PROCESS PLAN EXPRESSION, GENERATION, AND ENHANCEMENT FOR THE VERTICAL WORKSTATION MILLING MACHINE IN THE AUTOMATED MANUFACTURING RESEARCH FACILITY AT THE NATIONAL BUREAU OF STANDARDS

Dr. Thomas R. Kramer
Guest Worker, National Bureau of Standards, &
Research Associate, Catholic University

November 19, 1987

NBSIR 87 - 3678

Funding for the research reported in this paper was provided to Catholic University under Grant No. 60NANB5D0522 and Grant No. 70NANB7H0716 from the National Bureau of Standards.

Certain commercial equipment and software are identified in this paper in order to adequately specify the experimental facility. Such identification does not imply recommendation or endorsement by the National Bureau of Standards, nor does it imply that the equipment or software identified are necessarily the best available for the purpose.

CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

PROCESS PLAN EXPRESSION, GENERATION, AND ENHANCEMENT FOR THE VERTICAL WORKSTATION MILLING MACHINE IN THE AUTOMATED MANUFACTURING RESEARCH FACILITY AT THE NATIONAL BUREAU OF STANDARDS

I. INTRODUCTION

1. CONTENTS

This paper deals with process plans used for the milling machine in the Vertical Workstation (VWS) of the Automated Manufacturing Research Facility (AMRF) at the National Bureau of Standards. Chapter II gives background about process planning. Chapter III describes the specific process plan protocol used for the VWS milling machine. Chapter IV describes how process plans are generated automatically for the VWS milling machine. Chapter V describes how process plans for the VWS milling machine are enhanced automatically. The descriptions pertain to the system in use during the spring of 1987.

2. AUDIENCE

The paper is intended to be useful to people interested in concepts and technical details of the VWS, particularly AMRF personnel who are running the VWS or maintaining or improving the software for the VWS. The paper is intended to be useful also to other researchers in automated manufacturing. A knowledge of the computer language LISP is useful but not essential to reading this paper. Detailed documentation of the LISP functions that are involved with the systems described here is being prepared separately.

3. BRIEF VWS DESCRIPTION

The VWS is a computer-integrated automated machining workstation. It includes a control system, a computer-aided design system, an automatic process planning system, and an automatic nc-code generator. The principal machinery is a milling center (Monarch VMC-75 with a GE2000 controller) and a robot (Unimate 4070 with a Val II controller) to tend the milling center. There is quite a bit of ancillary hardware. The system is controlled from a microcomputer (Sun 3/160 with 6M memory, BW monitor). Running in stand-alone mode, it is possible to design and machine a simple metal part within an hour. The VWS may also be run as an integrated part of the AMRF. The workstation is described in more detail in [K&J1].

The software for the VWS is written in the computer language LISP. In this paper this software is called the VWS2 system. Six principal modules comprise the VWS2 system: the Production Management Operating System (the control system), the State Table Editor, the Equipment Program Generator, the Part Design Editor, the Process Planner, and the Data Execution module. Two of the VWS2 modules (the Automatic Process Planning module and the Data Execution module) deal with process plans for the milling machine. Process plans are the output of the Process Planning module and are part of the input to the Data Execution module.

To produce a part from scratch, the user sits at the Sun workstation and creates a design using the Part Design Editor. The Process Planner is then called to write a plan for how to machine a part of that design. Next NC-code is generated automatically from the design and the plan by the Data Execution module. Finally the user tells the control system to make the part. The control system coordinates the activities of the workstation equipment so that the part blank is loaded onto the milling machine, the NC-code is sent to the milling machine and executed (making the part), and the finished part is unloaded.

4. DESIGN PROTOCOL

The VWS2 system uses a feature-based design protocol. The design protocol is described in detail in [K&J2]. The design of a part is expressed as a list of features on a piece of stock. The piece of stock is always a rectangular block. The design protocol currently assumes that all features are being made from one side of the block.

Although all the features and subfeatures are purely geometric, they were selected to be included in the system on the basis of being features commonly found on machined parts that could be produced in one, or at most a very few, machining operations. Each feature and subfeature is a removed volume.

The design of a part is a purely geometric description of the shape of a part and gives no idea of what machining operations are required to make the part.

The primary features in the system in September, 1987 are: chamfer_out, groove, hole, pocket, straight_groove, text, contour_groove, contour_pocket, and side_contour. There are also subfeatures which may be made on the primary features: chamfer_out, chamfer_in, countersink, and thread. A feature is specified in the system by giving its name and the values of several parameters which specify its location, shape, and size.

The design protocol includes the use of "reference features". If feature A is to be made at the bottom of feature B, then one of the parameters of feature A is "reference_feature", and the value of that parameter is the feature number of feature B. Whenever B is the reference feature for A, the outline of feature A must fit within the outline of feature B, and the bottom of feature B must be flat (except in the case of concentric drill holes).

Although a design could be prepared according to the VWS2 design protocol using a text editor, the only reasonable way to make a design is by using the VWS2 Design Editor. The Design Editor is a friendly system which runs on a Sun computer that engages the user in a dialog to find out what the user wants to make and prepares the design document for the user. An example design is shown in Table 1.

5. AMRF CONTROL HIERARCHY

5.1. The Hierarchy

The AMRF has currently implemented three levels of hierarchical control: cell, workstation, and equipment. This hierarchy has been described in many papers by other authors. A good overview is given in [McLE]. The cell takes orders from a human user, gives orders to the workstations, and receives data from the workstations. The workstations take orders from the cell, give orders to the equipment, return data to the cell, and receive data from the equipment. The pieces of equipment receive orders from a workstation and return data to the workstation.

Cell, workstation, and equipment controllers are all expected to be able to get data from a global AMRF database and to send data to the database.

A cell is expected to be able to take all necessary actions to produce a finished part from scratch. In the AMRF the cell includes:
1. several machining workstations to cut metal,
2. a materials handling workstation to provide tools and workpieces,
3. a cleaning and deburring workstation, and
4. an inspection workstation.

