INTERACTIVE PROCESS PLANNING IN THE AMRF

Peter F. Brown
Charles R. McLean

Factory Automation Systems Division
Center for Manufacturing Engineering
National Bureau of Standards
Gaithersburg, MD 20899

ABSTRACT

As more intelligent automated control systems are introduced into discrete parts manufacturing facilities, it will become increasingly difficult to maintain the centralized process planning systems in use today. A new approach is required that will permit distributed manufacturing operations planning via a network of cooperating, intelligent, process engineering systems. There are a number of reasons why manufacturing process decisions should be made locally by planning modules that are fully aware of a controller's current or expected capabilities. Expert planning modules should be developed for each controller or class of controllers that are or will be used in manufacturing installations.

To accomplish this goal of distributed, intelligent planning modules, work has started with the development of a semi-automatic interactive process planning system. This system has several unique features. First, a hierarchical planning system has been developed for multi-level factory architecture. Second, all activities within the factory are described by work elements. A work element is an activity at some level of the factory for which there are well- defined constraints. Third, standard interfaces have been defined to allow the passing of information between planning modules and controllers. These interfaces are used for the organization of the data and not for the data itself. Fourth, a semi-intelligent editor for the manipulation of these process planning data structures. These tools include editors for defining work elements and manipulating the process planning data structures. A graphic network editor is used for defining the "Precedence Graph" of a process plan. All system editors are based on windows and menu selections.

Interfaces to factory-wide databases for retrieval of information, such as raw stock and tooling, and CAD/Solid Modeling databases are under development. This last interface will serve three purposes: 1) the input of the initial part geometry to be manufactured, 2) the verification of changes to part geometry by the process engineer, and 3) the storage of intermediate geometries to be passed to other factory systems (inspection, machine tools, robots, vision systems, etc.). This paper describes research efforts at the National Bureau of Standards (NBS) by the staff of the Distributed Automated Process Planning System (DAPP) project to define and test this information processing architecture in the machine shop environment of the Automated Manufacturing Research Facility (AMRF).

1. INTRODUCTION

In designing a process planning system for the AMRF, the primary issue was not whether the system should employ variant or generative techniques. The most important concern was to identify the fundamental architectural concepts that would best support process planning in a small batch manufacturing facility where all production operations are under direct computer control. Important research questions deal with the functional relationships and the data interfaces between manufacturing control and planning systems: 1) How should planned tasks be specified to controllers? 2) How should alternatives be described? and 3) What formats should be used to pass data between the planning system and the controllers? The AMRF project involves developing a testbed for factory automation research to define and test the system interfaces between modules like process planning, geometric modeling, manufacturing control, data administration, network communications, and other factory subsystems. Within the AMRF, process planning is designed to be one of the primary programming tools of the factory. This paper describes the efforts of the AMRF process planning project to define robust interfaces to support both the future development of interactive process engineering tools and automated intelligent process planning systems.

Current Philosophies in Process Planning

There are two basic types of process planning systems in use today: variant and generative. Variant planning systems are based on a library of standard plans for different part families that a process engineer retrieves and edits, creating "variants" of basic plans. Generative planning systems employ expert system concepts, they reason using embedded knowledge and problem solving techniques to develop new plans. For a more detailed discussion of the state of the art of computer-aided process planning systems, see Chang and Wysk [1].

Variant systems typically rely on group technology classification and database management systems for their implementation. Standard process plans are developed for each family of parts produced and are stored in the database. When a new part enters the system, it is first classified by part family. The part classification code is used as a key to select a copy of the appropriate default plan from the database. This copy is then modified to reflect the specific processing required due to the unique characteristics of the new part. If a plan does not exist for the part's family, then a new default plan is created by an experienced process engineer and stored in the database system.

The technology that is required to implement this type of process planning system is readily available on main frame as well as personal computer systems. Indeed, almost all of today's commercial process planning systems employ variant techniques. With this approach most knowledge resides in the mind of the process engineer, the computer serves mainly as an organizing tool. Although intelligent generative systems are often more desirable, there are significant benefits that can be obtained from the variant approach. The development of a variant system forces an organization to study and classify the activities that it can perform in order to understand the part families it can produce in its shop. This exercise, in turn, reveals the kinds of equipment and labor skills that the shop really needs. But, there are some limitations in the variant approach. It can often be impractical if the shop produces small batches of widely varying parts. More time has to be spent defining new part families and modifying default plans. Furthermore, it does not capture the real knowledge or expertise of process engineers. The generative approach to process planning does address these issues.

