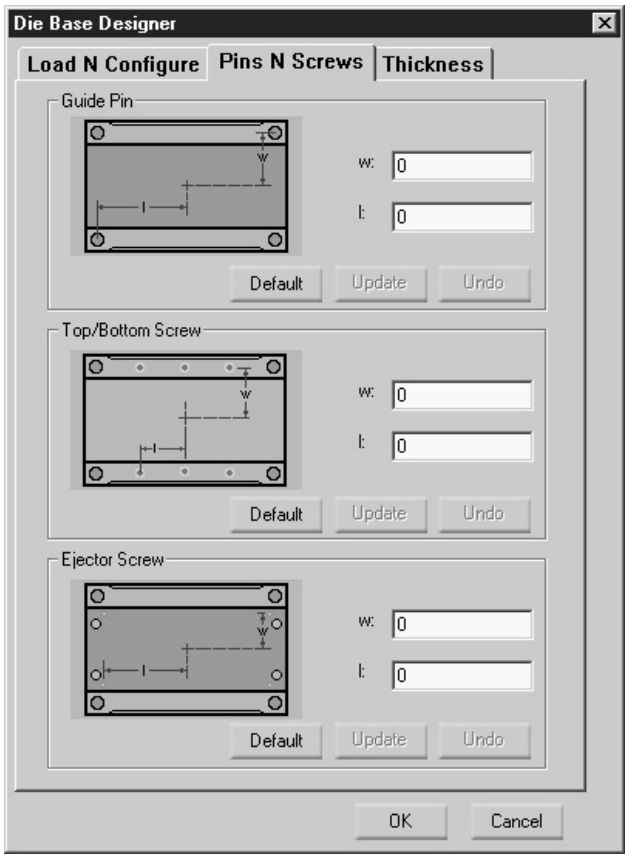


Figure 5.21: Interface of 'Pins N Screws'

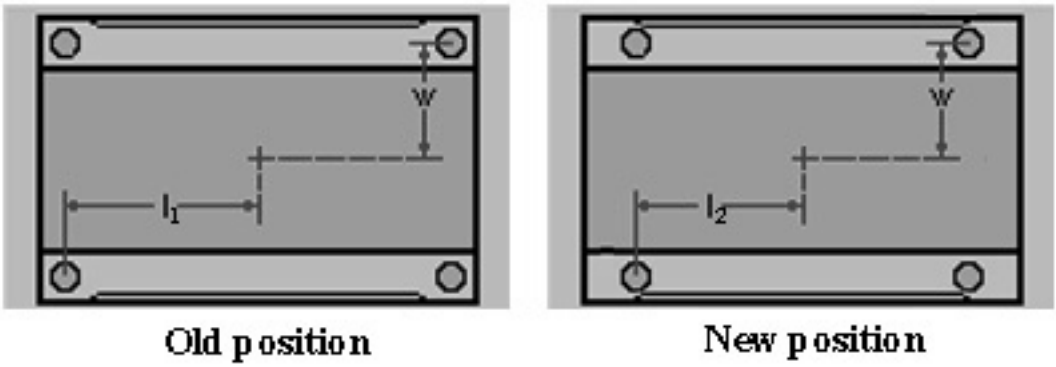


Figure 5.22: Example of a change in the *l* parameter of guide pin

5.7.3 Design of 'Thickness' Sub-module

The 'Thickness' sub-module (Interface shown in Figure 5.23) enables the editing of the thickness of various plates in the die base. All the standard die bases are modeled with additional plates that have default zero thickness. These plates are labeled as A1, A2, A3, B1 and B2. Hence the system allows the die designer to add additional plates to the standard die base by simply assigning a finite thickness to these plates. The

height of the various pins and screws placed inside the various plates and blocks will also be altered accordingly.

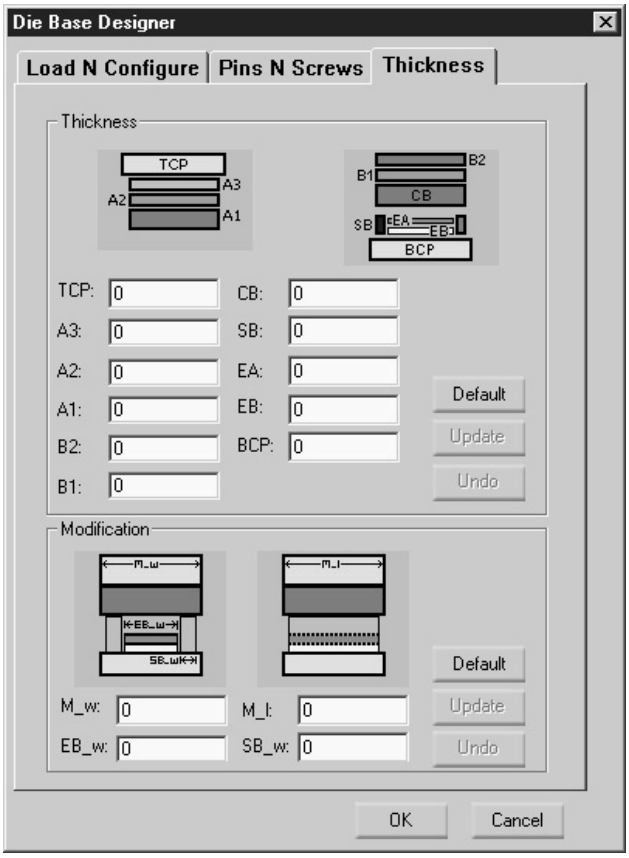


Figure 5.23: Interface of 'Thickness'

This sub-module enables the die designer to modify the length and width of the standard plates such as top clamp plate and bottom clamp plate, the width of the ejector block and the width of the space block.

5.8 Design of 'Ejector System Constructor' Module

The 'Ejector System Constructor' module is designed to handle the ejector system design stage. This module (Interface shown in Figure 5.24) assists the die designer in the design of ejector system and has to fulfill the following criteria:

1. Allow access to the pre-defined ejector system feature library.

2. Enable editing of the ejector pins before and after adding it to the project.

3. Allows the translation of the added ejector pins.

4. Provides function to trim the ejector pins so that the ejector pins will be flushed with the 'Parting Surface'.

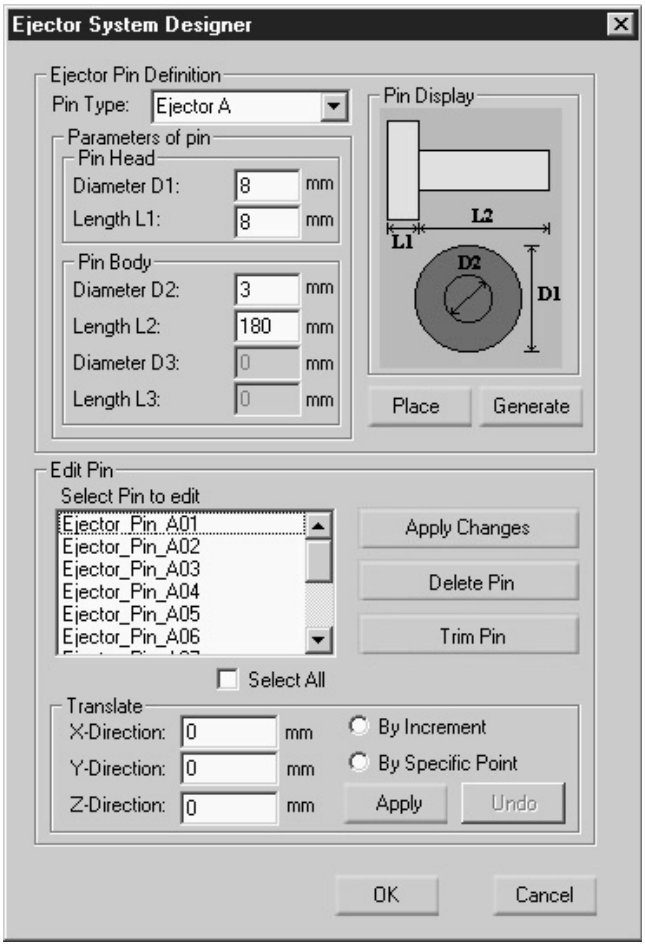


Figure 5.24: Interface of 'Ejector System Constructor'

The interface of the module 'Ejector System Constructor' enables the die designer to retrieve ejector pin features from the pre-defined ejector pin feature library and add it to the die project. As the ejector pins are generally placed on the ejector plate, the system automatically creates a sketch on the face of the ejector plate of the die base. The die designer only needs to create a sketch point to indicate the position of the ejector pin. As the ejector pin is modeled with a reference point, the ejector pin is

placed on the cavity insert by coinciding the reference point with the specified sketch point. If there is more than one cavity insert in the cavity layout, the system will automatically duplicates a similar ejector pin to the same location on the other cavity inserts. The placement of the ejector pins is similar to the placement of the overflows. Hence all the ejector pins are automatically placed on the ejector plate of the die base. Once an ejector pin is generated, an instance of the ejector pin will be added to the ejector pin listbox for easy selection. Controls are also designed to delete any unwanted ejector pins.

The interface also includes controls that allow the die designer to alter the various default parameters of the ejector pins according to his own preferences before adding the ejector pins. The same controls can also be used even after the ejector pins had been added.

With regard to the translation of the ejector pins, the system is designed such that the translation is restricted to perimeter of the ejector plate. The vertical position of the ejector pins cannot be altered. There is also no rotation function for the ejector pins as it is generally not necessary.

When the positions of the ejector pins had been confirmed, the system enables the die designer to automatically trim the ejector pins according to the ‘Parting Surface’. Before trimming the ejector pins, the die designer needs to ensure that the length of the ejector pins exceeds the ‘Parting Surface’. After the trimming, the ejector pins will be flushed with the ‘Parting Surface’.