A machining workstation is expected to be able to receive workpieces and tools and cut metal away from a workpiece to produce a final or intermediate part. A typical machining workstation includes:
1. a metal cutting machine (a turning center or machining center),
2. a robot to tend the metal cutter,
3. a roller table or other interface to materials handling, and
4. one or more fixturing devices for holding workpieces being cut. All the AMRF workstations have additional devices particular to the workstation.

A piece of equipment is expected to be able to carry out specialized functions such as cutting, holding, or moving workpieces. Equipment includes milling centers, robots, turning centers, powered fixtures, etc.

5.2. Carrying Out Plans

At each level of the AMRF hierarchy, the objectives of the level are met by having a control system carry out a process plan. The control systems have the ability to use parameterized plans. For example, if a tool is called for, the plan gives the tool type, and the id number of the tool is left as a parameter in the plan, to be filled in with the id number of a specific tool when the plan is being carried out. The control systems usually have the ability to make minor changes in the plans, as well. The basic plans, however, are usually developed outside of the control systems before any decision to actually carry out a plan is made.

The line between planning and control is quite thin. If a control system is able to call on a planning system that runs quickly enough to be used in real time, who is to say whether it is doing planning or exercising control? Any control system must make a decision before it can carry out the decision, and that decision-making can always be called planning. In a fast automated system, the distinction may be completely lost, but as long as it works, what it is called is not terribly important.

6. RELATED READING

This paper is one of about a dozen papers being prepared as part of the AMRF documentation to describe all aspects of the VWS. The others are [JUN], [KRA2], [KRA3], [KRA4], [K&J2], [K&S2], [KR&W], [LOVE], and [RUDD]. Other papers, prepared for professional meetings, also describe the VWS [KRA1], [K&J1], [K&S1], and [NA&J].

The brief description of the design protocol given above in section 4 is not adequate for a detailed understanding of the Process Planning module. The reader who wants details is referred to [K&J1] or [K&J2].

II. PROCESS PLANNING

1. OVERVIEW

1.1. Defining Process Planning

The term "process planning" does not have a simple universally accepted definition. In ANSI standard Z94.10 - 1972, it is defined as "a procedure for determining the operations or actions necessary to transform material from one state to another" [ANSI]. Chang and Wysk discuss the concept and say, "process planning could be defined as the act of preparing detailed instructions to produce a part" [CH&W]. This definition will suffice.

1.2. Contents of a Process Plan

The core of a process plan is a set of procedures that must be carried out in order to achieve the objectives of the plan. To be brief, we will call the procedures in a process plan "steps". In addition to a set of steps, a process plan may also have:
1. a list of requirements of tools and workpieces needed to carry out the plan,
2. administrative information such as the name of the plan, the version number, etc., and
3. a list of parameters used in the plan (if the plan is parametric).

Each step in a plan describes some operation to be carried out. It is convenient and feasible to give the description by giving the name of the operation (which we will call the "work element") and the values of several parameters required to describe the operation fully. Each work element has its own set of parameters, but a given parameter type may be used for many or all types of work elements.

What is supposed to occur when a step of a process plan is carried out must be commonly understood by the planner and the controller that carries out the plan.

1.3. What is a Process Planning Protocol?

A "process planning protocol" is a method of representing process plans. It should include a set of rules for how a process plan may be expressed and a description of how to interpret a plan. One method of giving the rules is through the use of Backus-Naur Form (BNF) notation. BNF notation allows precise and complete expression of the structure of a protocol, but it is very difficult to read. There is no rigorous method of describing how a plan is to be interpreted. Natural language is used.

This paper describes both the structure and interpretation of the VWS process planning protocol in English.

1.4. Precedent Steps

Traditionally, the steps in a process plan have been sequentially ordered and carried out in that order. It is usually possible, however, for the steps to be executed in some other order, with only the requirement that before a given step is carried out, some set of other steps must already have been completed. These other steps are called the "precedent steps" for the given step. A controller carrying out a plan which identifies the precedent steps for each step can then decide for itself how to sequence the steps to meet the precedence requirements.

In the rest of this paper, we will assume that the steps of a process plan are numbered sequentially as a convenient method of identification, but that execution of a plan is not necessarily carried out sequentially. Each step in a plan will be assumed to carry a list of precedent steps with it.

2. PROCESS PLANS IN THE AMRF

2.1. AMRF Process Planning Effort

The focal point for process planning in the AMRF is the Process Planning Project headed by Mr. Peter Brown in the Production Management Systems Group. The Group is headed by Mr. Charles McLean. The work of this project has been described in "Interactive Process Planning in the AMRF" by Mr. Brown and Mr. McLean [BR&M], and in "Research Issues in Process Planning at the National Bureau of Standards" [BR&R] by Mr. Brown and Dr. Steven Ray.

The AMRF Process Planning Project has developed an interactive process plan editor. It helps a user generate a process plan in AMRF standard format for the cell, any workstation, or any piece of equipment in the AMRF.

Dr. Dana Nau, working in the AMRF Process Planning Project, is developing an expert system for determining the machining operations needed to produce features on a part. The system is called SIPS (Semi-Intelligent Process Selector) and has been described in the paper "Hierarchical Abstraction of Problem-Solving Knowledge" [NAU].

2.2. AMRF Standard for Process Plans

The Process Planning Project has developed a standard protocol for expressing process plans in the AMRF. The standard is described in the paper "Process Plan File Format" by Dr. Ray and Mr. McLean [RA&M]. The AMRF standard is applicable to all levels of the AMRF control hierarchy (cell, workstation and equipment). The standard is given in BNF notation.

The standard prescribes both overall form and notation for expressing the plan. A description of the standard, as it applies to the VWS milling machine, is given below. This paper is not intended to prescribe how the standard applies to any part of the AMRF, but only to describe how it has been implemented for the VWS milling machine. Readers are referred to "Process Plan File Format" and to other papers of the Process Planning Project for prescriptive material.

In order that the same format may be used for all parts of the AMRF, the standard for process plans necessarily omits prescribing work elements. The cell, each workstation, and each piece of equipment, may have its own set of work elements.

2.3. Process Planning in the Vertical Workstation

The Vertical Workstation uses three types of process plans: workstation level plans, plans for the milling machine, and plans for the robot. This paper deals only with plans for the milling machine.