The main thrust in process planning research today is in the area of generative systems, for some examples see [15,17]. In these systems, artificial intelligence is used to automatically create a plan for a new part. An expert problem solving system uses an internal process knowledge base and part specific data to generate new plans. This approach requires that a full product definition or part model is encoded in the system in a form that is accessible by the expert system software. This model should include geometry and topology, a tolerance model, and information about the functionality of a part. The knowledge base contains information gathered from process engineers on the how and why of making process decisions for various types of parts. Decisions are often keyed to the different types of features that are typically produced on parts. With this approach, the knowledge base becomes a repository of knowledge gained from the many years of experience of many process engineers. It also permits the separation of the process knowledge from the part data, facilitating data driven automation.

To date, fully generative process planning has proved to be an elusive goal, but there are some signs of progress. The biggest problems have included the representation of features (pocket, slots, holes), processes (drill hole versus bore hole), and sequencing information (make pocket before hole). Furthermore, the outputs produced by process planning systems are non-standard. That is, the organization of data into forms or structures such as routing and operations sheets differs from system to system. As a general rule, plans are meant to be interpreted by human readers, rather than by automated control systems. In the future, it will be essential that process planning interact more closely with automated control systems. Major questions with respect to the inputs and outputs of process planning systems must be resolved before fully computer-integrated intelligent manufacturing systems become a reality.

The AMRF process planning project is tackling questions that concern the fundamental role of process planning in automated manufacturing facilities. Important areas to be addressed include: 1) the definition and parameterization of activities or processes, 2) the development of standard definitions for both design and manufacturing features, and 3) the establishment of a data representation scheme that can be used to organize and exchange information between planning, control and other factory systems. The AMRF process planning project has developed a number of workable solutions in these areas.

AMRF Process Planning Concepts

A primary goal of Automated Manufacturing Research Facility (AMRF) which has been established at the National Bureau of Standards (NBS) is to develop a small batch manufacturing system to support research and experimentation in automated metrology and interface standards for the factory of the future [2,3,4,5,6]. Since process planning is expected to become one of the primary tools for programming automated factories, its system interfaces are of great interest. Unfortunately, the conventional views and implementations of process planning systems are inadequate to support such a factory. The research approach at NBS focuses on identifying basic concepts that would support the integration of process planning directly with the software and hardware of manufacturing process control systems.

Presently, the AMRF is comprised of six manufacturing workstations which perform both production and support functions. Each of the three machining workstations has a numerically controlled machine tool, a robot manipulator, flexible part fixturing systems and local storage for tools and materials [7,8]. Another station, cleaning and deburring, has two robots, cleaning equipment, and buffing wheels. The inspection workstation contains a robot, a coordinate measuring machine, and surface roughness characterization device. The last workstation level system, the material handling system [9], consists of two automatically guided vehicles (AGV), trays for parts and tooling, a storage and retrieval system, tray roller tables in the workstations, and a tender area for manual support activities. Finally, all workstations have a controller consisting of one or more small computer systems and associated software.

Other major factory systems found within the AMRF include: a cell control system, user interfaces for design and modeling, process planning and off-line programming systems, a data administration system and a communications network. The major difference between the systems found in the AMRF and in conventional advanced manufacturing systems, is the number of different systems vendors involved. Manufacturing subsystems were consciously chosen from many different vendors to shed light on the "plug compatibility" problems that would be faced by industrial system integrators.

A major effort is underway within the AMRF to integrate the factory systems, identified above, into a single automated manufacturing environment. This integration will be accomplished using some of the hierarchical task decomposition techniques and real-time sensory interactive control concepts originally outlined by the robotics project at NBS [5,18]. With this approach, all control modules are arranged in a hierarchy. Each controller takes commands from only one higher level system, but it may direct several others at the next lower level. Long range goals enter the system at the highest level and are decomposed into subgoals to be executed at that level or passed down as commands to the next lower level. Status information, based on real-time sensory data collection, is generated at each level and is passed up as feedback to the next higher level. The preparation of planning data, that will enable these hierarchical control systems to achieve their goals, is the primary role of process planning.

