A prototype constraint-based system for the automation and optimization of machining processes

A Gayretli and H S Abdalla*
Department of Mechanical and Manufacturing Engineering, De Montfort University, Leicester, UK

Abstract: This paper presents an intelligent design system that enables designers to consider at the early stages of the design process all issues associated with the product life cycle. The evaluation and optimization of machining processes are among the most important aspects of these issues, requiring the involvement of various types of information from the different aspects of the product life cycle. The representation of this information in efficient format is very important for carrying out different design tasks. In order effectively to manage information exchange within different design and manufacturing domains, it is necessary to provide an efficient and timely communication network system. Therefore, various critical tasks such as overall coordination, control, consistency and data integrity have to be considered in order to avoid costly design iterations. This research article has focused on the development of a prototype system for machining process optimization.

The system uses a combination of both mathematical methods and constraint programming techniques and provides designers with the evaluation and optimization of feasible machining processes in a consistent manner at the early stages of the design process. As a result, unexpected and costly design iterations, which result in wastage of a great amount of engineering time and effort, and in a longer lead time, can be avoided.

The development process has passed through four major stages. Firstly, an intelligent constraint-based design system for concurrent product and process design, including a machining process optimization module, has been developed. Secondly, the product features, processes, cost, time and requirements have been represented in the format of constraints, frames, objects and production rules in order to be utilized to accomplish different design tasks. Thirdly, rules for the selection and optimization of feasible processes for complex features have been written, and finally, an information management system, with a conflict resolution mechanism, has been developed to achieve consistency in information exchange and decision-making activities between the different design areas.

Keywords: concurrent engineering, process optimization, cost estimation, constraints, knowledge-based systems, feature-based design, object-oriented programming, design consistency, conflict resolution

1 INTRODUCTION

The product development cycle encompasses several aspects, including marketing, conceptual design, detail design, process selection and quality control. In each of these areas there is an opportunity for manufacturers to gain competitive advantages by the outperforming of others.

The current trend forces companies to produce low-cost and high-quality products in order to maintain their competitiveness at the highest possible level. This can be achieved by the best use of manufacturing resources such as machine tools, cutting tools, labour and processes to minimize the amount of time spent adding cost. This results in an effective and fast response to market requirements. Concurrent engineering is a systematic approach to the integrated, concurrent development of a product and its related activities such as process selection, tooling and time/cost estimation in order to meet customer requirements. It is therefore necessary to achieve the concurrent involvement of
various life cycle perspectives in the product development process.

Concurrent engineering allows the design team to consider the factors affecting product cost, lead time, manufacturability and selection and evaluation of machining processes. Since researchers have given little attention to manufacturing cost estimation and optimization, this recent research work has concentrated on optimization of machining processes as it plays an important role in cost reduction [1]. It has been reported that almost 70 per cent of total product cost is based on decisions made at the early stage of the design process [2].

The various requirements associated with the perspectives of the design process such as part features, feature-process relations, machine tools, cutting tools, cost and time have to be taken into consideration in concurrent product development to satisfy the market expectation. These requirements also have a direct impact on the cost effectiveness of the design.

Representation of these requirements is very important for effective use during the design process so as to prevent designers/manufacturing planners from being engaged in a time consuming iteration process and accomplish many tasks such as manufacturability analysis, process selection and process time and cost estimation. Constraint-based systems are useful tools that are used to model and handle design requirements. However, complex designs cannot be easily represented in the form of constraints and variables. These systems should also offer flexibility to enable designers to attach new databases to the system. In addition, they must include an information management system for handling any design conflicts or constraint violation.

An efficient and timely communication network system should be provided within different design and manufacturing areas so that effective management of those constraints can be achieved. Various critical tasks such as overall coordination, control, consistency, and data integrity have to be considered in order to prevent costly design iterations. This necessitates establishing local area networks (LANs) within the organization to achieve the integration of different design areas in a consistent manner. Such integration needs a strategy for conflict resolution and reaching a consensus decision based on the perspectives of all related domains. Several authors have carried out research work in the area of information modelling and management, including Abdalla [3], O’Grady et al. [4], Noble [5], Karacali and Bell [6], Balasubramanian and Norrie [7] and Lander and Corkill [8].

There is no doubt that a concurrent product development approach improves productivity and helps to design products that offer high quality, reliability and less cost. It also reduces design iterations, hence leading to shorter product lead times. Recent research work has been carried out in the area of developing methods and tools for the estimation and optimization of manufacturing costs [9–14]. A number of papers have examined feature-based models focusing on machining form features [1, 15]. Research work in activity-based cost estimation of components can be found in the work of Shaikh and Hansotia [16], Das et al. [17, 18], Luong and Speeding [19] and Pham and Gologu [20].

Manufacturability analysis is another important factor in the drive to reduce product costs. A number of methods were developed to enable designers and manufacturing planners to address this detailed analysis [21–23]. Existing approaches to process evaluation and optimization are generally limited to feasibility evaluation and optimization of particular geometric specifications and available process combinations and capabilities. The best use of available alternative processes and concurrent consideration of manufacturability analysis and process evaluation and optimization have not yet been fully exploited.

Since large amounts of information exchange within different design areas and decision-making activities are involved in the design process, it is essential to develop an information management system including a conflict resolution mechanism to deal with any disagreements arising from different domains of the product life cycle. Therefore, the focus in this research work is on achieving such optimization, integration and consistency.