As configured in test runs in the spring of 1987, the VWS2 system may produce parts in three ways:
1. by executing stored NC-code,
2. by feeding a stored milling machine process plan and a design into the Data Execution module and then executing the NC-code created by the module,
3. by feeding a design into the Process Planning module and then feeding the output plan and the design into the Data Execution module and executing the NC-code created by that module. In all three cases, a stored workstation level process plan is used by the workstation controller.

When running in the third mode, the process planning done by the system may be completely hidden from the user. Viewed as a black box, the input to the system is a design and a piece of metal stock, and the output is a machined part.

III. PROCESS PLAN PROTOCOL FOR THE VWS MILLING MACHINE

1. OVERVIEW

1.1. Formats

To be usable in the VWS2 system, the format of a milling machine process plan in the LISP environment must be a LISP property list. Outside of the LISP environment there are two formats for process plans which are usable by the VWS. The first is an implementation for the VWS of the AMRF standard, which we will call the "VWS-standard" format. The second is a LISP-readable format. A knowledge of LISP will be necessary to understand some details of the LISP-readable format.

There is no AMRF standard for which work elements may be used in a process plan for a vertical machining center, so the details of individual work elements in the VWS-standard format are specific to the VWS. Many details of the VWS-standard format, however, do come from the AMRF standard. In the discussion below we have indicated for some (but not all) details whether the source of the detail is the AMRF standard.

In order to get into the LISP environment as a property list, a process plan which is LISP-readable may be "loaded". A process plan which is in VWS-standard format may be read in by a special reading function. If a process plan is already in the LISP environment, it may be printed out on paper or on disk in either format.

The general structure of the two formats is very similar. In each case the process plan includes a header section (principally administrative information), a requirements section (principally a list of tools needed for the milling machine), and a procedures section (principally a list of machining operations to be carried out). The VWS-standard format has a parameters section which is not included in the LISP-readable format. The print routine which prints the VWS-standard format, however, creates a parameters section. The reading and printing routines are such that if a process plan in proper format is resident in the LISP environment (regardless of the source of the plan) and the plan is:

1. printed from the LISP environment to file A in one format,
2. removed from the LISP environment,
3. restored in the LISP environment by reading in file A,
4. printed from the LISP environment to file B in the other format,
5. removed from the LISP environment a second time,
6. restored to the LISP environment a second time by reading in file B, and
7. printed from the LISP environment to file C in the first format,

then files A and C will be identical.

One awkward difference between the two formats is that the VWS-standard format requires use of upper case letters only (except in strings) while the LISP-readable format was developed entirely in lower case but had no explicit requirement for upper or lower. The VWS-standard requires upper case to satisfy AMRF database reports requirements, not because of the AMRF standard for process plans. This situation has been dealt with by converting all letters to upper case for the VWS-standard and to lower case for the LISP-readable. To avoid possible problems, a VWS-standard format plan should contain no lower case letters, even in strings, and a LISP-readable plan should contain no upper case letters.

Section 2 of this chapter describes detailed format requirements of process plans for the VWS milling machine. In general, the VWS-standard may be thought of as a "proof of the pudding" for the AMRF standard, in that it shows that the AMRF standard is workable for a specific machining center. Details of VWS-standard format plans are intended to be in compliance with the AMRF standard in most cases.

In some cases, however, the VWS-standard has requirements in addition to those of the AMRF standard. In other cases, the VWS2 system can deal with formats slightly at variance with the AMRF standard. For example,
1. The AMRF standard requires keywords, values, and parameter names to be 19 characters or fewer. The VWS-standard does not have a limit.
2. Carriage returns, line feeds and form feeds are not significant in the AMRF standard (they are just varieties of white space), but they are significant in some cases in the VWS-standard.

Other such cases are described below. All of the variances could be eliminated by further programming. None represents a logical impasse.

1.2. Steps

The heart of a process plan is the procedures section. This is a list of steps to be carried out. The first step is always initialize_plan and the last always close_plan. All the steps in between are machining operations. The VWS2 system currently supports 19 machining operations in addition to initialize and close. All but three of the machining operations are for cutting metal. Each machining operation step has a name, a step number, and several parameter-value pairs that make up the description of the step. Information about these operations is kept in the LISP environment in a "machine_ops" database.

The parameters for a machining operation usually include the feature number of a feature from the design. This serves as a pointer to geometric information about the feature, such as its depth or its center. The design itself is identified in the header of the process plan and is essential to the meaning of the plan. Because of this relation between plan steps and design features, it is feasible to have a process plan for only partial machining of a design. A part made according to such a plan would have some but not all of the features of the part specified in the design.

1.3. Process Plan Enhancement

In order to have a process plan be as broadly useful as possible, it is desirable that certain information NOT be in the plan and that the system which executes the plan have the freedom to make alterations to the plan. The plan is then altered (or enhanced) by the executing system just before it is executed, according to the user's desires and the workstation conditions prevailing at the time of execution.

One type of data which is best omitted until execution is the changer slot number of a tool. That way a single plan may be used for many different machine setups, as long as all the required tools are in some changer slot.

It is convenient if a single plan can be used for different initial workpieces. This is done in the VWS in three situations:
1. If a workpiece is too tall (but the right length and width). In this case it may be milled down to the correct height.
2. If a workpiece has been machined once and turned over so that it has a "slab" on top. In this case the slab may be milled off.
3. If a workpiece already has some of the features in the design in it. In this case the steps in the plan that would make the features that already exist are deleted.

It is also convenient to use a single plan for different fixturings of the workpiece.

Some users like to have speeds, feed rates, and pass depths included in a process plan, while other users prefer to omit those items and have them determined at the time NC-code is written. The VWS2 system gives the user a choice.

The Data Execution module has the capability to insert changer slot numbers in the plan, handle the three variations in workpiece geometry just described, handle different fixturings, and calculate speeds, feed rates, and pass depths. How this is done is described in more detail in Chapter V. For now, the important point is that the Data Execution module accomplishes these things by enhancing the process plan.