5.9 Design of ‘Cooling System Designer’ Module

The ‘Cooling System Designer’ (Interface shown in Figure 5.25) module is designed

to handle the cooling system design stage.

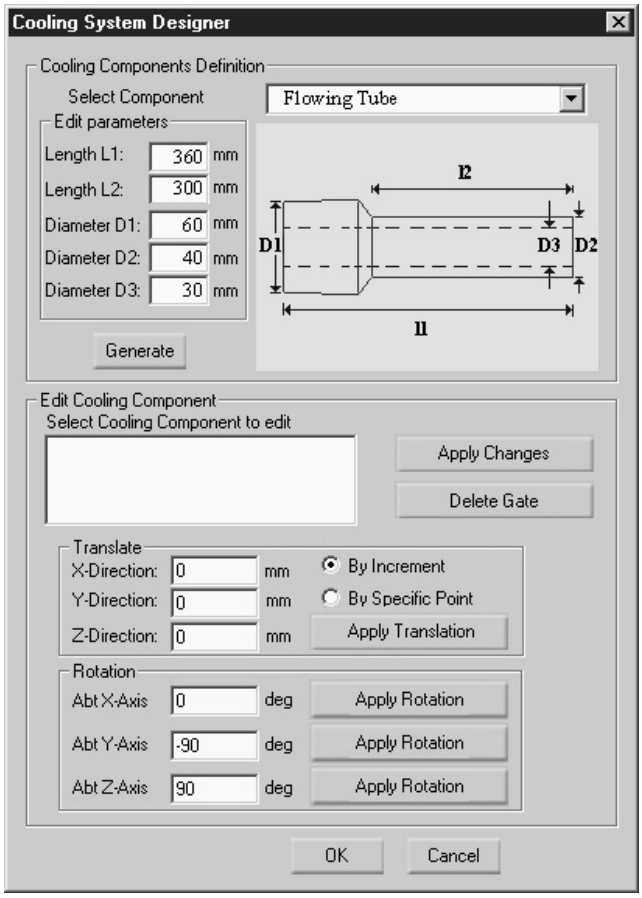


Figure 5.25: Interface of 'Cooling System Designer'

This module enables the die designer to add the cooling features to the die project, thus it must fulfill the following criteria:

1. Allow access to the pre-defined cooling system feature library.

2. Enable editing of the cooling features before and after adding it to the project.

3. Allows the editing of the orientation of the added cooling features.

The interface of the module 'Cooling System Designer' includes controls that allow the die designer to retrieve cooling features from the pre-defined cooling feature library and add it to the die project. The various types of cooling components as modeled from Procomps [37] catalogue, are shown in Figure 5.26. The placement of

the cooling components is similar to the placement of overflows as discussed in section 4.5.1. Once a cooling component is generated, an instance of each cooling component will be added to the cooling component listbox for easy selection. Controls are also designed to delete any unwanted cooling features.

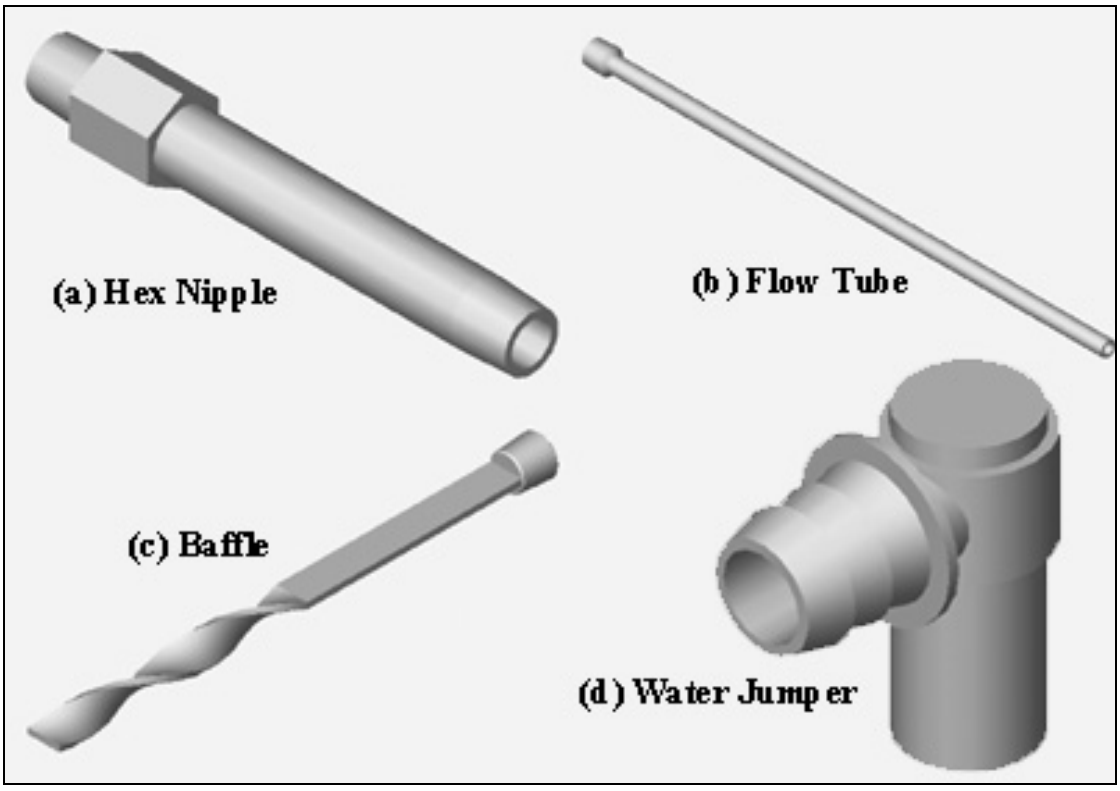


Figure 5.26: Examples of cooling components

The interface also includes controls that allow the die designer to alter the various default parameters of the cooling features according to his own preferences before adding the cooling features. The same controls can be used even after the cooling features had been added.

Regarding the editing of the orientation of the added cooling features, the system is designed such that the die designer can alter the orientation of the cooling features through translation and rotation in the same way as the other die casting components.

5.10 Design of 'Standard Components' Module

The 'Standard Components' (Interface shown in Figure 5.27) module is created as a supplementary module to cater to the addition of standard components to the die project.

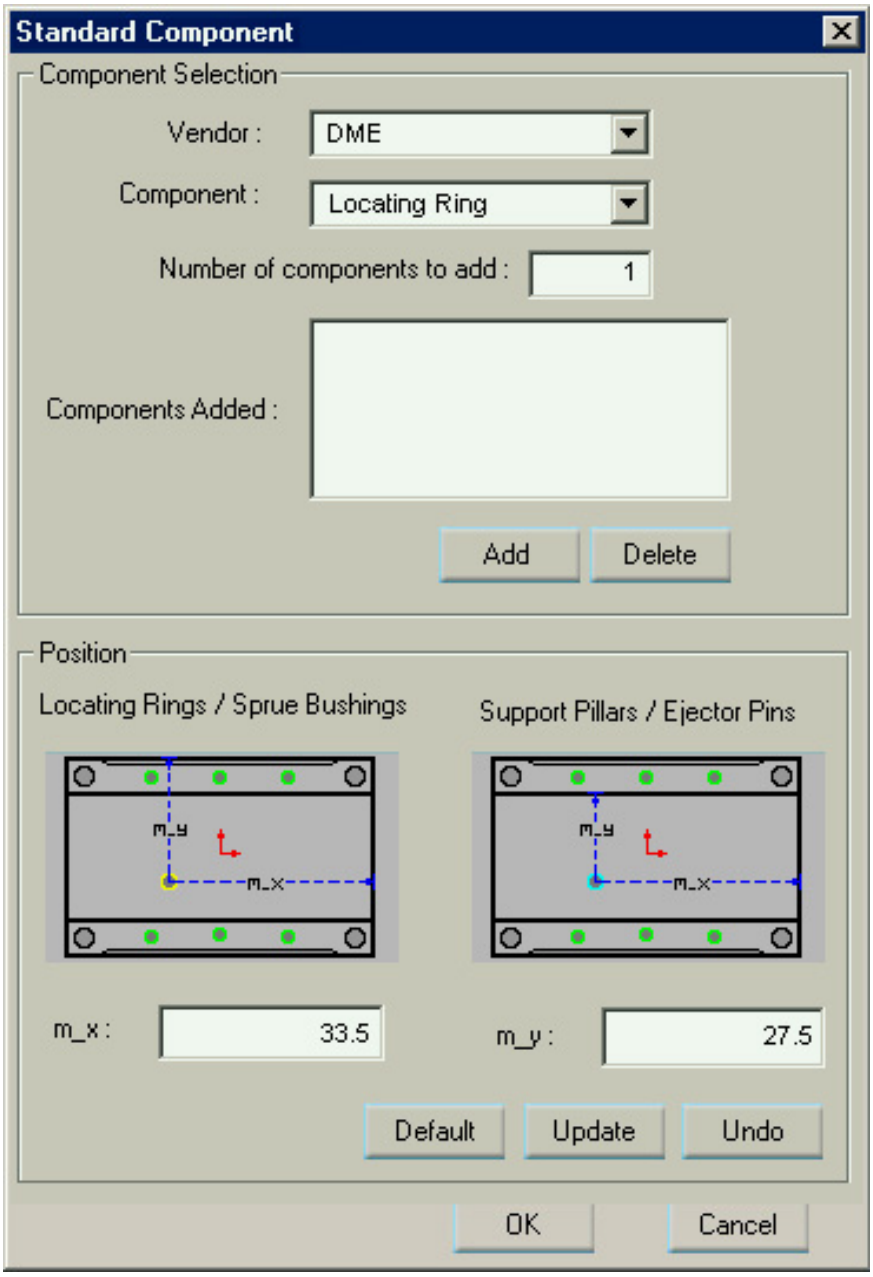


Figure 5.27: Interface of 'Standard Components'

The design of this module enables the die designer to add standard components used in die casting to the die project and must fulfill the following criteria:

1. Allows access to the pre-defined standard component feature library.

2. Enables editing of the standard component before and after adding it to the project.

3. Allows the editing of the orientation of the added standard components.

The interface of the module 'Standard Components' includes controls that allow the die designer to retrieve standard components from the pre-defined standard component feature library and add them to the die project. Presently, the components database includes components like locating rings, sprues and support pillars as shown in Figure 5.28. These components come in different configurations as indicated in their respective catalogs. The placement of the standard component features is similar to the placement of the other die casting features. Once a standard component is generated, an instance of each standard component will be added to the standard component listbox for easy selection. Controls are also designed to delete any unwanted standard component features.