2 PROPOSED CONSTRAINT-BASED SYSTEM FOR CONCURRENT PRODUCT DEVELOPMENT

The proposed model consists of a computer aided design (CAD) solid modelling system, user interface, design representation, consistency manager, constraint-based system, process optimization and manufacturability analysis, and various knowledge sources (Fig. 1). All of these elements interact with one another, subject to the type of information that is required. It provides the designer with a flexible access to any level of the design process.

The procedure for designing a component via this system requires the designer to interact with a CAD system to generate a component and its features. The product information obtained from this system is passed to the knowledge-base system (KBS) via the user interface. The KBS includes a number of rules for executing several tasks, constraints and information about various aspects of the design process. The user enters information associated with manufacturing resources and capabilities, together with other areas of the design process, as a set of constraints. Based on the information provided from the designer and other expertise, the system carries out various tasks. It begins with checking the manufacturing capacity, and then
features and dimensions of the component. It continues with the selection of processes, machines and cutting tools, and then goes on to evaluate process time and cost. The system provides the designer with an evaluation of all the decisions associated with part design by using the rules developed in the knowledge-base system.

It uses information associated with the manufacturing of form features, machine tool and cutting tool data, material data, criteria and goals to generate feasible process plans. The system gives recommendations on a design that cannot be manufactured with the available manufacturing resources. Also, it allows effective conflict resolution strategy for design inconsistencies arising from different areas of the design process, and provides the designer with a user-friendly interface, including visual and textural results from the analysis. The elements of the proposed system are explained as follows.

2.1 Constraint-based system

Constraints exist in terms of relationships between different requirements of the product during its life time, which have an important effect on the product cost/time and quality. The constraint-based system is a tool for representing and handling these requirements and can be formulated as sets of constraints. Constraints are used to model the requirements associated with various life cycle issues for effective use during the design process. It includes constraints of different product life cycle perspectives and design variables, together with a constraint propagation module for ensuring design consistency within the constraint network. When a value is assigned to a variable of the component, the constraints propagation module checks the assigned value to see if it violates any constraints. A valid solution is reached by the satisfaction of all constraints. When a constraint violation exists, the user is informed of the violation and given a warning, followed by some alternative suggestions from which he/she has to make a selection. In the system, constraints are formulated as rules, variables, values and domain.

The proposed model covers most aspects of design and manufacturing constraints, which include process constraints, machine constraints, material constraints, tooling constraints, part constraints, material handling constraints and tolerance and surface finish constraints. The constraint-based system is also linked to a consistency manager, ensuring consistent information exchange in the constraint network.

2.2 Consistency management

The consistency manager has the responsibility for managing the decision-making process and dealing with conflict situations, and providing the user with justification of decisions made on design. When it detects conflicts, warning(s) are given to the user with reasons for the conflict. A suitable strategy for solving conflicts is applied by the system in order to ensure design consistency in the constraint network and design output. The consistency manager allows designers to take necessary actions against the problems by giving some suggestions and to observe the design violations via the user interface.

2.3 Design representation

The knowledge-base system toolkit KEE (Knowledge Engineering Environment, developed by Intellicorp.) has been used to represent and model the product life cycle requirements and the design model. Building the design model and those requirements in a systematic
and well-organized way is essential in order to provide an effective interaction between the various design tasks such as design, manufacturing and tooling.

An object-oriented representation technique, frames, constraints and production rules have been used for the organization of knowledge. This consists of classes, objects, units, rules and/or methods with attributes inherited to all subclasses. The flexibility of these techniques enables designers to modify the existing objects or classes and add new units and attributes.

2.4 Process optimization and manufacturability analysis

This module has a rule-based algorithm for analysing the component and its features in order to select processes, machines and tools. The selected processes are then evaluated. The process times and costs of the components are estimated in order to ensure the manufacturability of the component. The system analyses processes and sequences and calculates the total machining cost of the product, including material, tooling, machining, overhead and labour costs. If the targeted process costs and times are not reached, then the system should be given advice from the designer to make modifications. This process will continue until a cost effective product is obtained. The rest of this paper discusses further details of how the process selection and optimization module functions.

2.5 User interface

To provide the user with an interactive design environment, a user-friendly interface has been developed as an important part of the expert system in order to enable the user to use the system easily and efficiently. KEE features such as menus and images were used to create the user interface so that the user-defined values can be obtained to accomplish design tasks. To enable the user to monitor constraint violations and value changes, active images are also incorporated in the user interface. Activating methods in slots by the use of a mouse was made possible by using the method actuators included in the user interface. In addition, the user interface enables users to interact with the CAD solid modelling system (Pro/Engineer) to generate three-dimensional solid models, add features and modify features and their attributes. The user communicates with the system using a superpanel, including menus, active images and method actuator, and results from the system are displayed in the Lisp listener, KEE output window and typescript window.

Fig. 2 Proposed model for the process of cost estimation and optimization

3 PROPOSED MODEL FOR PROCESS SELECTION AND OPTIMIZATION

As can be seen from Fig. 2, the proposed model consists of a form feature database, designer requirements, machining processes and constraints, an evaluation and optimization module and a user interface. Each module of the proposed system interacts with one another. The user interface provides the designer with the access to each module, and his/her requirements, formulated as a set of constraints (i.e. process time and cost, tool cost, set-up time and cost), will be the input to the system. For example, process costs, times and tool cost can be constrained by the predefined values provided by the user.