1.4. Geometric Information

It is feasible to handle geometric information in the steps of a process plan for a milling machine in two ways:

1. Geometric information may be placed directly in a step. For example, if a straight groove is to be made, the coordinates of the endpoints of the groove could be placed in the machining step for milling the groove.

2. Geometric information may be referenced indirectly by putting a pointer to it in a step. For example, if a straight groove is feature 3 in a design, the design may be designated in the process plan, and the plan step that calls for milling the groove may simply reference feature three.

The VWS2 system uses the second method whenever the geometric information is available in a plan. By using a pointer, minor changes in a design may be made without requiring changes in a plan for machining a part of that design. Using a pointer also eliminates having duplicate geometric information in two places. Using a pointer requires that both the design and the plan be available when the plan is to be executed, however.

If a process plan step is independent of a feature in a design (in a zero-setting operation, for example), then it is necessary to carry geometric information in the process plan step.

In the VWS2 system, the "machine_ops" database carries a list for each operation of those parameters whose values are to be extracted from the design and those whose values are to be extracted from the plan. This information is used by the NC-code writing subsystem.

At least two variations in this method of handling geometric information would be feasible but have not been tried in the VWS2 system.

1. Geometric information could be transferred from the design to the enhanced process plan when the enhanced plan is created.

2. The process plan could be allowed to carry either a pointer or direct geometric information (or both, as long as the system has a method of deciding which to use in case of duplication).

There is an AMRF standard for part designs. It uses boundary representation rather than feature-based representation. Part designs in the AMRF standard format cannot currently be used by the VWS2 system. A method of using an AMRF standard format design in the VWS would be to parse the boundary representation into a feature-based representation (either automatically or on a user-interactive system) and then use the feature-based representation.

1.5. Examples

Table 1 shows an example of the design of a part. We will call it the XYZ part for future reference. The design is given according to the VWS design protocol in LISP-readable format. A picture of the XYZ part prepared by the VWS Part Design Editor is shown in Figure 1. The XYZ part is a demonstration part containing at least one of each feature and subfeature type in the system. It has no mechanical function.

Table 2 shows a process plan in LISP-readable format prepared by the VWS2 automatic process planner for making the XYZ part. Table 3 contains the same process plan printed by the VWS2 system in VWS-standard format. Table 4 shows an enhanced version of the same plan in LISP-readable format. The enhanced version can be printed in VWS-standard format. The enhanced version has two more steps than the other two versions, one to cut a too-tall workpiece down to size, another to establish z-zero with respect to the top of the part. The enhanced version also has several additional parameters in each step and an additional tool requirement. The enhanced version includes 17 of the 21 work elements in the system.

The NC-code for the XYZ part which was generated by the VWS2 system from the design and plans shown in the tables is given in Appendix A.

1.6. Applicability

Although the process plan protocol described here was created for the VWS milling machine, plans prepared according to the protocol could be used by any milling machine able to carry out the 21 work elements contained in the protocol. With the possible exception of three zero-setting work elements, most modern NC-controlled milling machines have the capability to carry out the process plans generated in the VWS. Thus the plans themselves would not need modification. In this sense the VWS process plan protocol is machine independent.

However, in order to use a process plan to generate NC-code automatically for some other machine, modifications in the Data Execution module would be needed. This might be as easy as a revision of the printing routine in that module, or it might require deeper structural changes in the code-writing subsystem.

Table 1. LISP-Readable XYZ Part Design Document

```
(setplist 'xyz
'(features  (features
  1  (1  feature_type      side_contour
       depth                 0.2
       corners          (corners
           1  (1  x 5.25  y 2.8  radius 0)
           2  (2  x 5.75  y 2.8  radius 0)
           3  (3  x 5.75  y 0.2  radius 0)
           4  (4  x 5.25  y 0.2  radius 0)))

  2  (2  feature_type      text
       text                  x
       font                  round
       lower_l_x             5.38
       lower_l_y             2.1
       height                0.4
       depth                 0.015
       line_width            0.1187434)

  3  (3  feature_type      text
       text                  y
       font                  round
       lower_l_x             5.38
       lower_l_y             1.3
       height                0.4
       depth                 0.015
       line_width            0.1187434)

  4  (4  feature_type      text
       text                  z
       font                  round
       lower_l_x             5.38
       lower_l_y             0.5
       height                0.4
       depth                 0.015
       line_width            0.1187434)

  5  (5  feature_type      side_contour
       corners          (corners
           1  (1  x -0.75  y 3  radius join_back)
           2  (2  x  5.25  y 2.25  radius 0.27)
           3  (3  x  5.25  y 2.8   radius 0)
           4  (4  x  5.75  y 2.8   radius 0)
           5  (5  x  5.75  y 0.2   radius 0)
           6  (6  x  5.25  y 0.2   radius 0)
           7  (7  x  5.25  y 0.75  radius 0.27)
           8  (8  x -0.75 y 0 radius join_ahead)
           9  (9  x  0.75  y 1     radius 0.27)
           10 (10 x 0.75  y 2     radius 0.27))
       depth                 0.75
       reference_feature     1)

  6  (6  feature_type      hole
       center_x              0.875
       center_y              2.375
       diameter              0.1719
       depth                 0.8
       bottom_type           conical
       thread_diameter       0.19
       threads_per_inch      24
       thread_depth          0.6
       countersink_diameter  0.35
       reference_feature     1)

  7  (7  feature_type      hole
       center_x              0.875
       center_y              0.625
       diameter              0.1719
       depth                 0.8
       bottom_type           conical
       thread_diameter       0.19
       threads_per_inch      24
       thread_depth          0.6
       countersink_diameter  0.35
       reference_feature     1)
```

```
8  (8  feature_type          pocket_corners
       upper_l_x               1.45
       upper_l_y               2
       lower_r_x               4.5
       lower_r_y               1
       depth                   0.3
       corner_radius           0.3
       chamfer_in_depth        0.04
       reference_feature       1)

 9  (9  feature_type          contour_groove
       corners          (corners
          1  (1  x 1.25  y 2.57  radius 0.2)
          2  (2  x 5     y 2.11  radius 0.2)
          3  (3  x 5     y 0.87  radius 0.2)
          4  (4  x 1.25  y 0.43  radius 0.2))
       depth                   0.1
       width                   0.125
       bottom_type             flat
       reference_feature       1)

 10 (10 feature_type          hole
       center_x                3.0
       center_y                1.5
       diameter                0.5
       depth                   thru
       reference_feature       8)

 11 (11 feature_type          groove
       upper_l_x               2.875
       upper_l_y               1.85
       lower_r_x               4.25
       lower_r_y               1.15
       depth                   0.3
       width                   0.25
       corner_radius           0.2
       bottom_type             round
       chamfer_in_depth        0.05
       chamfer_out_depth       0.04
       reference_feature       8)

12 (12 feature_type      contour_pocket
       corners          (corners
          1  (1 x 1.25  y 1.9   radius 0.14)
          2  (2 x 2.0    y 1.5   radius join_back)
          3  (3 x 1.25  y 1.1   radius 0.14)
          4  (4 x 2.75  y 1.1   radius 0.14)
          5  (5 x 2.0   y 1.5  radius join_ahead)
          6  (6 x 2.75  y 1.9   radius 0.14))
       depth              0.3
       reference_feature 8)

13 (13 feature_type      straight_groove
       x1                      2.2
       y1                      1.8
       x2                      3.0
       y2                      1.5
       depth                   0.1
       width                   0.125
       bottom_type             flat
       reference_feature       8)

14 (14 feature_type      straight_groove
       x1                      2.2
       y1                      1.2
       x2                      3.0
       y2                      1.5
       depth                   0.1
       width                   0.125
       bottom_type             flat
       reference_feature       8))

header    (header
  material           aluminum
  design_id          xyz
  block_size      (block_size
       length             6
       width              2.95
       height             1.45)
  description "demo part")))
```