The AMRF process plan data structures are intended to be generic so that they can be used in a variety of manufacturing organizations from small shops to large factories. Process plan data structures have been defined, using formal language specification techniques, that can be transmitted electronically between planning and control computers. Although the formats are quite readable, they could easily be enhanced by print formatting routines to be made more suitable for human interpretation and execution.

By defining standard process planning data structures, an organization will be able to develop planning systems in a modular fashion. An interactive plan editing system can be developed initially. Later expert planning modules can be added without a change to basic data formats or execution system architectures. Another important benefit of standard data structures is that it permits the implementation of planning systems by multiple independent developers. It will also allow for the design of intelligent control systems that will be able to accept these standard process plans. By taking this approach, many organizations may be able to participate in the development of planning and control systems. Each developer could focus his efforts on developing specialized intelligent planning capabilities, building upon the programming work of others.

Our approach has focused on first defining process planning data structures that could eventually be used to construct a distributed generative process planning system. Such a system would involve the dynamic interaction between intelligent planning and control systems at each level within the

AMRF. A number of interface issues between the planning and control systems must be resolved. Some of the interface issues that fall within the realm of the process planning project include: 1) the development of a feature-based representation of part geometry to be used as an input to process planning, 2) the specification of a plan syntax to be used as a neutral file format for transferring plans out to target control systems, and 3) the definition of basic work elements, i.e. generic or specific manufacturing activities that each control system is capable of executing.

The work element is the basic procedural entity in the AMRF planning and control system. The work element is a function or activity which is carried out by a manufacturing control system at a particular level in the factory hierarchy. A work element has a name, a set of parameters, a duration, and a list of precedent steps numbers. The numbers identify the steps in the plan that must be performed prior to this one. Work elements are parameterized and organized into procedure specifications within process plans. The parameters of complex work elements, usually performed by higher level systems, refer to lower level process plans. These process plans specify the decomposition of the complex activity into simpler work elements supported at the next lower level in the hierarchy. Within controllers, work elements are implemented as subroutines that carry out error checking, database transactions, as well as physical changes to the manufacturing environment.

Generic Data Interfaces

A major goal of the AMRF project is the identification of generic functions and data structures for advanced manufacturing systems that could be used as a basis for the development of industry-wide interface standards. Generic interfaces, which are relevant to process planning, have already been defined and implemented within the AMRF to support interaction between a diverse set of applications processes. A communications mailbox protocol has gives AMRF applications processes access to each other over the communications network [10]. A control command-status protocol [10] has been developed which provides a means by which supervisory controllers can assign production work orders to subordinates and receive feedback status. A work order management system, described in [10], has been implemented in which process plans are used to specify the decomposition of complex jobs into simpler tasks [10]. A level independent neutral process plan file format has been developed for transferring this data between planning and control systems [12]. Process plans are deposited in a common database for later retrieval and execution by automated manufacturing control systems. A generic interface to the common database [13] has been created to give control systems ready access to required data, such as: command and status messages, work orders, process plans, control programs, geometry descriptions and other reference data.

Although there are many differences between the automated control systems found at each major level in the AMRF hierarchy, they all seem to have some functions and responsibilities that are characteristic of project managers. Hence, project management concepts have provided a foundation for defining the behavior of planning and control systems within the AMRF. Project managers, regardless of their level within an organization, tend to perform some generic planning and execution functions. Typical functions include: 1) work decomposition or problem reduction -the breakdown of complex activities into a interdependent network of simpler ones that can be routinely carried out by subordinates, 2) resource management - the identification, acquisition and allocation of required resources, and 3) estimation or prediction - the analysis necessary determine project cost, time and quality trade-offs.

Project managers often use network scheduling tools such as critical path method (CPM) or program evaluation and review technique (PERT) to define, sequence and monitor project activities. A detailed discussion of PERT, CPM, and other project management methodologies can be found in [11]. The data that is typically required by these systems includes: activity specifications and precedence relationships, resource requirements, time and cost estimates. With the exception of cost estimates, the process plan file structure is designed to convey this information to control systems.