The form feature database was built to include various types of form feature. The selection of the feasible processes for the component is carried out using the form features and parameters retrieved from the form feature database.

Manufacturing information such as feature type, material, length and diameter ratio, cutting tool specifications, process availability, machine accessibility, process sequences, tolerances, surface finishes, optimum cutting parameters, cost and time were included in the processes and constraints module. Processes and constraints were also represented as hierarchies and objects in this module in order to evaluate and optimize the machining processes for a component. The system carries out an analysis of features of the component and then chooses the feasible machining processes for the component and calculates process times and costs, subject to the manufacturing constraints. For example, a component that has a form feature with a tight tolerance, special surface finish, and complex shape may need a machining centre to be manufactured by using a special cutting tool.

The required tolerance and surface finish of the feature could be obtained by a reaming and grinding process. Finally, the system, including a rule-based algorithm to evaluate the chosen processes based on
some criteria given in the design specification, calculates the total process time and cost of the component. If no process combinations are found acceptable, the system generates a dialogue including suggestions on the design, which the user has to take into consideration. This process can be continued until a set of feasible process combinations is obtained.

3.1 Feature representation

A feature is a generic entity which possesses product information and which may be used for design or communication in design, manufacturing and other engineering tasks such as assembly, manufacturing, process selection, cost/time estimation and maintenance. The representation of the features should be explicit in a form that matches manufacturing knowledge. Analysis of the form features directly associated with certain machining processes has an important effect on generating a detailed process plan. In this analysis, manufacturing form features are the lynch pin of the generation and optimization of machining processes and provide communication between designer and process planners to consider how their decisions can affect the product and process design. The use of manufacturing form features helps designers to simplify process planning without consideration of component manufacture in unlimited ways. Therefore, the feature-based representation technique has been used to represent the component and features in greater detail so that the designer, process and assembly planners or an expert system can use the same model to carry out various design tasks. Cost effective process planning can be achieved by the definition of manufacturing form features that are derivable from topological and geometrical description of the component. For example, a slot is a form feature defined by its parameters such as name, diameter, depth, locations, tolerance, process and surface finish. Based on these parameters the machining processes, set-ups, fixture and cutting tools can be chosen. The most common form features manufacturable on machine tools, and a three-dimensional model of an engine head composed of such features, are shown in Figs 3 and 4.

The proposed model contains production rules and knowledge about the form features and machining processes. These rules manipulate the behaviour of the feature and process data which are represented in a structured way and an effective format made to reach a feasible solution. Manufacturing environment capabilities (i.e. production size, maximum length, diameter, tolerance, surface finish and tools) are contained in the rules. Manufacturing form features are represented by using an object-oriented representation technique, as shown in Fig. 5.

3.2 Representation of design and manufacturing knowledge

Knowledge representation techniques used to represent
design and manufacturing knowledge in this research are described in detail as follows.

3.2.1 Constraints

In the design process, there exists a huge amount of information from different aspects of the product life cycle in the form of design requirements, which have to be met by the designed product, and those requirements that can be represented as constraints. For instance, from a process planner's point of view, the tolerance of a hole could be a constraint or a customer could constrain the product cost. Constraints represent certain limitations or restrictions on design variables. Design and manufacturing variables can be effectively held in a slot or a rule class, and can be kept between certain values defined as constraints. In this research, design and manufacturing requirements are formulated as a set of constraints in slots of a unit and the production rules.

The various types of constraint used in this research are as follows:

(a) domain (i.e. machine tools, cutting tools, manufacturing capacity, etc.);
(b) equations [material removal rate (MRR) = \( \pi Dr^2 N \) or \( 8 \geq d \leq 80 \text{ mm} \)]
(c) production rules (If (feature is hole and surface roughness \( \geq 6 \)) then (process is drilling))
(d) logical constraints (Do you want to change the diameter? (YES or NO))

Design and manufacturing variables are stored in the slot of an object, including working constraints. An example of how manufacturing requirements are formulated as constraints is shown in the following rules in which a form feature has its own variables (diameter, depth, lower_tolerance, upper_tolerance and surface_finish).

The lower_tolerance variable has its own constraint: \( \text{isp}(> = \text{lower}_\text{tolerance} 0.025), \) or the lower_tolerance of the feature must be equal or exceed the limit 0.025 mm. Figure 6 shows the flow chart of process selection for blind holes:
(slot_end_milling_rule_1)
(if (what is in block-slot)
   (the length of ?what is ?length)
   (the width of ?what is ?width)
   (the depth of ?what is ?depth)
   (the finish_allowance of ?what is ?finish_allowance)
   (the lower_tolerance of ?what is ?lower_tolerance)
   (the upper_tolerance of ?what is ?upper_tolerance)
   (the surface_finish of ?what is ?surface_finish)
   (lisp (<= ?lower_tolerance -0.005))
   (lisp (>= ?upper_tolerance 0.005))
   (lisp (> = ?surface_finish 0.8))
   ......)
then
   (the volume of ?what is (lisp (- (* (* ?length ?width) ?depth)
   (lisp (format t
       "~%"