Figure 1. Drawing of XYZ Part

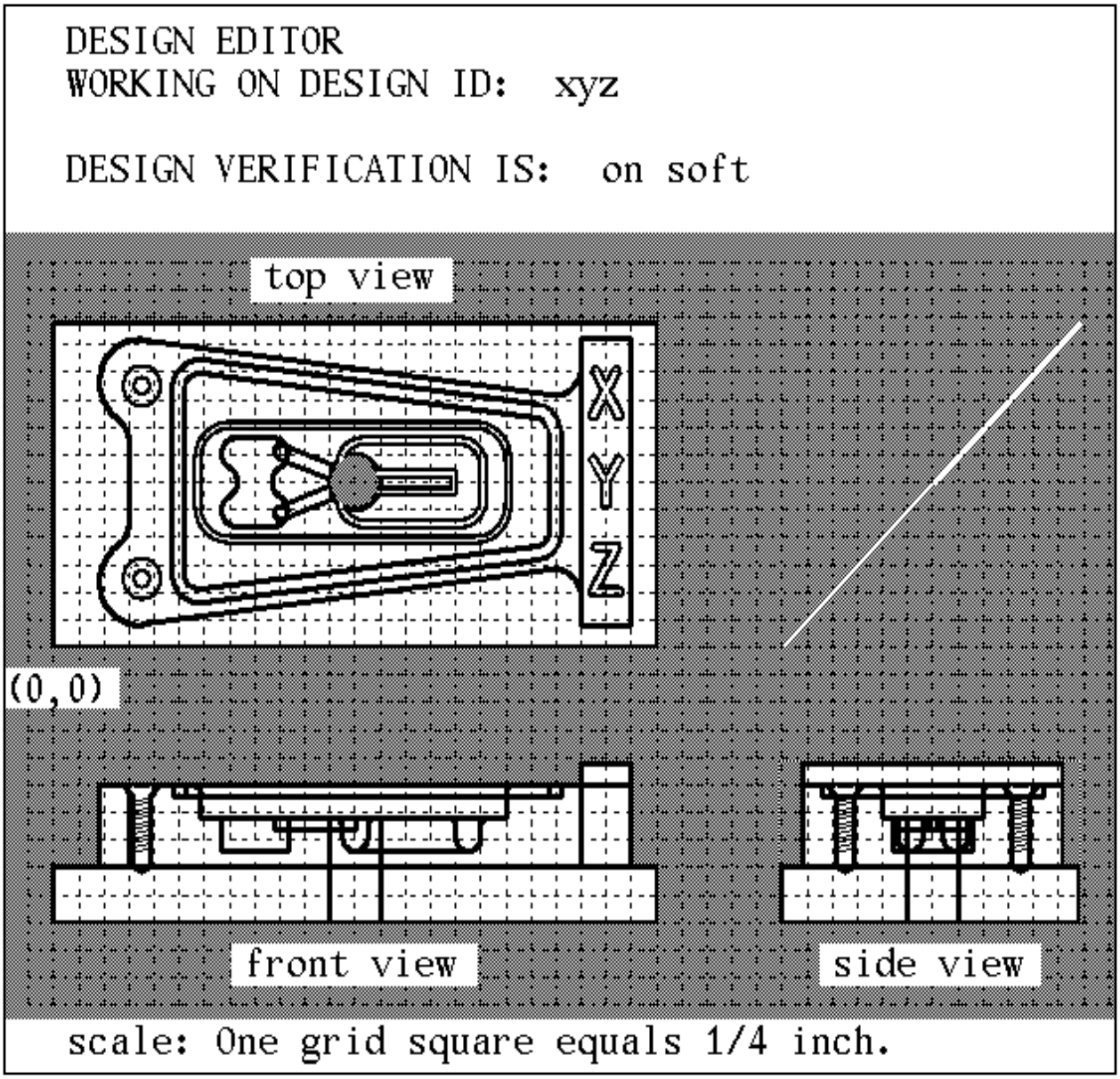


Table 2. LISP-Readable Process Plan for XYZ Part

```
(setplist 'xyz_plan
'(header   (header   plan_id   xyz_plan
                    design_id xyz
                    material  aluminum)
 steps    (steps
  1 (1  work_element    initialize_plan
      prog_name       demo part)
  2 (2  work_element    set0_corner
      tool_type_id    probe_0.25
      corner          1
      x_offset        0.0
      y_offset        0.0
      near_x          17.3
      near_y          7.45
      precedent_steps  (1 ))
  3 (3  work_element    mill_side_contour
      feature_id      1
      tool_type_id    end_mill_1.0_2_ab
      precedent_steps  (2 ))
  4 (4  work_element    mill_text
      feature_id      4
      tool_type_id  ball_nosed_end_mill_0.25_4_bs
      precedent_steps  (3 ))
  5 (5  work_element    mill_text
      feature_id      3
      tool_type_id  ball_nosed_end_mill_0.25_4_bs
      precedent_steps  (4 ))
  6 (6  work_element    mill_text
      feature_id      2
      tool_type_id  ball_nosed_end_mill_0.25_4_bs
      precedent_steps  (5 ))
  7 (7  work_element    mill_pocket
      feature_id      8
      tool_type_id    end_mill_0.5625_2_ab
      precedent_steps  (6 ))
  8 (8  work_element    mill_side_contour
      feature_id      5
      tool_type_id    end_mill_0.5_2_ab
      precedent_steps  (7 ))
  9 (9  work_element    mill_contour_groove
      feature_id      9
      tool_type_id    end_mill_0.125_2_ab
      precedent_steps  (8 ))
 10 (10 work_element    drill_hole
      feature_id      6
      tool_type_id    drill_0.1719_2_abs
      precedent_steps  (9 ))
 11 (11 work_element    drill_hole
      feature_id      7
      tool_type_id    drill_0.1719_2_abs
      precedent_steps  (10 ))
 12 (12 work_element    machine_chamfer_in
      feature_id      8
      tool_type_id    chamfer_0.375_3_abs
      precedent_steps  (11 ))
 13 (13 work_element    machine_countersink
      feature_id      7
      tool_type_id    countersink_0.75_1_ab
      precedent_steps  (12 ))
 14 (14 work_element    machine_countersink
      feature_id      6
      tool_type_id    countersink_0.75_1_ab
      precedent_steps  (13 ))
 15 (15 work_element    tap_thread
      feature_id      7
      tool_type_id    tap_0.19_0_abs
      precedent_steps  (14 ))
 16 (16 work_element    tap_thread
      feature_id      6
      tool_type_id    tap_0.19_0_abs
      precedent_steps  (15 ))
 17 (17 work_element    mill_contour_pocket
      feature_id      12
      tool_type_id    end_mill_0.25_2_ab
      precedent_steps  (16 ))
 18 (18 work_element    mill_straight_groove
      feature_id      14
      tool_type_id    end_mill_0.125_2_ab
      precedent_steps  (17 ))
 19 (19 work_element    mill_straight_groove
      feature_id      13
      tool_type_id    end_mill_0.125_2_ab
      precedent_steps  (18 ))
 20 (20 work_element    mill_groove
      feature_id      11
      tool_type_id  ball_nosed_end_mill_0.25_4_bs
      precedent_steps  (19 ))
 21 (21 work_element    drill_hole
      feature_id      10
      tool_type_id    drill_0.5_2_abs
      precedent_steps  (20 ))
 22 (22 work_element    machine_chamfer_in
      feature_id      11
      tool_type_id    chamfer_0.375_3_abs
      precedent_steps  (21 ))
 23 (23 work_element    machine_chamfer_out
      feature_id      11
      tool_type_id    chamfer_0.375_3_abs
      precedent_steps  (22 ))
 24 (24 work_element    close_plan
      precedent_steps  (23 )))
tool_requirements  (probe_0.25
      ball_nosed_end_mill_0.25_4_bs
      end_mill_0.5625_2_ab          end_mill_0.5_2_ab
      end_mill_0.125_2_ab           drill_0.1719_2_abs
      chamfer_0.375_3_abs           end_mill_1.0_2_ab
      countersink_0.75_1_ab         tap_0.19_0_abs
      end_mill_0.25_2_ab            drill_0.5_2_abs)))
```