A process plan is comprised of four major sections: 1) Descriptive Header - contains static index and summary data, 2) Parameters - lists all variables for which real values must be substituted at execution time, 3) Requirements List - identifies all resources to be used during the execution of the plan, and 4) Procedure Specification - describes all work elements, their precedence relationships, and their attributes and specific value bindings. The next sections are devoted to a discussion of the process plan format.

Procedure Specification

The Procedure Specification is probably the most important section of the process plan, it describes not only all of the activities or work elements to be performed, but gives information about their order of execution. This information can be represented as a precedence graph. Figure 1 shows the precedence graph for the machining operation to be performed on a part. This graph allows the process engineer to explicitly state that some steps may be done in any particular order, allowing for parallel activities, while other have a strict sequence or precedence relationships. The nodes of a precedence graph are the work elements. The graph structure used to represent process plans permits the specification of alternate activity sequences. Intelligent control systems can use this information to continue the manufacture of a part when some forms of error conditions arise. The control system can search through the precedence graph to see what other nodes or step can be performed while notifying a supervisor of any unresolvable problems, a major step in integrating sensory feedback with intelligent manufacturing planning and control systems.

The precedence graph in figure 1 represents the process plan for the part shown in figure 2. Figure 2a shows a part for which a process plan is to be written; the part is broken down into a feature graph [14,15], which defines features (such as pockets, grooves, holes) and the access. The access defines which features block or cover other features. Once the features are determined one can define the procedure specification as to how to produce the part. Figure 1 shows the order in which we wish to produce the work elements (nodes of the graph): INIT, CHAMFER_OUT, POCKET, GROOVE, CHAMFER_IN, HOLE, and CLOSE. These work elements correspond to the features defined in the feature graph. The precedence relations, as drawn in the precedence graph, can be interpreted to mean that after initializing the machine (INIT), the next step could be either CHAMFER_OUT, POCKET or GROOVE, in any order (in the feature graph these features do not interact, so they could be produced in any sequence). But before either the CHAMFER_IN, or the HOLES, can be produced, the POCKET operation had to be performed. It is important to point out here that the holes could have been produced before the pocket, but the process engineer decided that it would be best to produce the holes after the pocket. Thus the precedence concept is used to limit or structure the machining sequence. This graph can now be linearized by various constraints, such as minimizing tool changes, tolerance stack-up, etc.

This is the highest level that the process engineer will deal with a single part, in terms of the manufacturing features. These features will then be decomposed into a set of machining activities that are best suited for the constraints on a feature (such as its tolerance attributes). Using the previous example the hole feature may be produced by a simple twist-drilling operation. If the hole feature required tight positional and roundness tolerances, several machining steps might be needed, such as: center-drill, twist- drill, and reaming. Using the process planning editor, the process engineer will first define the part features. Then using an expert process selection module, the features will be decomposed into a set of machining process steps. The output of the expert system is in the process plan format [12]. This will allow the process engineer to modify the individual processes, as well as to monitor the specified processes.

The procedure specification contains the information about the sequence of the operations to be performed. Work element parameters reference hardware systems and software data objects used in the performance of a particular process. This information is consolidated into the Requirements List section of the process plan.

Requirements Lists

The requirements list section contains a list of all the hardware and software needed to execute the procedure specification we have just described. This structure has a similar function to that of a bill of materials. When a plan is executed in the AMRF, a controller can check to see that all items listed in the requirements list are available before executing the procedural steps of a process plan. This section of a process plan could also be used by the scheduler to determine that all items are available before the plan is even released for production. The requirements lists also identifies all other process plans referenced in the procedure specification.

In an effort to make the process engineers job easier, the generation of the requirements list can be done automatically. When the process engineer defines a work-element, there is a procedure for identifying items that will be added to the requirements list. Upon completion of the procedure specification, the system supervisor will query each node to ascertain what items it requires to perform its task, and these items are then added to the requirements list. Currently, there is only minimal checking for duplication of items. After the system has generated the requirements list, it is available for editing or viewing by the process engineer.