   THE POSSIBLE PROCESSES TO BE EVALUATED
   ===================================================
   FEATURES           POSSIBLE PROCESSES
   -----------------------------------------------------
   ~D                 MILLING
   -----------------------------------------------------
                        EDM
   -----------------------------------------------------
   ?what()
   (the first_process.selection of ?what is ok)
   (the possible_process_1 of ?what is milling)
   (the possible_process_2 of ?what is edm)
   (more rules))
(slot_end_milling_rule_2)
(if (what is in block-slot)
   (not (the first_process.selection of ?what is ok))
Frames as a knowledge representation technique are used for storing interconnected information about a design and an object. Knowledge representation of stereotypical objects can be achieved effectively by using the frames consisting of a name and a number of slots. Various types of value [i.e. numerical (12, 24), logical (yes or no), procedural (methods) and symbolic (steel)] can be used in the slots. The frames of the KEE are very flexible so that images and active values to any slots can be attached to monitor value changes. Facets as attributes of slots (i.e. value class, inheritance role, maximum and minimum cardinality) allow description of values of a slot and how they are passed down the hierarchy. The example below shows how a product design specification can be represented in the form of frames:

**(Blind_hole_rule_1)**

(if (what is in blind_holes)
  (the diameters of ?what is ?diameters)
  (the depth of ?what is ?depth)
  (the lower_tolerance of ?what is ?lower_tolerance)
  (the upper_tolerance of ?what is ?upper_tolerance)
  (the surface_finish of ?what is ?surface_finish)
  (......)
  (lisp (>= ?lower_tolerance 0.025))
  (lisp (>= ?upper_tolerance 0.15))
  (lisp (>= ?surface_finish 1.6))
  (......)
  then
  (lisp (format t "*****The possible process for -d is drilling, end_milling and edm.*****" ?what))
  (the volume of ?what is (lisp (* (* (/ ?pi 4) (?diameters ?diameters)) ?depth)))
  (the first_process_selection of ?what is ok)

**Superclass:** Product Specification

**Subclasses:** Part, Manufacturing Cell Capacity, Available Machine Tools

**Properties:** (part type)

(value (lambda (self))

(with-keep (setq choice (prompt-use 'choice-multiple
  :choices '(rotational non-rotational)
  :prompt "Please select one:"
  :few-choices-mode 'menu)))))

(inheritance method)

(valueclass (h[?Unit: method keedatatypes])) (default nil)

(activimage((?Unit: windowpane-availability-of-machine.constraints
  manufacture007]))

  :unique_values (cardinality.min (1)) (cardinality.max (1))

(length .....)

(the_possible.process.1 of ?what is drilling/counterboring)

(the_possible.process.2 of ?what is milling)

(the_possible.process.3 of ?what is edm)

(......)))

3.2.2 Frames
3.2.3 Production rules

Knowledge and facts about a problem domain can be represented as a rule of the form \textit{IF} premises \textit{THEN} conclusion. The production rules are very effective for storing design and manufacturing constraints in the production rules. The proposed model uses production rules as values of unit attributes (slots) to be manipulated and inherited from higher classes to subclasses. The production rules included in slots enable a complex structured rules system to interact with different sets of rules associated with different units. A combination of these rules with methods, which are LISP functions stored as a value of a slot, allows the rule system to run fast and efficiently. As an example, in the \textit{prototype-testing-rule-1} shown below, total manufacturing cost is calculated by using a LISP function as shown in the rules set out below.

The command \textit{UNITMSG} sends a message to the \textit{total cost} method in the \textit{part} unit to perform the calculation of the total process cost. The calculated value will be the new value of the \textit{total_manufacturing_cost} slot:

\begin{verbatim}
(prototype-testing-rule-1)
(if (the total_m_cost_control of part is ?total_m_cost_control)
 (?total_m_cost_control = ok)

then

(the total_manufacturing_cost of part is (lisp (unitmsg 'part 'total-cost))))

(prototype-testing-rule-2)
(if (the total_manufacturing_cost of part is ?total_manufacturing_cost of part)
 (the target_manufacturing_cost of part is ?target_manufacturing_cost)
 (lisp (>= ?total_manufacturing_cost of part ?target_manufacturing_cost))

then

(lisp (format t "~s\%\% The estimated manufacturing cost of part is higher than the proposed target manufacturing cost. The target manufacturing cost is ~D $ and the estimated manufacturing cost is ~D $,~s\%\%"

?total_manufacturing_cost

?target_manufacturing_cost))

(prototype-testing-rule-3)
(if (more rules)))
\end{verbatim}

3.2.4 Object-oriented programming

The object-oriented programming technique enables designers to model real world concepts as objects which are collections of data grouped together in terms of similarities in their structure and behaviour [24]. By using this technique, design and manufacturing objects such as machine tools, cutting tools, features, material features and machining elements are organized into various classes represented in hierarchies (Fig. 7).

A class has a name and several subclasses (consisting of a number of objects) with a number of slots (attributes such as capacity, power, feed rate, size and tolerance). All classes can be broken down into subdivisions so that all components of the class are considered. An object or a member of a class (i.e. machines, cutting tools and materials) can be added to a subclass to represent the available manufacturing resources of a company. Inheritance is an important characteristic of the object-oriented programming technique. It enables the designer to define a specific value in a higher class to be inherited by the lowest class of the hierarchy.