Table 3. AMRF Standard Process Plan for XYZ Part

```
--PROCESS_PLAN--

--HEADER_SECTION--
  PLAN_ID         := XYZ_PLAN;
  PLAN_VERSION    := 1;
  PLAN_TYPE       := INSTRUCTION_SET;
  DESIGN_ID       := XYZ;
  MATERIAL        := ALUMINUM;
  PROCESS_ENGINEER := "VWS2 AUTO  PROCESS PLANNER";
--END_HEADER_SECTION--

--PARAMETERS_SECTION--
  $$WORKPIECE     : WORKPIECE;
  $$TOOL_SET      : TOOL_SET;
  $$TOOL1         : TOOL;
  $$TOOL2         : TOOL;
  $$TOOL3         : TOOL;
  $$TOOL4         : TOOL;
  $$TOOL5         : TOOL;
  $$TOOL6         : TOOL;
  $$TOOL7         : TOOL;
  $$TOOL8         : TOOL;
  $$TOOL9         : TOOL;
  $$TOOL10        : TOOL;
  $$TOOL11        : TOOL;
  $$TOOL12        : TOOL;
--END_PARAMETERS_SECTION--

--REQUIREMENTS_SECTION--
<<1>>  WORKPIECE
  ( WORKPIECE_ID   => $$WORKPIECE);

<<2>>  TOOL_SET
  ( TOOL_SET_ID    => $$TOOL_SET,
    COMPONENTS     => (3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14));

<<3>>  TOOL
  ( TOOL_TYPE_ID   => PROBE_0.25,
    TOOL_ID        => $$TOOL1,
    COMPONENT_OF   => 2 );

<<4>>  TOOL
  ( TOOL_TYPE_ID   => END_MILL_1.0_2_AB,
    TOOL_ID        => $$TOOL2,
    COMPONENT_OF   => 2 );

<<5>>  TOOL
  ( TOOL_TYPE_ID   => BALL_NOSED_END_MILL_0.25_4_BS,
    TOOL_ID        => $$TOOL3,
    COMPONENT_OF   => 2 );

<<6>>  TOOL
  ( TOOL_TYPE_ID   => END_MILL_0.5625_2_AB,
    TOOL_ID        => $$TOOL4,
    COMPONENT_OF   => 2 );

<<7>>  TOOL
  ( TOOL_TYPE_ID   => END_MILL_0.5_2_AB,
    TOOL_ID        => $$TOOL5,
    COMPONENT_OF   => 2 );

<<8>>  TOOL
  ( TOOL_TYPE_ID   => END_MILL_0.125_2_AB,
    TOOL_ID        => $$TOOL6,
    COMPONENT_OF   => 2 );

<<9>>  TOOL
  ( TOOL_TYPE_ID   => DRILL_0.1719_2_ABS,
    TOOL_ID        => $$TOOL7,
    COMPONENT_OF   => 2 );

<<10>>  TOOL
  ( TOOL_TYPE_ID   => CHAMFER_0.375_3_ABS,
    TOOL_ID        => $$TOOL8,
    COMPONENT_OF   => 2 );

<<11>>  TOOL
  ( TOOL_TYPE_ID   => COUNTERSINK_0.75_1_AB,
    TOOL_ID        => $$TOOL9,
    COMPONENT_OF   => 2 );

<<12>>  TOOL
  ( TOOL_TYPE_ID   => TAP_0.19_0_ABS,
    TOOL_ID        => $$TOOL10,
    COMPONENT_OF   => 2 );

<<13>>  TOOL
  ( TOOL_TYPE_ID   => END_MILL_0.25_2_AB,
    TOOL_ID        => $$TOOL11,
    COMPONENT_OF   => 2 );

<<14>>  TOOL
  ( TOOL_TYPE_ID   => DRILL_0.5_2_ABS,
    TOOL_ID        => $$TOOL12,
    COMPONENT_OF   => 2 );
--END_REQUIREMENTS_SECTION--

--PROCEDURE_SECTION--
<<1>>  INITIALIZE_PLAN
  ( PROG_NAME      => "DEMO PART",
    TIME           => "0000:01:00:00" );

<<2>>  SET0_CORNER
  ( TOOL_TYPE_ID   => PROBE_0.25,
    CORNER         => 1,
    X_OFFSET       => 0.0,
    Y_OFFSET       => 0.0,
    NEAR_X         => 17.3,
    NEAR_Y         => 7.45,
    PREC_STEPS     => (1),
    TIME           => "0000:01:00:00" );
```

```
<<3>>  MILL_SIDE_CONTOUR
  ( FEATURE_ID    => 1,
    TOOL_TYPE_ID  => END_MILL_1.0_2_AB,
    PREC_STEPS    => (2),
    TIME          => "0000:01:00:00" );

<<4>>  MILL_TEXT
  ( FEATURE_ID    => 4,
            TOOL_TYPE_ID          =>    BALL_NOSED_END_-
MILL_0.25_4_BS,
    PREC_STEPS    => (3),
    TIME          => "0000:01:00:00" );

<<5>>  MILL_TEXT