In the example given in Figure 1, the requirements lists would contain a list of tools, process plans, control programs (N/C, robot, or inspection), fixtures, robot grippers, etc. Figure 3 shows the major fields of each entity in the requirements list. The fields are a label, a descriptive name, a set of attribute-value pairs, and pointers to any sub-elements of an item. This pointer item is used to describe assemblies or complex items.

Parameters Section

The current implementation of the process planning system is an interactive system, process plans are prepared off-line. The parameters section allows the engineer to specify in a symbolic way that a particular item is to be used, but does not actually specify a serial number (i.e. specify plan variables). In a simple example, the process engineer wishes to specify that a 1/2" 2 fluted endmill should be used for a milling operation. At the time of plan creation the process engineer could identify this tool as $$tool- 001 (the syntax of a process plan has all parameters preceded by $$), and when the plan is being executed $$tool-001 will be replaced by the actual serial number of the physical tool. In this way the process planner can specify completely how a job should be done without overly constraining the execution. It permits the passing of information that is useful to the work-element software, but is not known at planning time, rather only at run-time.

Header Section

The header section contains certain bookkeeping information used to index or catalog the plan. As the planning system discussed in this paper is dynamic, not all of the entities listed here are fixed. Several of the fields in the header are used by the data administration system as keys for retrieval of plans. These fields are PLAN-ID, PLAN-VERSION, PLAN-TYPE, and PLAN-NAME. In addition there are other header fields that will be used to keep track of important information such as PROCESS-ENGINEER, PART- NUMBER, GT-CODE, ENGINEERING-DRAWING-#, etc (See figure 3).

In order to be consistent and save on the duplication of work needed to read process plans, all levels of process plans use the same internal format. It is important to reiterate here that these planning structures are used to organize the data, not to limit the specific data that appears in the process plan. As long as one accepts the process planning data structures and their associated formats, the user can define or associate any kind of functionality to work elements that he/she wishes. The next section will discuss the file format developed to exchange information between planning systems and other systems (controllers, databases, etc.).

2. FILE FORMAT

As part of interface standards work, a method has been developed for exchanging process plans between various systems. Using formal language specification techniques (Backus-Naur), a grammar has been defined for the process plan data structures. Using this grammar, an ASCII file containing the process plan can be generated (for an example see figure 3). This file can be passed between various computer systems, translated back into a control systems internal representation of activities to be performed. It is then used to sequence the part through manufacturing. To test our specifications we have developed and written parsers in several languages to construct the appropriate data structures. The file exchange format is quite human readable as we make liberal use of formatting when writing the ASCII file.

3. PROCESS PLAN HIERARCHY

The same basic structure is used for process plans at all levels of the factory. In the AMRF, currently only the lowest three levels of control are operational: Cell, Workstation and Equipment. The names that have been given to the classes of plans at these levels are, respectively: Routing Slips, Operation Sheets, and Instructions Sets (see Figure 4). The role of the plans at each level is described in subsequent sections.

Cell Routing Slips

Cell routing slips are used to coordinate the movement and processing of materials, parts, tools, and other needed items between and at workstations. A brief example will best illustrate this idea, (see figure 5). In the example the cell control system is told to deliver a tray of parts and one of tooling to the vertical workstation; the vertical workstation is told to receive the two trays, then to setup the tooling area, machine the lot of parts, takedown the tooling area, ship out the trays and, then to finally have the material handling system deliver the trays to some other AMRF system.

Each one of the nodes in the graph represent a work element. The node MACHINE_LOT will decompose into an activity at the next level in the AMRF, which is the workstation level. The process planning data package at this level is known as the Operation Sheet.

Workstation Operation Sheets

The next lower level of factory control is the workstation level. Process plans at this level are used to coordinate equipment level activities. The MACHINE_LOT work element, from the previous example, can be decomposed into an entire operation sheet. Figure 6 shows a simple sequence of MOVE_PART, MACHINE_PART, and MOVE_PART, which would be repeated for the number of parts that are in the lot. MOVE_PART, which is used for loading a part into a fixture, involves the coordination of the robot, machine tool, and fixturing system. Finally, this level of the process plan is decomposed into a sequence of tasks for the equipment to perform. The decomposition of the MACHINE_PART work element provides a good example of the next lower level, the Instruction Set.