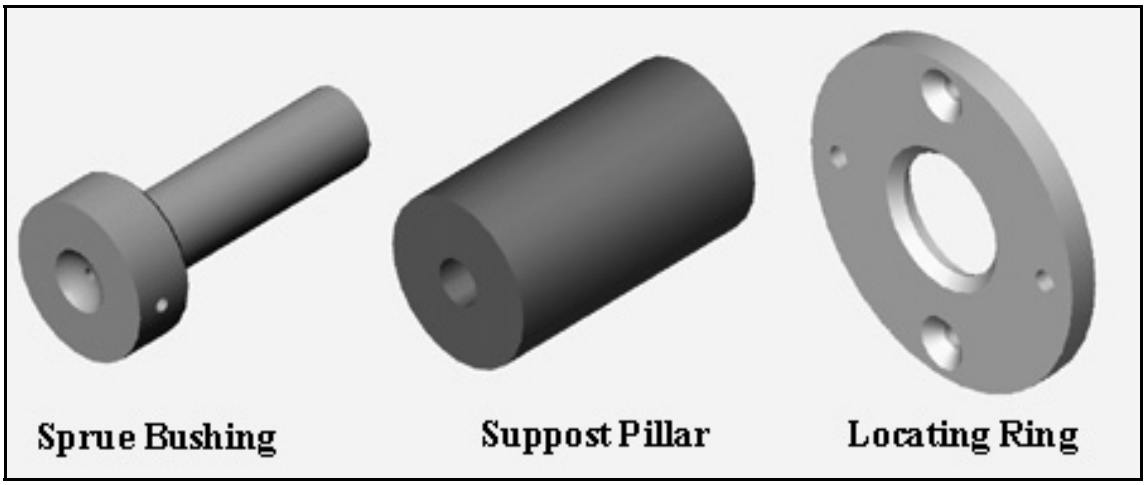


Figure 5.28: Examples of standard components

Regarding the editing of the orientation of the added standard component features, the system is designed in such a way that the die designer can alter the orientation of the

standard components through translation and rotation in the same way as the other die casting components.

CHAPTER 6 : SYSTEM IMPLEMENTATION & CASE STUDIES

In this chapter, the implementation of the prototype die casting die design system is introduced. Case studies are then performed to demonstrate how the prototype system can reduce the lead time of die casting die design. Discussions on the merits of the die design system are presented at the end of the chapter.

6.1 System Implementation

A prototype die design system is developed on the platform of the commercial SolidWorks 2001 CAD system under Windows Millennium environment. The development tools used include SolidWorks API, MFC and Visual C++. The prototype system consists of eight design modules. Standard die casting feature libraries are also built and added to the design modules. The complete system architecture of the die design system for die casting is shown in Figure 5.2.

6.2 Case Studies

In order to illustrate the capability of the developed die design system for die casting, case studies were carried out using two different product models. The first product model, a push button housing which contains an undercut will be discussed in details. The other product model, motor housing, which do not contain any undercuts, will be briefly studied.

6.2.1 Case Study A: Push Button Housing

In this case study, the product model used is a push button housing for pedestrian push buttons. The material used is aluminum with space for a sign placard. It is selected for study because (i) it contains through holes and (ii) it contains undercut features. The push button housing is shown in Figure 6.1.

The die casting die design is to include the following:

- 2-cavity layout of cavity inserts
- A movable core slide to cater for the undercut feature
- A complete gating system
- A die base
- An ejector system
- A cooling system

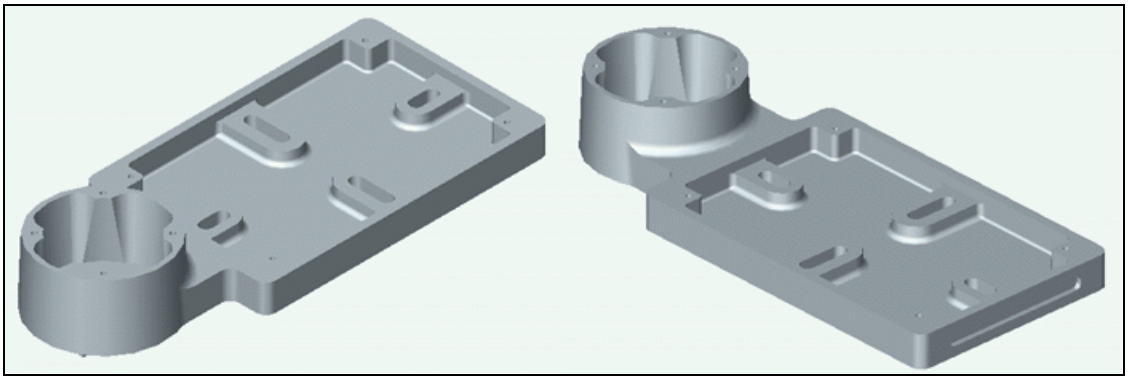


Figure 6.1: Push button housing

In this case study, the die design starts from the very beginning where the New Project sub-module is used to load in the product model. At the same time, a working directory is created and the measuring unit is set as millimeters. The die project is also

set to scale the product model based on the centroid.

Using the module ‘Cavity Insert Builder’, the cavity inserts (ejector and cover) can be easily constructed. The first phase of the construction of the cavity inserts is to design the cavity block and this is done through ‘Bolster Breaker’ sub-module. After the default containing box is generated which contains the product model, the containing box size is enlarged and set at 360mm by 190mm by 125mm. Using ‘Parting Line Selector’ sub-module, the ‘Parting Line’ is easily selected as illustrated in section 4.2.1 and a radiated surface of dimension 80mm is created. The surfaces are then knitted together as illustrated in section 4.2.2 using the sub-module ‘Parting Face Generator’. The holes on the ‘Parting Face’ are then patched to form the ‘Parting Surface’ as illustrated in section 4.2.3. The sub-module ‘Bolster Breaker’ is then called upon to create the ‘Parting Surface’, which is made up of the radiated surface, parting faces and the ‘patched’ faces. Once created, the parting surface is then used to split the containing box into the respective ejector and cover cavity inserts. The splitting up process is also performed through the sub-module ‘Bolster Breaker’. The ejector and cover cavity inserts built using ‘Cavity Insert Builder’ are shown in Figure 6.2(a) and 6.2(b) respectively.

Since this product model has an external undercut, a core slide is needed. Using ‘Core Slide Designer’ module, the die designer can design the necessary core slide for the undercut. The design of the core slide is divided into two parts. The first part is the design of the core head, which is accomplished through ‘Core Slide Designer’ sub-module ‘Head Design’. The second part is to create a slide body for the slide head. The ‘Core Slide Designer’ sub-module ‘Body Design’ is used to create the required side body.

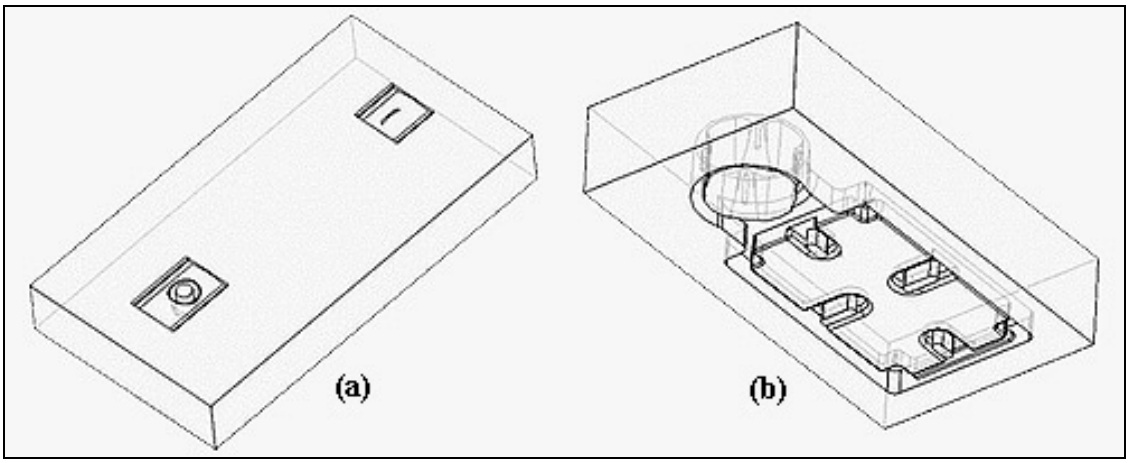


Figure 6.2: Cavity inserts for push button (a) Ejector cavity insert (b) Cover cavity insert

Using the sub-module 'Head Design', the die designer begins with the selection of whether the location of the core is on the ejector or cover cavity insert. In this case study, it is on the cover cavity insert as shown in Figure 6.3(a). The cover cavity insert part file will then be automatically opened as the active window upon its selection. The die designer then performs an extruded cut on the protruding undercut area manually, to remove the protruding portion. After the die designer had accomplished the extruded cut, the die design system will automatically generate the core slide head as shown in Figure 6.3(b). An extension of 30mm is also added to the core head.