3.3 Problem-solving techniques

A model of the knowledge base is built to use information about different domains to carry out various analyses. However, the proposed system needs problem-solving techniques and reasoning heuristics in order to find answers to the queries passed by the designer. Object-oriented programming, production rules and the
3.3.1 Object-oriented programming (OOP)

As mentioned in the previous section, knowledge bases consist of objects, which can carry problem-solving behaviour in the form of a LISP function stored in the total_cost slot of the part object as shown in Section 3.2.3. The behaviour or a function of the method (which includes the LISP function) is stored inside objects of the knowledge bases. The LISP functions in the method slots accomplish the given tasks such as time/cost calculations of a component. This is shown as follows:

```
(total_cost

(value

((lambda (thisunit)

(let ((cost_values (get.values 'part 'total_m_cost)))

(apply #'+ cost_values))))

(inheritance method)

(valueclass (# [Unit: method keedatatype]))

(default nil))
```
3.3.2 Production rules

Production rules are also used for reasoning. Several rules take the expert from an intermediate conclusion to a more precise conclusion. In the proposed system, several rule classes have been developed and connected to each other, namely the conclusion of one rule is included in the premise of another rule. This is called chaining. When chaining commences, conclusions of one rule class match premises of another rule class. Chaining is used either in a forward or backward direction.

3.3.2.1 Forward chaining. Forward chaining tries to find implications of new information. It generally starts from the input of new data from the user or from a different domain of the knowledge base. It is called event-driven or data-driven reasoning. The system scans the rules whose premises include the new fact. If all the premises of a rule class are true, the conclusions of the rule class will be asserted in turn and become new information. The system then searches for the rule that possesses this new information as a premise, and checks to see if all premises of this rule are true. This process continues until no rules are found with the matching premises.

3.3.2.2 Backward chaining. Backward chaining tries to verify a given fact or hypothesis. As backward chaining starts with the goal of proving something, it is termed goal-driven. The system scans the rules whose conclusions match the fact to be verified. The fact is found to be verified if all the premises of the rule class in question can be verified in turn.

3.3.3 Combination of the problem-solving techniques

In this research work, the combination of the above-mentioned techniques are used in the same rule classes effectively. Using only one technique is not sufficient to
carry out the design tasks in a reasonable time. Therefore, rule classes may become very long and complex. Thus, maintenance, editing and updating are not easy tasks for the designer. A reasoning technique may be good for a certain application, as each technique has its own advantages. For instance, these methods are very effective and fast for calculations, and can be activated in rule classes. Backward chaining is very effective to obtain input from the user by using user dialogues. Forward chaining is very good for searching for specific information in the database. Thus these reasoning systems were combined in the rule classes to reduce the size of rule classes, create more powerful rule applications and make the system flexible and run more efficiently.

3.4 Process selection and optimization

Process planning aims to produce components in accordance with the design specification in order to achieve the highest possible quality. However, economic considerations should be taken into account. After an optimal process plan is generated, it has to be improved with respect to the given criteria. Process optimization needs to be carried out at several detail levels. At the highest level, processes, machine tools and also the sequencing of operations can be chosen. At a more detailed level, the selection of feasible cutting parameters such as feed rate, cutting speed and depth of cut should be carried out. Also, process times and costs of processes, tools and set-ups are calculated at this level.

3.4.1 Criteria for process selection

Process selection requires a number of criteria to be considered for feasible process selection for each form feature. Some criteria are shown as follows:

(a) feature type: through hole, blind hole, and slot;
(b) material: hardness and machinability;
(c) length and diameter ratio: 8/1 for twist drilling, 5/1 for boring, 20/1 for EDM;
(d) availability: YES/NO from user;
(e) tolerance: broaching ±0.025 and boring ±0.04;
(f) surface finishes: drilling 1.6–6.3 and reaming 0.8–3.2.

As an example, some of the above criteria are contained in rules for process selection as shown in Section 3.2.2 for process selection for a blind hole. Further details of the process selection procedure are available from the authors on request.

3.4.2 Process evaluation

The material to be used for the component has an important effect on the selection of the machining parameters such as tool material, cutting speed, feed rate and tool diameter. These machining parameters

Fig. 8 Optimization approach
can be obtained from various machinery handbooks (e.g. reference [25]). Some of the cost data figures on the estimation of direct labour cost, time, material cost, operations, parts and manufacturing technologies are obtained from reference [26]. It is also used to consider the parameters and specification of machining tools (i.e. set-up times, time to stop and start machines, cutting tools and fixtures). Figure 8 shows the proposed model for selection and evaluation of the machining processes.

Process optimization is defined as the reduction in total process times and costs on the basis of process variables (i.e. cutting speed, feed rate, cutting force, power and surface finish constraints) [27]. A table was generated to consider form features, selected possible processes for each feature and the criteria for the optimization and evaluation of the processes for the component. By using this table, constraints for processes for each feature, based on their input on process time, cost and tool cost, can be considered. The form features of the component, possible processes and constraints are shown in Table 1. The system carries out process time and cost calculations. Each form feature is matched with the processes that can produce it, and are listed in the table.

For example, feature_1 has three possible processes (process_1, process_3 and process_n). Generating this table helps to consider the possible processes for each form feature so that the evaluation of the combination of all processes for the component can be carried out by assigning only one process to each feature in turn. This provides users with the evaluation of hundreds of process combinations for a part.