Equipment Instruction Sets

The bottom level of the process planning hierarchy is the Instruction Set. This is a detailed sequence of operation for an equipment level to perform. The example given here is the MACHINE_PART work element that is carried out by the vertical milling machine. Primitive work elements appearing in the Instruction Set describes features such as pocket and holes that are to be produced. Figure 1 shows the machinable features and machine tool work elements that make up the part. The example has shown how high level tasks can be broken down into sucessfully smaller and smaller tasks using the process planning data structures.

4. THE AMRF PLANNING SYSTEM

The specification of a data flow model and neutral data interchange formats was a major step in the development of the AMRF interactive process planning system. The model of data flow between planning and control assumes that each controller in the hierarchy retrieves plans that have been placed in the common database by the process planning system (see Figure 3). The primary role of the process planning system, in this model, is to provide interactive tools for generating and storing process plans for new production parts which are later executed by the control systems. Specified inputs to the planning system include: process planning work orders assigned by facility control, initial and final part geometry specifications, definitions of controller work element capabilities, various kinds of reference data, and the plan editing decisions of a skilled process engineer. Outputs from the planning system include: work order status information for facility control, graphics displays for the engineering user, process plans which define manufacturing sequences for each control system involved in production operations, and part model specifications for each new intermediate geometry. This section describes the architecture and operation of the current implementation of the AMRF process planning system.

Lisp, Flavors, and the Lisp Machine

The process planning system was written entirely in Lisp using an object-oriented programming environment. The development plan called for first building an interactive process planning system around the work elements and the interfaces to computer control systems previously described. The more long range goals call for the development of expert planning modules. A decision was made to develop the system using a Lisp machine so that it would be easy to upgrade to more expert planning modules. Lisp, the primary language of the AI community for the development of such systems, has a number of advantages. It is an interactive language so changes can be tested almost immediately, instead of the edit-compile-debug cycle of more conventional languages. Tools for constructing friendly user interfaces, are provided that are mouse and menu- driven. To aid in software development, there are window and menu- based debugger and inspection tools. A language sensitive editor and an object oriented programming environment are also provided. As discussed earlier in the paper, the planning system's most fundamental concept is the work element, which is analogous to the operator concept from artificial intelligence problem solving systems. Each work element has a set of constraints, and when evaluated, makes a specified state change in the system. For a more detailed discussion of the subject see [16]. These work elements, when linked together, form a process plan which describes how to make a part. This representation can be supported in a robust way by the use of an object-oriented environment known as Flavors. In this environment one defines objects, giving them certain behavior. To activate an object a message is sent to that object asking it to invoke a method, (a procedure which changes the object data and/or initiates other messages).

Flavors, Object-Oriented Programming and the Planning System

Earlier we described how a precedence graph is used by the planning system to represent how manufacturing operations are to be performed. Internally, a directed graph is used to represent the precedence relationships. Each node in the graph represents a work- element, whose precedence relationship is defined between a node's parents and its children. In object-oriented programming, each node can be given a behavior, such as how to act as a node in the network. The current implementation of object nodes includes how to add or delete oneself within a network, display oneself, modify ones attributes and values, etc.

In the requirements lists section of the paper, the automatic generation of the requirements was described. This serves as a good way to illustrate the benefits of object oriented programming. In the methods of a work element is a list of requirements that must be specified to complete this task. When the process engineer asks the system to automatically generate the requirements list, an internal supervisor sends the requirements message to each work- element. They, in turn, respond by sending there requirements to the requirements-list supervisor. This supervisor then sorts the requirements list checking for duplicates, etc. In this way we were able to develop software that closely matches the way people conceptually handle such problems.