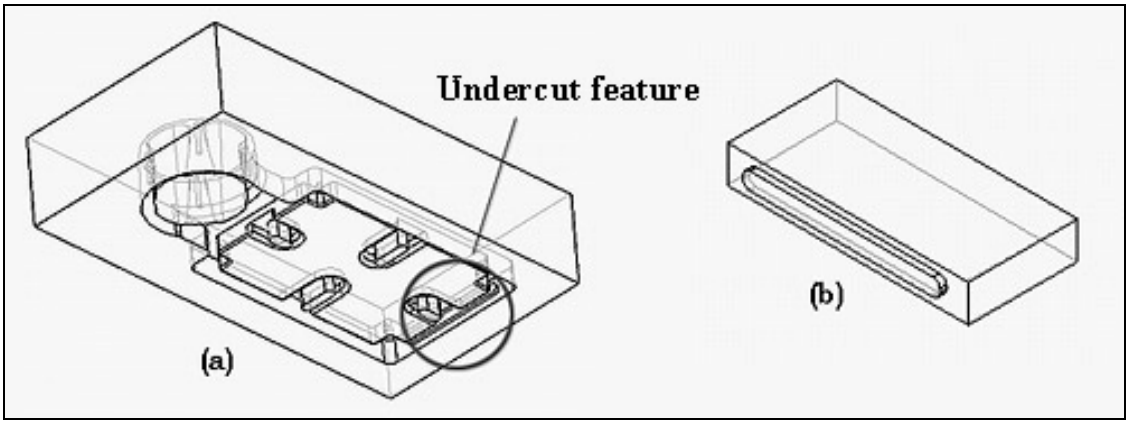


Figure 6.3: (a) Undercut feature of push button (b) Core slide head

This core slide head is then added to a standard core slide body through the sub-module 'Body Design'. The system is inbuilt with feature libraries that contain

predefined standard core slide bodies that can be retrieved through 'Body Design' sub-module. The core slide body selected for this case study is of the angle pin type. Upon addition, the core slide body will be automatically mated with the core head. Figure 6.4 shows the complete cavity inserts with core slide mechanism.

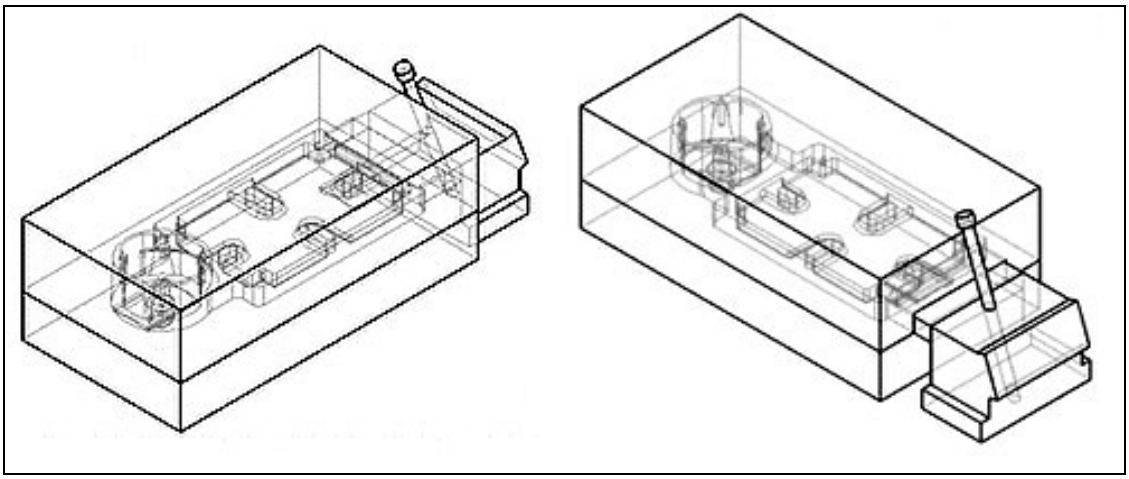


Figure 6.4: Cavity inserts for push button with core slide mechanism

Upon completion of the cavity inserts and the addition of a core slide, the die designer uses the module 'Gating System Constructor' to design the gating system for the pair of cavity inserts. The first step is to design a cavity layout for the cavity inserts. A 2-cavity layout was chosen. The cavity layout was created by 'Layout' sub-module and the distance between the cavity inserts is set as 500mm as shown in Figure 6.5.

Next, suitable gates were added to the cavity inserts. The exact procedure of the addition of gating components has been illustrated in section 5.6. The type of gate selected for this case study is a fan gate. Subsequently runners were added and the runner type utilized in this case study was a semi-circular runner. The runners were designed to follow a Y-shaped path. The final step is to include overflows. In this case study, two overflows were added. The gating system created is shown in Figure 6.6.

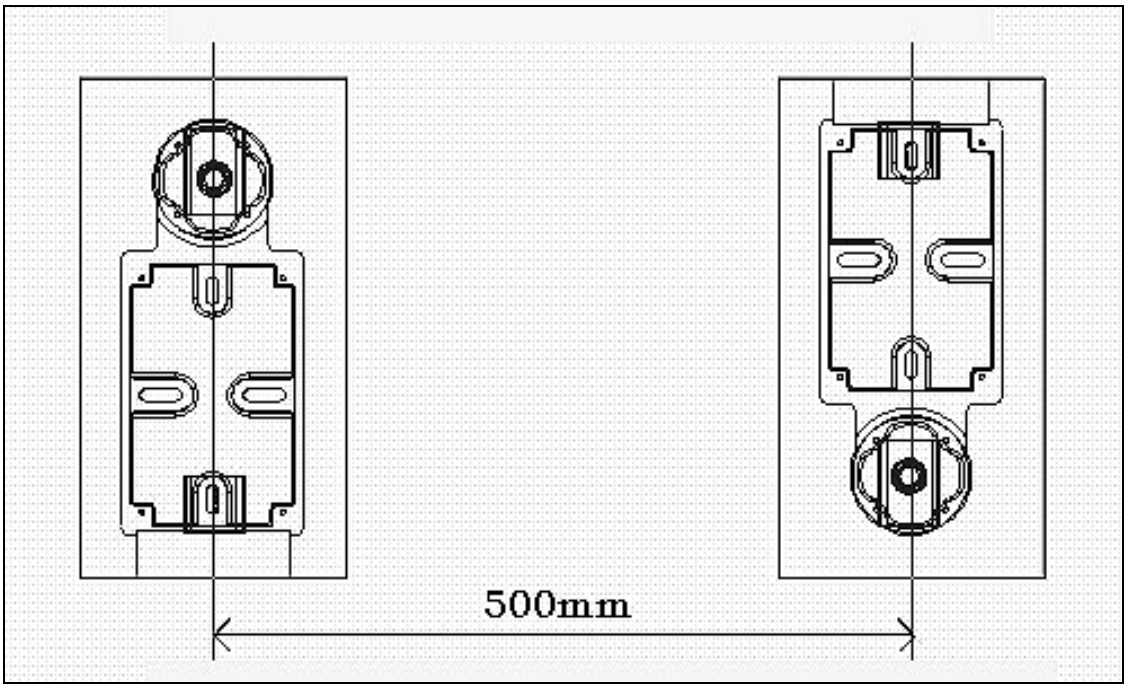


Figure 6.5: Cavity layout for push button with distance 500mm apart

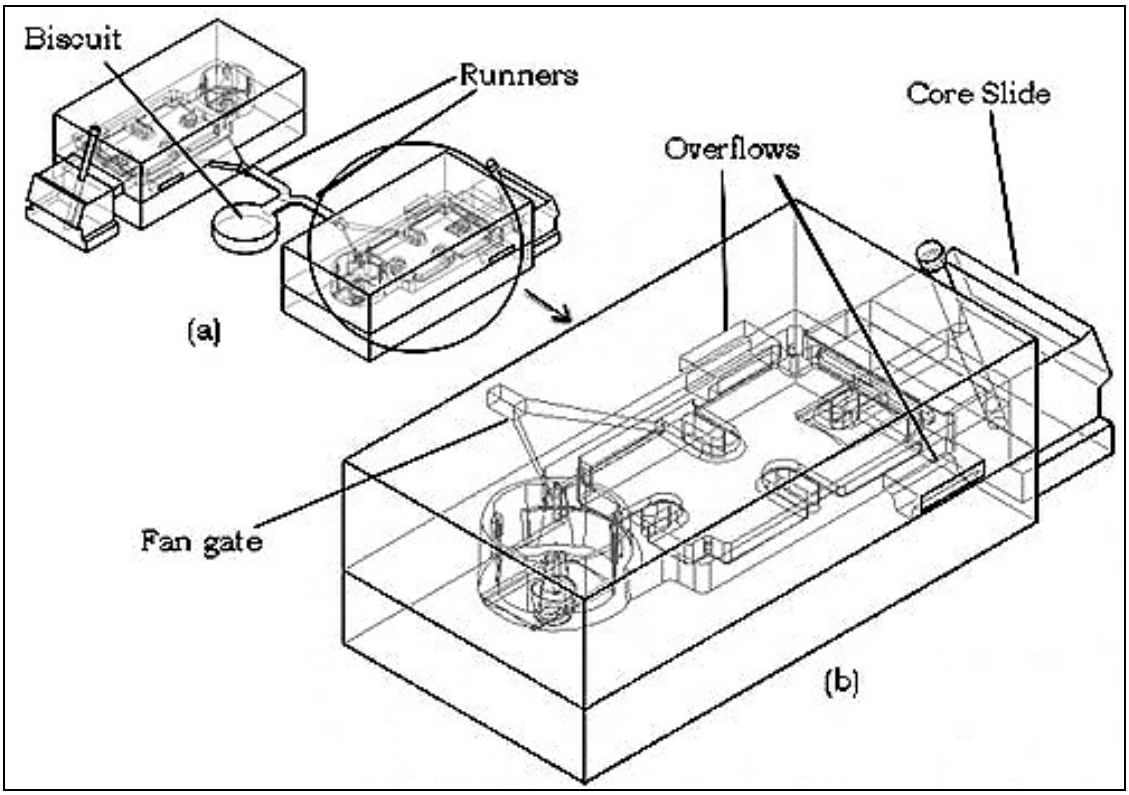


Figure 6.6: (a) Gating system added for push button (b) Enlarged view of an cavity insert

Next, a die base was added to contain the cavity inserts using the module 'Die Base Designer'. The die base used in this case study is of the vendor DME series D. The size of the die base added is 1000 mm by 800 mm by 344 mm.