However, this process may take time to find the feasible processes if hundreds of processes for a feature exist in the system. Therefore, to find the feasible processes in a reasonable time, the unrealistic processes
have to be left out of the table. A rule-based algorithm was developed to deal with this problem, and is presented later. The system also allows the users to define the maximum allowable cost and time for the candidate processes of a feature. This makes the system very flexible in keeping the total process cost and time of the part or a single form feature to the predefined values.

Then, the system evaluates the set of process combinations or possible processes for the form features of the component by comparing process cost and time and the other variables against the user’s requirements. Feasible processes will be those that meet the user’s requirements. One of the feasible process combinations is selected as a solution, or further evaluation can be carried out. Specific constraints (i.e. process cost and time) can be changed to obtain a feasible solution unless any process combinations satisfy user requirements. Changing the constraints also enables a process combination to be obtained if necessary. In addition, the user interface informs the user of results of the process selection.

### 3.4.3 Process time and cost estimation

The calculation of process time and cost is carried out using standard formulae. As the proposed approach uses feature-based cost estimation and optimization of machining processes, the following formulae were used for the estimation of process time and cost:

\[
T_{\text{process}} = \frac{\text{form feature volume}}{\text{material removal rate}}
\]

The volume of the form features was calculated by using standard formulae. Material removal rates change from one process to another, subject to certain criteria (i.e. tool diameter and type, type of material and cutting parameters). Material removal rates of some machining processes are shown in Table 2.

Reductions in process time/cost require maximization of the material removal rate (MRR) depending on the following parameters:

- \(T = T_{\text{max}}\) (tool life)
- \(D = D_{\text{max}}\) (tool diameter)
- \(f = f_{\text{max}}\) (feed rate)
- \(V = V_{\text{max}}\) (speed)
- \(W = W_{\text{max}}\) (depth of cut)

Cutting parameters (i.e. cutting tool diameter, feed rate, spindle speed and depth of cut) should have maximum values to maximize the MRR. Also, cutting tool selection has to be taken into account. In addition, properties of the selected material, such as hardness, machinability and electrical conductivity, affect the cutting parameters. MRR for any processes is calculated using the related formulae after suitable cutting tools have been selected in order to meet material requirements. The estimated process time/cost for producing the form features is then calculated. The values of the productive hour cost (PHC) for various processes from the cost estimation databases have been used to calculate process costs.

Total process cost is calculated by using PHC values as follows:

\[
\text{Total process cost} = \text{lot time} \times \text{PHC} \quad (2)
\]

Lot time has to be calculated subject to the quantity of part or form features. A list of some of the PHC values (US$/h) for processes including set-up costs is shown below:

- Machining centre: 12.53
- Drill/counter bore/ream: 11.64
- Face, side, slot, form, end: 12.20
- Drilling machine set-up: 10.84
- Tool life and replacement: 11.25
- EDM: 12.05
- Chemical machining: 9.67
- Milling machine set-up: 12.10

<table>
<thead>
<tr>
<th>Processes</th>
<th>Machining time</th>
<th>Spindle power</th>
<th>Tool life</th>
<th>Material removal rate</th>
<th>Form feature volume</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Conventional</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Drilling</td>
<td>(2l/V_i))k + 2l/V_i (i \geq 1)</td>
<td>(\text{TN}/63 030)</td>
<td>---</td>
<td>((\pi D_t^2/4)/N)</td>
<td>(\pi D_t^4/4h)</td>
</tr>
<tr>
<td>Rough milling</td>
<td>(F_t/MRR)</td>
<td>(F_t/V_i/33 000)</td>
<td>(C/V_f^2/\alpha_p)</td>
<td>(W_{m_a}/hN)</td>
<td>Feature volume</td>
</tr>
<tr>
<td>Finish milling</td>
<td>(Sh/V_i)</td>
<td>(F_t/V_i/33 000)</td>
<td>(W_{m_a}/hN)</td>
<td>Feature volume</td>
<td></td>
</tr>
<tr>
<td>Grinding</td>
<td>(LT_{\text{Dia}}/(W/P))/(\pi V_i)</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>Feature volume</td>
</tr>
<tr>
<td><strong>Non-conventional</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EDM</td>
<td>(F_t/MRR)</td>
<td>---</td>
<td>---</td>
<td>49 cm³/h*</td>
<td>Feature volume</td>
</tr>
<tr>
<td>Electrochemical machining</td>
<td>(F_t/MRR)</td>
<td>---</td>
<td>---</td>
<td>Max. 1000 cm³/h*</td>
<td>Feature volume</td>
</tr>
<tr>
<td>Laser beam machining</td>
<td>(F_t/MRR)</td>
<td>---</td>
<td>---</td>
<td>Average 0.4 cm³/h*</td>
<td>Feature volume</td>
</tr>
</tbody>
</table>

Definitions: \(l_{\text{max}}\) = maximum drilling depth; \(T_{\text{r}}\) = total stock removed from the diameter; \(V_i\) = feed rate; \(N\) = spindle rotational speed (r/min); \(k\) = index for the number of drilling cycles; \(D\) = diameter; \(D_t\) = tool diameter; \(F_t\) = feature volume; \(l\) = length of the hole to drilled; \(\eta_{m_a}\) = machine efficiency; \(h\) = depth; \(\text{MRR}\) = material removal rate; \(F_c\) = cutting force; \(C\) = a constant determined by the geometry of the hole, tool material and part material; \(V_c\) = cutting speed; \(\eta_{c}\) = depth of cut (in); \(g_i\), \(\alpha_i\), \(\beta_i\) = coefficients, \(W\) = width of cutter; \(n\) = number of teeth; \(L\) = length of the part grind; \(T\) = torque; \(Dia\) = original diameter; \(W_i\) = width of the grinding wheel; \(P\) = traverse for each work revolution in fraction of wheel width; \(f_i\) = infed of wheel per pass; \(V_p\) = workpiece peripheral velocity; \(Surf\) = feature surface; \(\Omega\) = overlapping factor, cutting depth for circumferential milling and width of cut for face milling.