The implementation scheme employed also allows for modular system design in the construction of the user interface. Within the planning editors there are three major modules: 1) the internal data representation (a precedence graph), 2) a supervisor, and a 3) user interface manager. The process engineer interacts with the interface manager. The IM is a set of windows that display the current process plan. Items are mouse sensitive (when the mouse is moved around the screen these items are (highlighted). The user points at an item, clicks a mouse button, and a message is sent to the active window highlighting the chosen item. The addition of a new node to the precedence graph is a good example. The interface manager displays a list of all the valid work elements that the user can choose from. One of these items is then selected. A message is then sent to a supervisor describing the transaction to take place (i.e. add a new work element named "Do it" after the fourth step in the process plan). The supervisor then sends the appropriate messages to the internal data structures. Each process plan is an individual entity in the Lisp machine, so the supervisor keeps track of what is the active plan (the one actually being edited) and then it knows where the update message is to be sent. There is no limit on the total number of process plans that can be in the Lisp machine, but currently only one is active for editing. When the user wishes to change to a different plan, the user interface sends a message to the supervisor asking for the plan. The supervisor then gets from the planning system internals the list of applicable work elements on this level and the actual plan.

Through a series of transactions, the window is updated with this new plan, and the work-element that can be added to the plan are then displayed. This scheme was modularized in order that various modules could be distributed to different computers to implement a more distributed planning system.

5. SUMMARY

A process planning system has been implemented to demonstrate the concepts described in this paper. This system is currently being tested and evaluated within the AMRF. This last section of the paper will describe some of the major tools we have implemented, current work in progress, and finally the future direction we will be taking.

Current Implementation

The current implementation of the process planning system has several unique tools for the development of process plans. As described earlier, the work element is the fundamental object within the process planning system. The dynamic nature of the AMRF require an easy to use, flexible tool for creating, editing and viewing the work elements. This is the work element and requirements database. This data base allows AMRF projects to define the work elements and requirements for their workstations and equipment. The tool is graphically oriented, showing the hierarchy of the facility. Figure 7 illustrates the tool. The three levels of the AMRF are shown including the equipment associated with each workstation. The work element choices for the vertical mill are displayed in the menu. The last tool is a work element template editor. This tool is provided for control system implementors to define the work elements their systems can execute.

The output of this system is to generate lisp code to be used by the planning system. The second major set of tools is for the editing of actual process plans. These tools take shape in three areas; a top level for storage and retrieval of process plans for particular parts, a graphical editor for the development of the precedence graph, and a more text oriented tool for editing the individual values of a work elements attributes. The top level tool for keeping track of process plans allows the user to read plans in from files, define new process plans, edit existing plans or browse through all existing process plans. The structure of the system currently uses part names as the highest level. Each part can be opened to see its routing slips, operation sheets, and instruction sets. Each item is indented a certain amount to signify its level within the AMRF hierarchy. This tools resembles a hierarchical file system in a computer. Once the process engineer selects the plan to be edited there are two tools that can be used to develop plans. Figure 1 shows the tool referred to as a graphic-net editor. This tool is used at the more conceptual level of planning to lay out the high level of tasks to be performed. Another tool is used to fill in all of the attributes of a work element. This tool has a different set of windows for each of the major process planning internal data structures, procedure specification, requirements lists, and header section. When the process engineer is finished editing the process plan, it is written to a file using the file specification discussed earlier in the paper.

System Interfaces Under Development

There are currently three major subsystems under development, the interface to a CAD/ Solid modeling system, the data administration system (DAS) and the work order management system. All of these interfaces will be used to obtain information about the parts to be produced. The solid modeling interface will serve as a tool to get information about the part geometries to be produced including the dimensional and tolerancing information. It will also be useful for the process engineer to verify the changes to the part geometry, and to do high level interference checking against fixture, and other forbidden volumes. The interface to the DAS will be used to store process plans and to search for process plans within the DAS. It will also be used by the planning system to obtain information about the production capacity of the entire facility (obtain tooling reports, etc.). The interface to the work order management system will provide a tool by the facility level system to give planning jobs to the process planning system and for the planning system to report back on the status of process plans.

Future Work and Conclusions

This first implementation of the planning system has provided quite a learning experience for the staff of the process planning project. The use of Flavors, the lisp machine, development of a file exchange specification, and prototype expert planning systems has given insight into a number of strengths and weaknesses of our current approach. These lessons have pointed out a number of areas that need work in future implementations. A more robust internal representation is needed to allow multiple relationships to exists between work elements in the precedence graph. We have been exploring using commercially available expert system shells such as Knowledge Craft, or KEE. These tools would provide some form of portability across a variety of computer systems. A second major thrust is to design the planning system to be more closely tied in with control systems, so that the planning module gives tasks to a controller while exploring possible alternative paths. A final area of major interest is the development of expert planning modules. Currently we are testing a interface to SIPS [17] for the work of transforming features to process steps. We have been enhancing the knowledge base to reflect the process capabilities of the AMRF, and are using the system as a learning tool for the development of future expert planning modules.