Then the die designer utilized the 'Ejector System Constructor' module to add the ejector system for the die base. In this case study, ten ejector pins were added. Eight of these pins are typical ejector pins as shown in Figure 6.7(a) while the remaining two are reinforced ejector pins as shown in Figure 6.7(b). These two ejector pins are modeled according to the DME catalogue. After the positions of the ejector pins had been finalized, they are trimmed according to the 'Parting Surface' as shown in Figure 6.8.

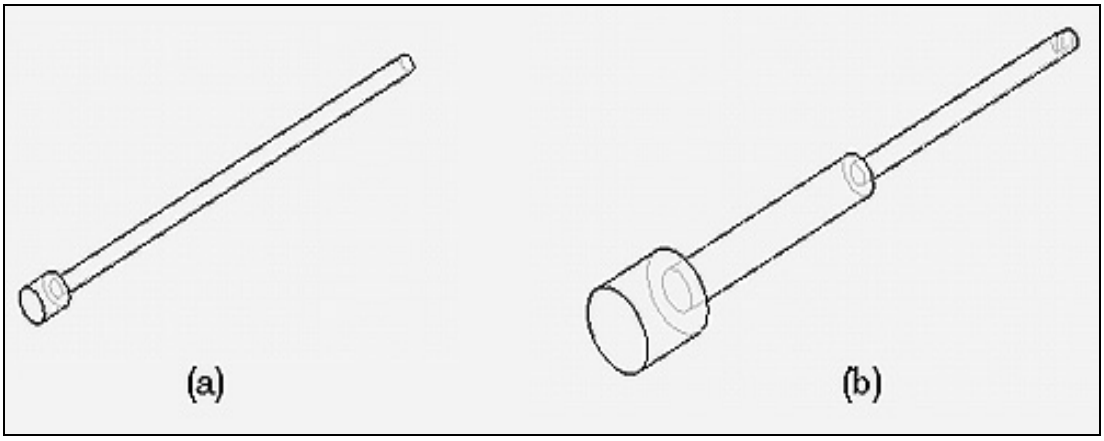


Figure 6.7: (a) Typical ejector pin (b) Reinforced ejector pin

With the addition of the ejector system, the die designer had successfully created a die design consisting of a 2-cavity layout of cavity inserts, core slides, gating system, die base and ejector system as shown in Figure 6.9.

Next, the cooling tubes were designed through the module 'Cooling System Designer'. Seven identical flow tubes were added to the top of the cavity inserts. The inner diameter and length of the flow tubes is 10 mm and 1040 mm respectively. The flow tubes were modeled according to the Procomps catalogue as shown in Figure 6.10.

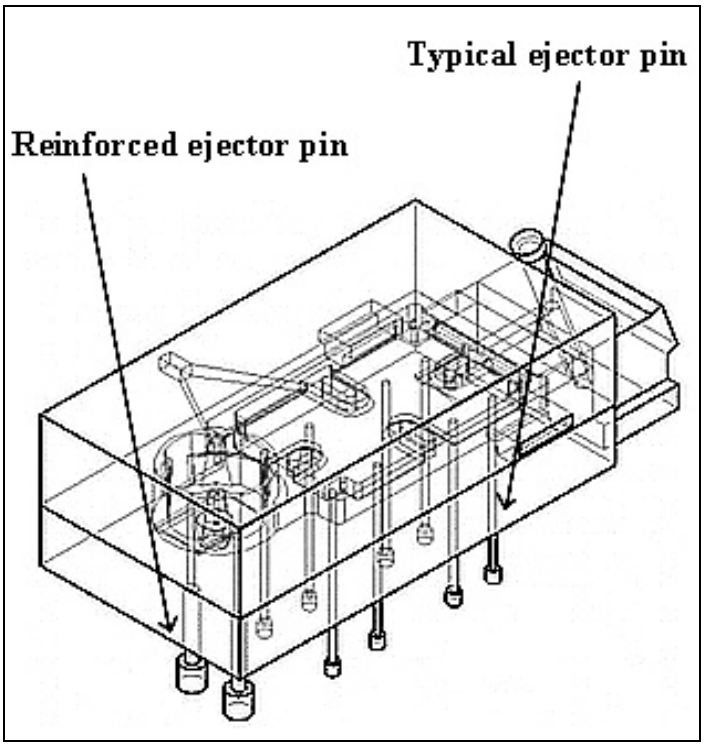


Figure 6.8: Cavity insert with ejector system for push button

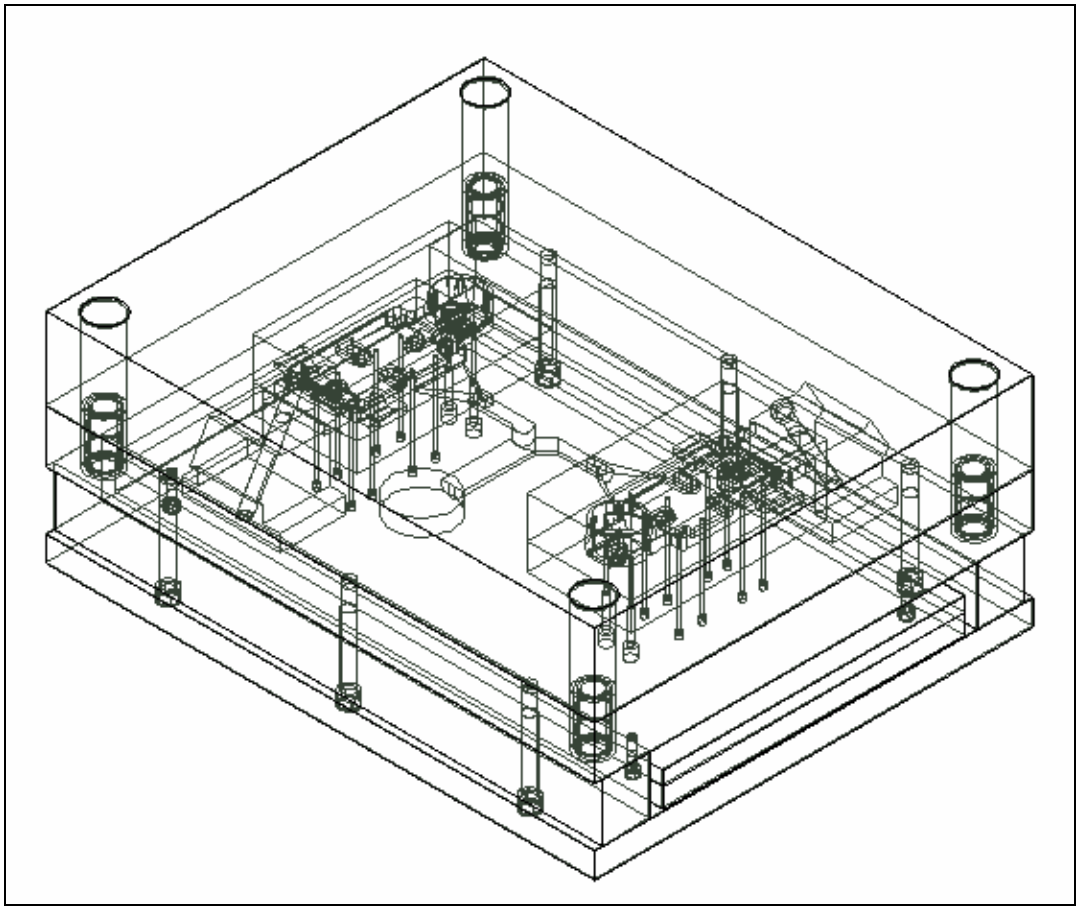


Figure 6.9: Cavity inserts, slides, die base, gating and ejector systems for push button

The flow tubes are at equal intervals of 50 mm apart and arranged to be parallel to the longer side of the die base so as to prevent interfering with the core slide mechanism. A portion of 20 mm of the flow tubes is extended out of the die base so that water hose or jumpers can be connected to the end of the flow tubes. There were no cooling channels placed below the cavity inserts because the flow of the molten metal is concentrated on the cover cavity insert.