  ( FEATURE_ID    => 3,
            TOOL_TYPE_ID          =>    BALL_NOSED_END_-
MILL_0.25_4_BS,
    PREC_STEPS    => (4),
    TIME          => "0000:01:00:00" );

<<6>>  MILL_TEXT
  ( FEATURE_ID    => 2,
            TOOL_TYPE_ID          =>    BALL_NOSED_END_-
MILL_0.25_4_BS,
    PREC_STEPS    => (5),
    TIME          => "0000:01:00:00" );

<<7>>  MILL_POCKET
  ( FEATURE_ID    => 8,
    TOOL_TYPE_ID  => END_MILL_0.5625_2_AB,
    PREC_STEPS    => (6),
    TIME          => "0000:01:00:00" );

<<8>>  MILL_SIDE_CONTOUR
  ( FEATURE_ID    => 5,
    TOOL_TYPE_ID  => END_MILL_0.5_2_AB,
    PREC_STEPS    => (7),
    TIME          => "0000:01:00:00" );

<<9>>  MILL_CONTOUR_GROOVE
  ( FEATURE_ID    => 9,
    TOOL_TYPE_ID  => END_MILL_0.125_2_AB,
    PREC_STEPS    => (8),
    TIME          => "0000:01:00:00" );

<<10>>  DRILL_HOLE
  ( FEATURE_ID    => 6,
    TOOL_TYPE_ID  => DRILL_0.1719_2_ABS,
    PREC_STEPS    => (9),
    TIME          => "0000:01:00:00" );

<<11>>  DRILL_HOLE
  ( FEATURE_ID    => 7,
    TOOL_TYPE_ID  => DRILL_0.1719_2_ABS,
    PREC_STEPS    => (10),
    TIME          => "0000:01:00:00" );

<<12>>  MACHINE_CHAMFER_IN
  ( FEATURE_ID    => 8,
    TOOL_TYPE_ID  => CHAMFER_0.375_3_ABS,
    PREC_STEPS    => (11),
    TIME          => "0000:01:00:00" );

<<13>>  MACHINE_COUNTERSINK
  ( FEATURE_ID    => 7,
    TOOL_TYPE_ID  => COUNTERSINK_0.75_1_AB,
```

```
<<14>>  MACHINE_COUNTERSINK
  ( FEATURE_ID    => 6,
    TOOL_TYPE_ID  => COUNTERSINK_0.75_1_AB,
    PREC_STEPS    => (13),
    TIME          => "0000:01:00:00" );

<<15>>  TAP_THREAD
  ( FEATURE_ID    => 7,
    TOOL_TYPE_ID  => TAP_0.19_0_ABS,
    PREC_STEPS    => (14),
    TIME          => "0000:01:00:00" );

<<16>>  TAP_THREAD
  ( FEATURE_ID    => 6,
    TOOL_TYPE_ID  => TAP_0.19_0_ABS,
    PREC_STEPS    => (15),
    TIME          => "0000:01:00:00" );

<<17>>  MILL_CONTOUR_POCKET
  ( FEATURE_ID    => 12,
    TOOL_TYPE_ID  => END_MILL_0.25_2_AB,
    PREC_STEPS    => (16),
    TIME          => "0000:01:00:00" );

<<18>>  MILL_STRAIGHT_GROOVE
  ( FEATURE_ID    => 14,
    TOOL_TYPE_ID  => END_MILL_0.125_2_AB,
    PREC_STEPS    => (17),
    TIME          => "0000:01:00:00" );

<<19>>  MILL_STRAIGHT_GROOVE
  ( FEATURE_ID    => 13,
    TOOL_TYPE_ID  => END_MILL_0.125_2_AB,
    PREC_STEPS    => (18),
    TIME          => "0000:01:00:00" );

<<20>>  MILL_GROOVE
  ( FEATURE_ID    => 11,
    TOOL_TYPE_ID  => BALL_NOSED_END_MILL_0.25_4_BS,
    PREC_STEPS    => (19),
    TIME          => "0000:01:00:00" );

<<21>>  DRILL_HOLE
  ( FEATURE_ID    => 10,
    TOOL_TYPE_ID  => DRILL_0.5_2_ABS,
    PREC_STEPS    => (20),
    TIME          => "0000:01:00:00" );

<<22>>  MACHINE_CHAMFER_IN
  ( FEATURE_ID    => 11,
    TOOL_TYPE_ID  => CHAMFER_0.375_3_ABS,
    PREC_STEPS    => (21),
    TIME          => "0000:01:00:00" );

<<23>>  MACHINE_CHAMFER_OUT
  ( FEATURE_ID    => 11,
    TOOL_TYPE_ID  => CHAMFER_0.375_3_ABS,
    PREC_STEPS    => (22),
    TIME          => "0000:01:00:00" );

<<24>>  CLOSE_PLAN
  ( PREC_STEPS    => (23),
    TIME          => "0000:01:00:00" );
--END_PROCEDURE_SECTION--
--END_PROCESS_PLAN--
```