In closing, we have developed a scheme for implementing an intelligent process planning system, and for interface this system to control system within the AMRF. With this information processing architecture, we will be laying the ground work for the next generation of manufacturing facilities.

Acknowledgment

The author acknowledges the efforts of the following people who contributed to the design and implementation of the AMRF process planning system: Steve Ray, Diana Gordon, Brian Drummond, and Mark Unger.

REFERENCES

(1) Chang, T. and Wysk, R., "An Introduction to Automated Process Planning Systems," Prentice-Hall, Englewood Cliffs, New Jersey, 1985.

(2) Simpson, J.A., Hocken, R.J. and Albus, J.S., "The Automated Manufacturing Research Facility

of the National Bureau of Standards", Journal of Manufacturing Engineering, 1, Vol. #1, (1982).

(3) Hocken, R. and Nanzetta, P., "Research in Automated Manufacturing at NBS", Manufacturing Engineering, 91, Vol. #4, (1983).

(4) Nanzetta, P., "Update: NBS Research Facility Addresses Problems in Setups for Small Batch Manufacturing", Industrial Engineering, June (1984).

(5) Furlani, C. et al., "The Automated Manufacturing Research Facility of the National Bureau of Standards", Proc. of the Summer Simulation Conference, Vancouver, BC, Canada, July 11-13, 1983.

(6) McLean, C., Mitchell, M. and Barkemeyer, E., "A Computing Architecture for Small Batch Manufacturing", IEEE Spectrum, May 1983.

(7) McLean, C.R., "The Vertical Machining Workstation of the AMRF: Software Integration", Proceedings of the 1986 Winter ASME Conference, Anahiem, California December 1986 (Submitted).

(8) Magrab, E., "The Vertical Machining Workstation of the AMRF: Equipment Integration", Proceedings of the 1986 Winter ASME Conference, Anahiem, California December 1986 (Submitted).

(9) McLean, Charles R., and Wenger, Carl E., "The AMRF Material Handling System Architecture," Proceedings of the Fifth Annual Control Engineering Conference, Rosemont, IL, May 1986.

(10) McLean, C.R., "AMRF System Architecture Document", Draft Technical Report, National Bureau of Standards, January 1986. (11) West, J.D., Levy, F.K., Management Guide to PERT/CPM,Prentice Hall, Englewood Cliffs, New Jersey, 1977.

(11) West, J.D., Levy, F.K., Management Guide to PERT/CPM, Prentice Hall, Englewood Cliffs, New Jersey, 1977.

(12) Brown, P., Gordon, D., "Process Plan Data Exchange Format", Draft Technical Document, July 1986.

(13) Barkemeyer, E., Mitchell, M. (NBS), Mikkilineni, K., Su, S., Lam, H. (Univ of Florida), "An Architecture for Disributed Data Management in Computer Integrated Manufacturing", NBSIR86-3312, January 1986.

(14) Henderson, M.R., "Extraction of Feature Information from Three Dimensional CAD Data", PhD. Thesis, Department of Mechanical Engineering, Purdue University.

(15) Hummel, K., Brooks, S.,"A Structure for the Representation of Manufacturing Features in the XCUT Process Planning System", Proceedings of th 1986 Winter ASME Conference, Anahiem, California, December 1986 (Submitted).

(16) Ray, S. "A Knowledge Representation Scheme for Processes in an Automated Manufacturing Environment", Man and Cybernetics Conference, October 1986 (submitted).

(17) Nau, D.S. and Chang, T.C., "Hierarchical Representation of Problem-Solving Knowledge in

a Frame-Based Process Planning System", Journal of Intelligent Systems, Vol. 1, (1986).

(18) Haynes, L., Wavering, A., "Real-Time Control System Software", Proceedings of IEEE International Conference on Robotics and Automation, Vol 3, 1896.

The NBS Automated Manufacturing Research Facility is partially supported by the Navy Manufacturing Technology Program.