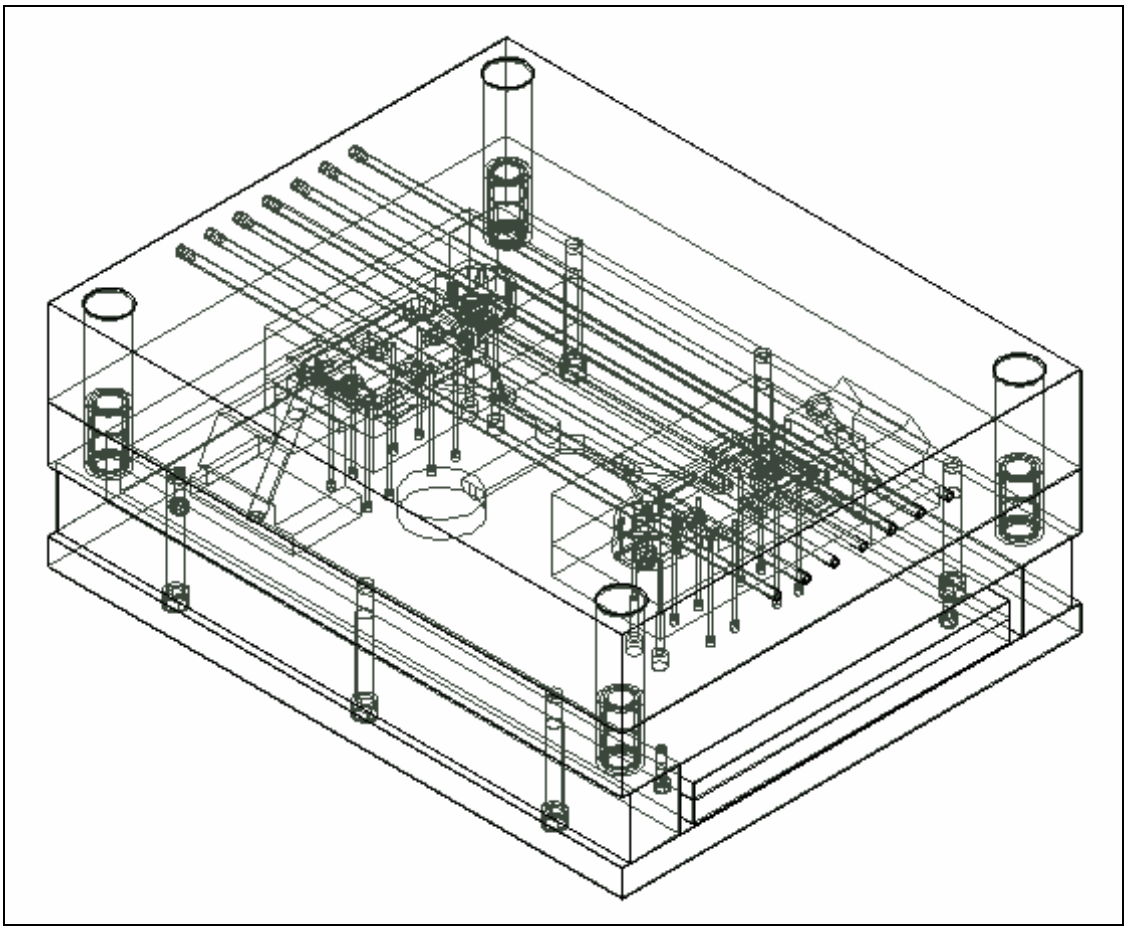


Figure 6.10: Cooling channels as assembled in die base for push button

The final step in the die casting die design process involves adding in some standard components like locating ring and sprue. These two components were modeled according to the DME catalogue and added through the module 'Standard Components'. The locating ring was placed on the top clamp plate and aligned concentric with the biscuit. The sprue was located directly underneath the locating

ring.

With the addition of the standard components, the die designer had successfully created a die design consisting of a 2-cavity layout of cavity inserts, core slides, gating system, die base, ejector system, cooling system and a couple of standard components. The final die design system for die casting is shown in Figure 6.11.

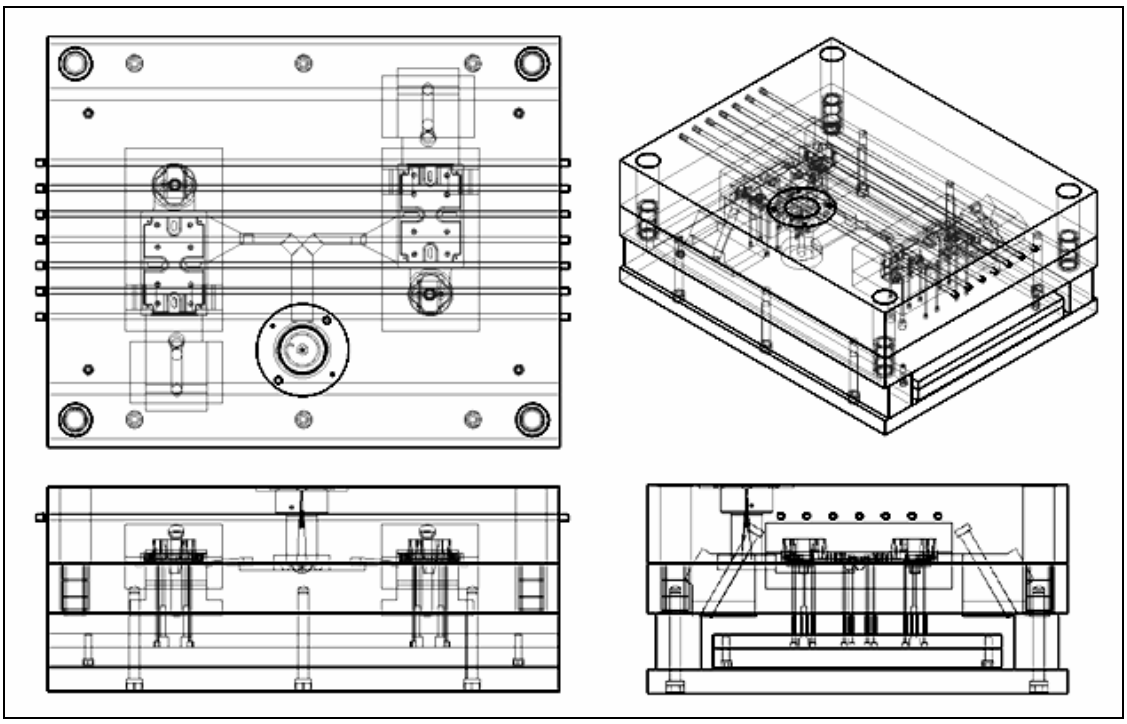


Figure 6.11: Final die casting die design for push button using the proposed system

6.2.2 Case Study B: Motor Housing

As this product model is relatively simpler, it is briefly discussed. The product model is a motor housing as shown in Figure 6.12.

Using the module 'Cavity Insert Builder', the cavity inserts (ejector and cover) for the motor housing and the hard disk cover, are easily created as shown in Figure 6.13.

Since the motor housing does not contain undercuts, core slides are not needed. Therefore the 'Core Slide Designer' module can be bypassed. The next step for the die

designer is to use the module 'Gating System Constructor' to design the gating system for the pair of cavity inserts. To simplify the die design, the layout selected is a single-cavity layout.

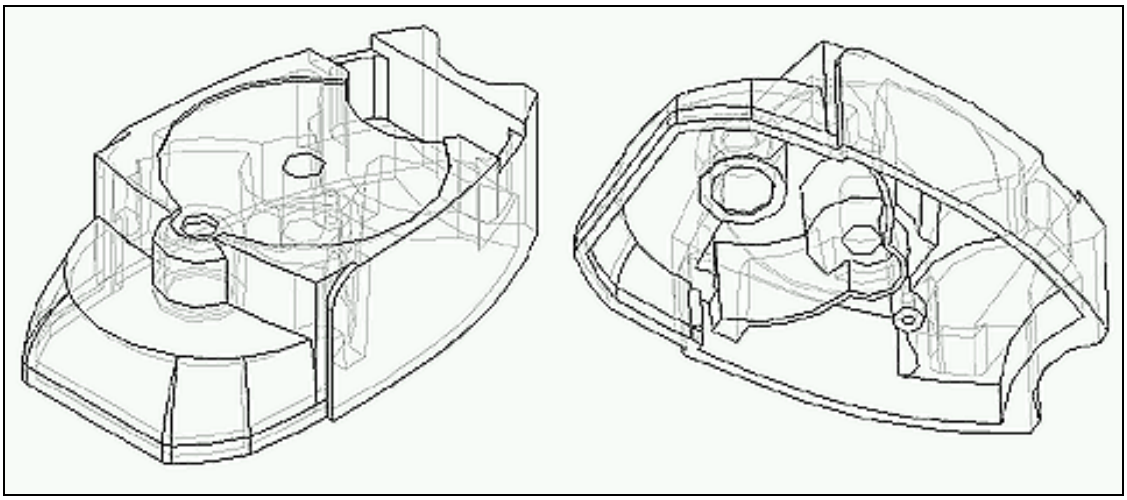


Figure 6.12: Motor Housing

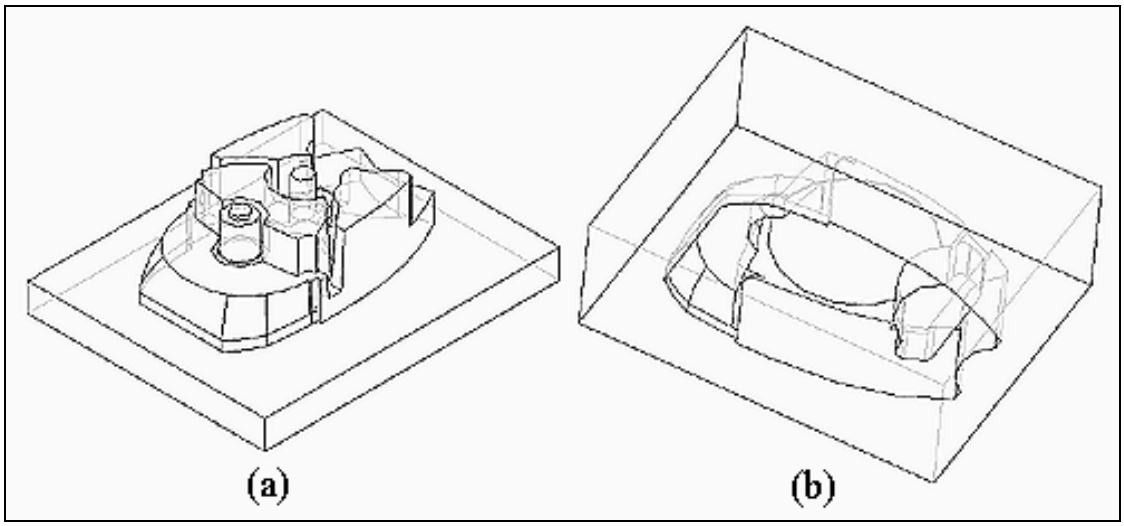


Figure 6.13: Motor housing cavity inserts (a) Ejector cavity insert (b) Cover cavity insert

Next, suitable gates were added to the cavity inserts. The type of gate added is a fan gate. The runner type utilized is a circular runner. As the motor housing is arranged in a single-cavity layout, the runners followed a straight path. Two overflows were also added for the motor housing. The gating systems created for the motor housing is shown in Figure 6.14.