* MRR is subject to characteristics of materials (type, hardness, electrical conductivity, etc.).
Boring and facing 13.16
Surface grinding 11.25
Vertical internal grinding 11.25
Horizontal milling, drilling, boring 13.16
Chemical machining 11.49
Travelling wire electrical discharge machining 12.05
Die casting 11.57
Injection moulding 11.07

Using the above figures, the total cost is formulated as follows:

\[ \text{Total cost} = \text{material cost} + \sum \left( \text{lot time} \times \text{PHC} \right) + \text{tool cost} + \text{set-up cost} \] (3)

Set-up times for various machine tools are available in machining handbooks and were used to estimate set-up costs to obtain a more accurate cost estimation [25, 26, 28].

3.5 Information management and design consistency

Different design tasks (i.e. material selection, manufacturability analysis, process selection and optimization) need a huge amount of information to be accessed and shared in the knowledge base so that they can be carried out. This also necessitates the addition of new information to the knowledge base used to carry out such tasks. As seen from Fig. 9, several agents representing the life cycle aspects of the product have their own tasks to carry out. They have a common knowledge base to access necessary information.

Agents as an entity are capable of solving locally generated problems through communication with other agents [29]. They have responsibilities for solving a given task in a design problem such as process selection and capacity checking. Agents should include a limited amount of program for dealing with the given subtasks in order to execute each task faster and at less cost. Also, they will be created or modified easily when necessary. Agents should interact with one another and exchange information in order to accomplish their own task. In order to ensure consistency in the constraint network, any new information from users or agents is propagated by the constraint-based system. This checks to see whether or not the new information causes constraint violation. Agents share information, while consistent information flow is achieved in the system. This is shown in Fig. 10.

An agent has to access the design input from the knowledge base in order to accomplish its task. A design input that violates any constraints of the agent will be detected by the consistency agent. A message will then be sent to a method carrying a small program, which is a function of LISP responsible for conflict resolution. Alternatives will be presented to the users by a menu from which a design agent has to be selected. Alternatively, the system may ask the user to write his/her answer at the user prompt by giving necessary explanations. The new information does not violate the constraints of the agent and is included in the knowledge base in order to be utilized by other agents whenever required. The user interface informs the user of conflicts immediately. For instance, in the system,
the manufacturing capacity agent contains constraints on the production capacity. If a production quantity defined by the user is not between the constraints, the user will be informed by the user interface and a number of alternative solutions for sorting out this problem will be displayed.

The user will have to select from the suggested alternative or type an answer at the prompt window as shown in Fig. 11. The production quantity required by the user will be consistent information in the system if it does not violate the constraints of the manufacturing capacity agent. By using this system, consistent information flow between different tasks of the product life cycle is achieved without costly design iterations caused by conflicts in the design process.

3.5.1 Consistency monitoring in the system

As part of the design consistency approach, the system enables the designer to monitor inconsistencies in the system. This provides the designers with visual displays of constraint violation. In the developed system, design variables are restricted by sets of constraints, which are represented in terms of rules, frames and intervals.

The knowledge engineering environment (KEE) offers various types of image that can be used to change their attributes (colours, shapes and texts) subject to the alarm values to be highlighted and watched by the designers. However, this does not give enough information about inconsistencies. An explanation system is incorporated in the system to inform the designers of reasons for the violated constraints shown on the design consistency control panel. The typescript window is used for this purpose. When the system detects any inconsistency, it is shown on the panel, and then reasons for the conflict are given in textual format in the typescript window (Fig. 12).

3.6 User interface

In order to provide good interaction between the system and the user, a user-friendly design environment was developed. The development of the user interface was carried out using KEE toolkit and other Lisp programmes. KEE as an expert tool kit has good facilities for developing interactive user interfaces. The specification of the user interface was initially defined by Berrais [30]. The developed user interface is shown in Fig. 13 and consists of three major elements. The first element is the concurrent engineering menu, including a design for manufacture (DFM) button which activates a multiple choice menu for carrying out design tasks such as process selection, cutting tool selection and cost/time estimations. It also includes buttons for loading necessary files and starting the system and resetting the databases. In addition, there is a help menu for the user. By using a mouse, the menus and buttons can be easily activated. The second part of the user interface is the design consistency control panel on which various active images are placed and linked to the design variables. The active images change their attributes such as text and colour to show the design inconsistencies. Any changes made to the variables are reflected on the images linked to them. The third element of the user interface is the typescript window which provides the designers with textual displays of the reasons for the conflicts, results of the analysis carried out, suggestions for the resolution of the conflict and prompts for answers and queries.