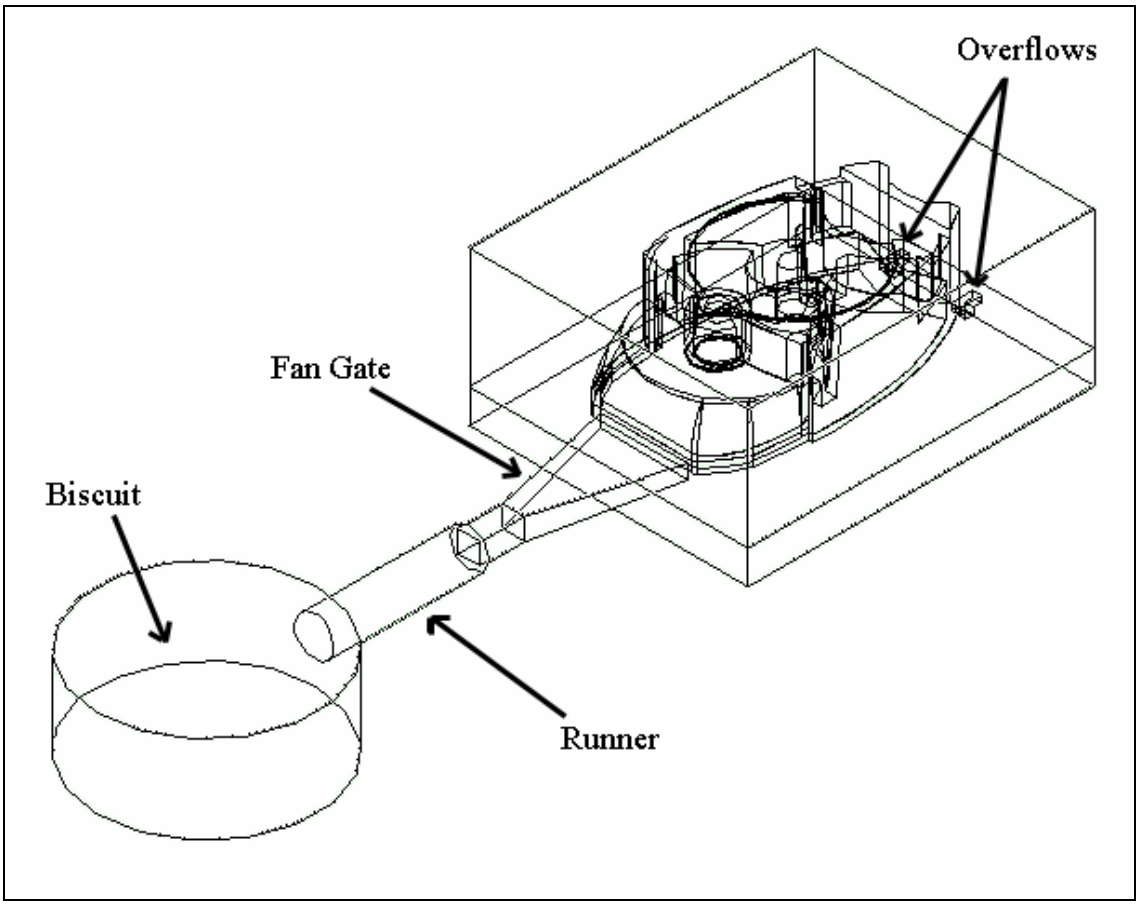


Figure 6.14: Gating system added for motor housing

Next, a die base was added to contain the cavity inserts using the module 'Die Base Designer'. The die base used for both product models is of the vendor DME series D. The size of the die base added for the motor housing is 496 mm by 496 mm by 400 mm as shown in Figure 6.15. It is smaller than the one used for the push button housing as it only contain a cavity insert.

Next the die designer includes ejector pins and cooling flow tubes to the die assembly using the modules 'Ejector System Constructor' and 'Cooling System Designer' respectively. The last task is to use the 'Standard Component' module to add the locating ring and the sprue. The final die design system for the motor housing is shown in Figure 6.16.

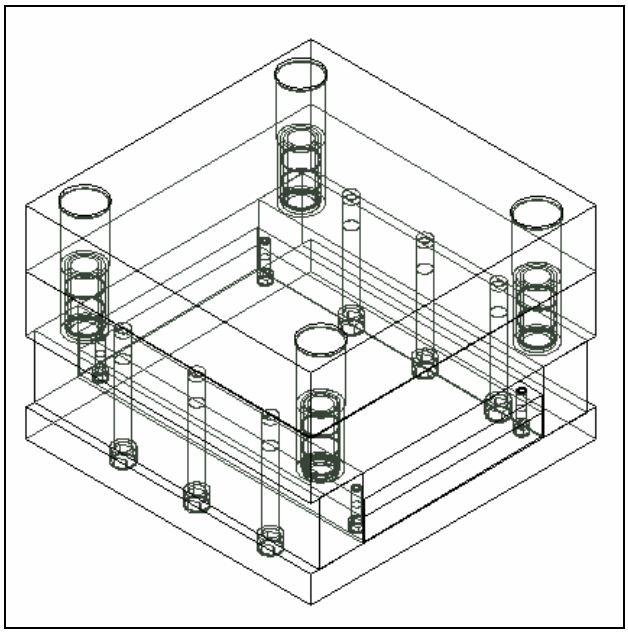


Figure 6.15: Die base used for motor housing with a smaller configuration

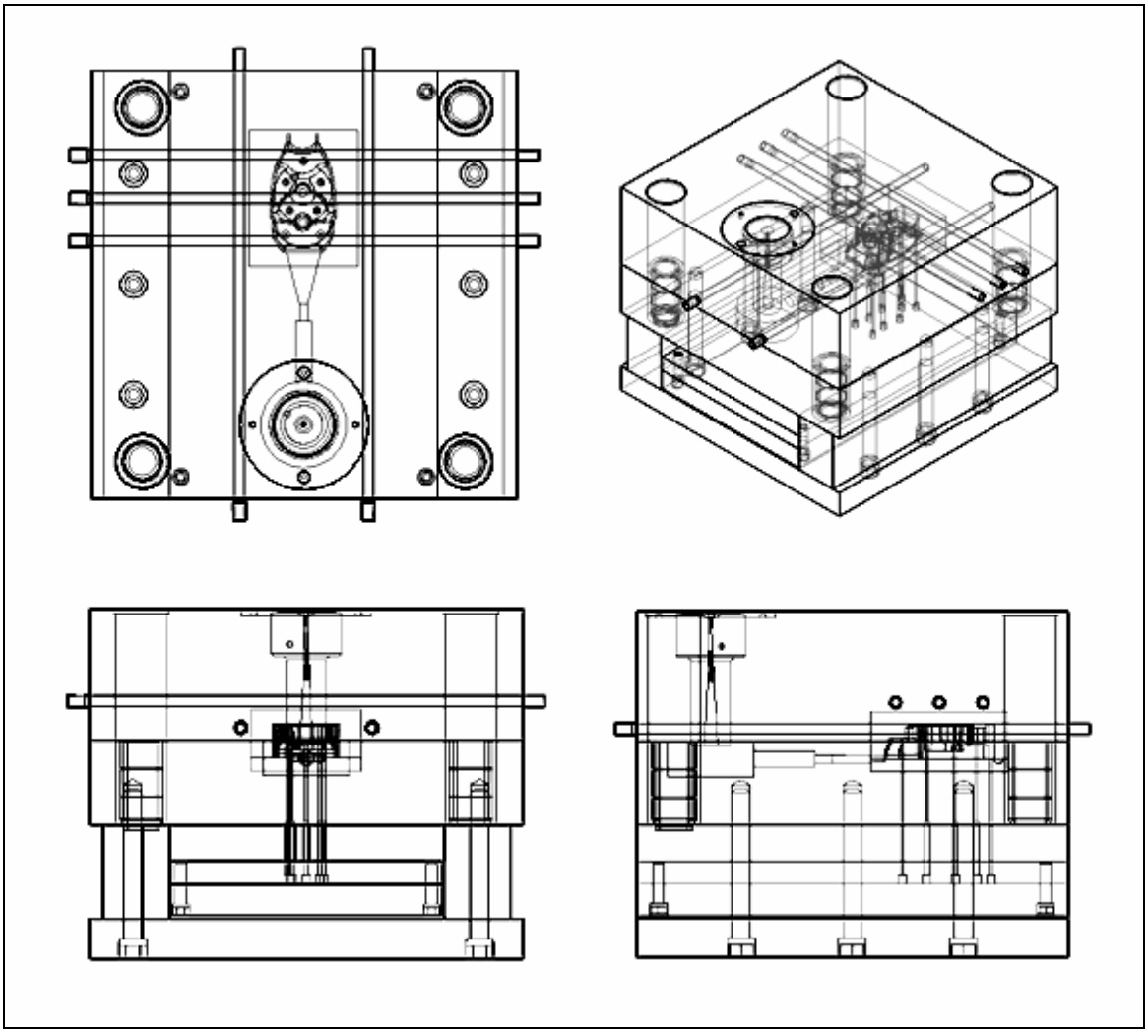


Figure 6.16: Final die casting die design for motor housing using the proposed system

6.3 Discussion

As the aim for the die casting die design system is to reduce lead times and production cycles, it is important to examine the duration of the die casting die design carried out by the proposed system. The total time taken for the entire die design process usually takes less than a day, which is a significant reduction in the lead time as compared to conventional die design process for die casting. One of the main factors for the reduction in lead time is the extensive use of standardization of the die casting design process. The algorithms written that greatly ease the process of selecting parting line, hole patching etc. also play a crucial role in the reduction of the duration for die casting die design.

In addition, the die designer can also edit the various parameter values of the standard die casting components as and when needed. For example, if the die designer felt that the diameter of the previously added locating ring was too small, he can easily open the die design project file and use the ‘Standard Components’ module to make the necessary editing. As an add-in to SolidWorks, the die designer can also utilize the full functionalities of SolidWorks. This is extremely useful when the die casting product part model is too complicated. For example, the hole patching function of the die design system can only patch planar holes. When product models contain non-planar holes that need to be patched, the operation can be very tricky. SolidWorks is equipped with functionalities like hole filling and surface lofting that can do the job. Hence developing the system on commercial CAD software provides convenience for die designers and save precious time.

CHAPTER 7 : CONCLUSION & RECOMMENDATIONS

7.1 Conclusion

In terms of adding value to raw material and cost effectiveness, few other processes can match die casting. The increasing use of lighter-weight metal components, such as aluminium and magnesium die castings, has highlighted the importance of shortening the lead-time of die design. The die casting industry will greatly benefit if more comprehensive applications software are developed that integrate the different die design stages and allow the editing of die design.

A prototype die design system for die casting has been completed. The prototype of the Windows based CAD die design system for die casting is implemented on the platform of the commercial SolidWorks 2001 CAD system under Windows Millennium environment. The development tools used include SolidWorks API, MFC and Visual C++. The prototype system consists of eight design modules and is inbuilt with standard die casting feature libraries.

Initial case studies show that it is able to reduce the lead-time of die design as it integrates the various stages of die design process. Through such a system, die designers are able to design a die with cavity inserts, die base, gating system, ejector system, cooling system and standard components. Moreover it ensures that when die designers modify the die design, the changes can be easily performed, thus removing the need to re-design another die for the same product.

7.2 Contributions

The major contributions of this research are outlined as follows:

- Integrated the various stages of die design process for die casting.
- Allowed easy customization and modifications to the die design as and when necessary.
- Automated or semi-automated several die design process like parting line search, parting surface generation, hole patching, etc.
- Increased standardization by providing feature libraries of predefined standard die casting features that can be loaded conveniently to the die casting project.

In summary, the die design system for die casting had expedite the die design process and demonstrate how increased standardization can lower lead times and reduce production cycles of die design. The methodology used can be applied on a broad range of applications that involve designing with standard components.

7.3 Recommendations

While the research had contributed much to the die design process for die casting, there are some limitations to the proposed die design system for die casting. These limitations had led to the following areas for future work:

7.3.1 Enhance the existing in-built feature libraries

As the proposed die design system for die casting is only a prototype, the feature libraries in built in the system are not sufficient. The system can be enhanced, in the standardization aspect, where more types of die bases, gating features, core slide

features, ejector pins and many more die casting components can be predefined and added.

7.3.2 Develop more computational capabilities

Further enhancement of the proposed system includes more computational abilities, like the calculation of cavity filling time, venting analysis and gating system analysis.

7.3.3 Improve usability and efficiency

As a prototype system, the system is not focused on user-friendliness and efficiency. Hence future work can look into how to simplify some of the more complex areas in the die design system. Moreover, more efficient algorithms can be derived for the automation of the die design process.

References

[1] http://www.diecasting.org, North American Die Casting Association.

[2] A. C. Street, The Diecasting Book, 2nd Edition, Portcullis Press, 1986, pp. 3-17

[3] E.A. Herman, Die Casting Dies: Design, North American Die Casting Association, 1992, pp. 15-24.

[4] C. C. Tai, J. C. Lin, “A runner-optimization design study of a die-casting die”, Journal of Materials Processing Technology, 84, pp. 1-12, 1998

[5] C. C. Tai, J. C. Lin, “The optimal position for the injection gate of a die-casting die”, Journal of Materials Processing Technology, 86, pp. 87-100, 1999

[6] Ching-Chih Tai, “The optimization accuracy control of a die-casting product part”, Journal of Materials Processing Technology, 103, pp. 173-188, 2000

[7] Shamsuddin Sulaiman and Tham Chee Keen, “Flow analysis along the runner and gating system of a casting process”, Journal of Materials Processing Technology, 63, pp. 690-695, 1997

[8] B. H. Hu, K. K. Tong, X. P. Niu, I. Pinwill, “Design and optimization of runner and gating systems for the die casting of thin-walled magnesium telecommunication parts through numerical simulation”, Journal of Materials Processing Technology, 105, pp. 128-133, 2000

[9] CASTFLOW^{TM} Software, Castec Australia Pty. Ltd., 2000

[10] http://www.magmasoft.com, MAGMA Software, Germany, 2001

[11] D. F. Allsop, D. Kennedy, “Pressure Die Casting (Part 2): The Technology of the Casting and the Die”, Oxford, New York: Pergamon Press, 1983

[12] S. H. Wu, K.S. Lee and J. Y. Fuh, “Feature-Based Parametric Design of a Gating System for a Die-Casting Die”, The International Journal of Advanced Manufacturing Technology, 19, pp. 821-829, 2002.

[13] W. S. Zhang, S. M. Xiong, B. C. Liu, “Study on a CAD/CAE System of Die Casting”, Journal of Materials Processing Technology, 63, pp. 707-711, 1997.

[14] A. Voss, “Case reusing systems – survey, framework and guidelines”, Knowledge Engineering Review, 12(1), pp. 59-89, 1997.

[15] K.S. Lee and C. Luo, “Applcation of Case-Based Reasoning in Die-Casting Die Design”, The International Journal of Advanced Manufacturing Technology, 20, pp. 284-295, 2002.

[16] G. P. Syrcos, “Die casting process optimization using Taguchi methods”, Journal of Materials Processing Technology, 135, pp. 68-74, 2003

[17] www.hotflo.com, HotFlo Diecasting Pty. Ltd, 1999-2003

[18] Shao-Chiung Lu, R. Allen Miller, and Gary L. Kinzel, “Computer Aided Local Modifications for the Transition from Part Design to Die Design”, Computers in Engineering, 1, pp. 483-489, 1994

[19] http://www.diedifice.com

[20] DieCAS™, Technalysis, Inc.

[21] J. C. Choi, T. H. Kwon, J. H. Park, J. H. Kim and C. H. Kim, "A Study on Development of a Die Design System for Diecasting", The International Journal of Advanced Manufacturing Technology, 20, pp. 1-8, 2002.

[22] K. S. Lee, J. Y. H. Fuh, Y. F. Zhang. A. Y. C. Nee and Z. Li, "IMOLD: An intelligent plastic injection mold design and assembly system", The Proceeding of the 4th International Conference on Die and Mold Technology, 4-16 June, Kuala Lumpur, Malaysia, pp. 30-37, 1997.

[23] J. Monedero, "Parametric Design: a review and some experiences", Automation in Construction, 9, pp. 369-377, 2000.

[24] D. Roller, "An approach to computer-aided parametric design", Computer-Aided Design, 23(5), pp. 385-391, June 1991.

[25] R. Anderl, "Parametrics for product modeling", in J. Hoschek and Teubner, Stuttgart, 1994.

[26] Tu, J. S., Foran, R.K., Hines, A.M. and Aimone, P.R., 1995, "Integrated procedure for modeling investment castings", *JOM*, Vol. 47, No.10, pp.64-68.

[27] Y. M. Chen and C. L. Wei, "Computer-aided feature-based design for net shape manufacturing", Computer Integrated Manufacturing Systems, 10(2), pp 147-164, 1997.

[28] Lee, K.S., Alam, M.R., Rahman, M., and Zhang, Y.F., 2000, "Automated process planning for the manufacture of lifters", The International Journal of Advanced Manufacturing Technology, Vol.17, pp.727-734, 2000.

[29] S. Shimizu and M. Numao, "Constraint-based design for 3D shapes", Artificial Intelligence, 91, pp 51-69, 1997.

[30] I. Fudos and C. M. Hoffmann, "Constraint-based parametric conics for CAD", Computer-Aided Design, 28, No. 2, pp. 91-100, 1996

[31] R. Anderl and R Mendgen, "Modelling with constraints: theoretical foundation and application", Computer-Aided Design, 28, No. 3, pp. 155-168, 1996

[32] http://www.solidworks.com, SolidWorks Corporation.

[33] SolidWorks 2001 API Release Notes

[34] Jeff Kurtz and Jerry Kurtz, SAMS Teach Yourself Visual C++ 6 Online in Web Time, Sams Publishing, 1999.

[35] http://www.easydesksoftware.com/dll.htm

[36] Clayton Walnum, Special Edition Using MFC and ATL, Que® Corporation, 1997.

[37] http://www.procomps.com, Progressive Components International Corporation