3.7 Process optimization scenario

The optimization of machining processes should be carried out after a number of design analyses (i.e. manufacturing cell capability, feature types, dimensions and tolerances subject to production quantity, available
processes and their constraints). The rule-based algorithm for the evaluation and optimization of the machining processes consists of two major steps: machining process selection for the component and the optimization of these processes. Steps 1 to 4 select the feasible processes for each form feature according to material, lot size, tolerance, surface finish and feature type. The other steps include the accomplishment of various tasks (i.e. selection of suitable cutting tools, machine tools and optimum cutting parameters, and calculations of process variables such as MRR, lot time, set-up cost and tool cost). Comparison of the selected feasible processes subject to the process variables is also carried out. The different steps and tasks involved in the process optimization process are shown in detail in Fig. 14:

1. Select a material from the database.
2. Get the lot size of part or features.
3. Get a form feature from the part.
4. Select feasible processes for the feature, satisfying requirements of the part (tolerance, surface finish, feature type, etc.).
5. Take one of the possible processes.
6. Select the biggest possible diameter, shortest length and available and cheapest cutting tools for the selected process.
7. Select available machine tools and fixtures.
8. Select optimum cutting parameters: depth of cut, cutting speed, feed rate and cooling conditions.
9. Calculate feature volume MRR, lot time, tool cost for the lot size and set-up cost for the process.
10. Calculate total cost and time of the process.
11. If there are possible processes left to be analysed, 12. Go to 5 or else go to 13.
13. Compare the possible processes for each form feature with each other and eliminate the processes that have higher tool, process and set-up cost and time values than others.
14. If there is only one process for the form feature, consider it to be the most feasible process or else ask the user to select one process from the following list:
   (a) If there are form features left to be analysed,
   (b) Go to 3 or else 17.
   (c) Calculate the final process cost of the part.
   (d) Inform the user.
   (e) End.

The design of a cylinder head has been evaluated by the developed system as shown in Fig. 4. The results drawn
Fig. 12  Consistency modelling
Fig. 13  User interface
The integration of various issues of the product life cycle in a more consistent manner at the early design stages has been seen as one of the major achievements in this work. This integrated system enables users to design a product that satisfies most of the requirements arising from the life cycle issues at low cost, less lead time and higher quality. This can only be achieved through the use of the state-of-the-art of a fully integrated IT system. Available systems cannot offer a complete solution to the integrated product development owing to the limitations on consideration of requirements of various life cycle domains, which necessitates complex and timely interactions between these various domains. The integrated prototype constraint-based system presented in this article has taken care of most of the problems mentioned earlier, and thus facilitates successful implementation of concurrent product and process design. The system enables designers concurrently to design successful products in an interactive design environment with complete product and process design satisfaction. The integration of various issues of product life cycle such as part representation, product design specification, manufacturability analysis, process selection and optimization, manufacturing capacity checking, process cost/time estimation, machine and cutting tool selection and prototype testing has been achieved.

Information from different design areas was organized in the form of objects, rules and constraints in a knowledge base in order to achieve effective use of the life cycle information to carry out various design analyses. The knowledge base allows designers to include other life cycle requirements. A rule-based system has been developed to access the life cycle information in the knowledge base, and to carry out design analysis such as the evaluation of form features and the optimization of the selected feasible processes based on the given requirements (i.e. production volume, lot time, tool cost and process cost and time). The integration of production rules with object-oriented programming was established in the rule classes in order to reduce the size of rule classes, create more powerful rule application and make the system more flexible and efficient.

A user-friendly interface, which consists of menus for design analysis and a design consistency panel for monitoring inconsistencies, has been developed for providing designers with complete results of the analysis, consistency monitoring and conflict resolution. Since the developed system is very flexible, the involvement of other activities of the product life cycle can easily be incorporated in the design process. The process selection and optimization module, which is the major part of the developed system, provides designers with the design of products concurrently, selection of machining processes and evaluation and optimization of those processes.

The process selection and optimization module enables the designers to carry out real-time cost estimation and generation of feasible process plans, and deal with conflict

**Fig. 14 Various steps involved in machining process optimization**

from the system showed that drilling and end milling were the most cost effective processes for the two features (Fig. 15).

### 4 CONCLUSIONS

A prototype constraint-based system for the evaluation and optimization of machining processes is demonstrated in this article. The proposed system consisted of a form feature database, designer requirements, machining processes and constraints, an evaluation and optimization module and a user interface.
The prototype constraint-based system for machining processes is designed to handle various situations using a constraint-based system at the early design stages. It also encompasses a rule-based algorithm for estimation and optimization of machining processes.

The rule-based algorithm provides the evaluation of available processes for the features of parts in terms of user requirements and process time/cost. The designers are provided with a complete report on the results of the process selection, time/cost estimation and optimization in order to ensure the feasibility of the part. Since the results drawn from the system are promising, using this
system a significant reduction in the product cost and lead time could be achieved. The design consistency management module was developed and incorporated in the system in order to detect any design conflicts among the different design domains. This system enables designers in general to consider various critical tasks (overall coordination, control, consistency and data integrity). Design inconsistencies between different design domains are solved by a conflict resolution system.

This research work has contributed to implementing the concurrent engineering strategy from four perspectives: integration, optimization, information management and design consistency. Further research is currently being undertaken to develop the system further and make it more comprehensive.

REFERENCES