Table 4. Enhanced LISP-Readable Process Plan for XYZ Part

```
(setplist 'xyz_plan
 '(header (header
        plan_id   xyz_plan
        design_id xyz
        material  aluminum)
  steps  (steps
        1  (1   work_element    initialize_plan
              prog_name       "demo part")
        2  (2   work_element    set0_corner
              tool_type_id    probe_0.25
              changer_slot    40
              corner          1
              x_offset        0.0
              y_offset        0.0
              near_x          17.3
              near_y          7.45
              precedent_steps (1))
        3  (3   work_element    set0_z
              tool_type_id    probe_0.25
              changer_slot    40
              x_loc           20.3
              y_loc           8.925
              offset          -0.05
              precedent_steps (2))
        4  (4   work_element    face_mill
              tool_type_id    face_mill_2.0_4_abs
              changer_slot    15
              upper_l_x       0.0
              upper_l_y       2.95
              lower_r_x       6
              lower_r_y       0.0
              depth           0.05
              z_surf          0.05
              stepover        1.0
              speed           2291
              feed_rate       18
              pass_depth      0.1
              precedent_steps (3))
        5  (5   work_element    mill_side_contour
              feature_id      1
              tool_type_id    end_mill_1.0_2_ab
              changer_slot    21
              stepover        0.5
              speed           1718
              feed_rate       17
              pass_depth      0.5
              precedent_steps (4 3 2))
        6  (6   work_element    mill_text
              feature_id      4
              tool_type_id    ball_nosed_end_mill_0.25_4_bs
              changer_slot    5
              speed           3437
              feed_rate       8
              pass_depth      0.125
              precedent_steps (5))
        7  (7   work_element    mill_text
              feature_id      3
              tool_type_id    ball_nosed_end_mill_0.25_4_bs
              changer_slot    5
              speed           3437
              feed_rate       8
              pass_depth      0.125
              precedent_steps (6))
        8  (8   work_element    mill_text
              feature_id      2
              tool_type_id    ball_nosed_end_mill_0.25_4_bs
              changer_slot    5
              speed           3437
              feed_rate       8
              pass_depth      0.125
              precedent_steps (7))
        9  (9   work_element    mill_pocket
              feature_id      8
              tool_type_id    end_mill_0.5625_2_ab
              changer_slot    18
              stepover        0.28125
              speed           3055
              feed_rate       17
              pass_depth      0.28125
              precedent_steps (8))
        10 (10  work_element    mill_side_contour
              feature_id      5
              tool_type_id    end_mill_0.5_2_ab
              changer_slot    2
              stepover        0.25
              speed           3437
              feed_rate       17
              pass_depth      0.25
              precedent_steps (9))
        11 (11  work_element    mill_contour_groove
              feature_id      9
              tool_type_id    end_mill_0.125_2_ab
              changer_slot    11
              speed           5200
              feed_rate       3
              pass_depth      0.0625
              precedent_steps (10))
        12 (12  work_element    drill_hole
              feature_id      6
              tool_type_id    drill_0.1719_2_abs
              changer_slot    25
              speed           5200
              feed_rate       4
              pass_depth      0.1719
              precedent_steps (11))
```

```
    13 (13 work_element    drill_hole
          feature_id      7
          tool_type_id    drill_0.1719_2_abs
          changer_slot    25
          speed           5200
          feed_rate       4
          pass_depth      0.1719
          precedent_steps (12))
    14 (14 work_element    machine_chamfer_in
          feature_id      8
          tool_type_id    chamfer_0.375_3_abs
          changer_slot    6
          speed           5200
          feed_rate       29
          precedent_steps (13))
    15 (15 work_element    machine_countersink
          feature_id      7
          tool_type_id    countersink_0.75_1_ab
          changer_slot    4
          speed           2291
          feed_rate       5
          precedent_steps (14))
    16 (16 work_element    machine_countersink
          feature_id      6
          tool_type_id    countersink_0.75_1_ab
          changer_slot    4
          speed           2291
          feed_rate       5
          precedent_steps (15))
    17 (17 work_element    tap_thread
          feature_id      7
          tool_type_id    tap_0.19_0_abs
          changer_slot    26
          speed           376
          feed_rate       300
          precedent_steps (16))
    18 (18 work_element    tap_thread
          feature_id      6
          tool_type_id    tap_0.19_0_abs
          changer_slot    26
          speed           376
          feed_rate       300
          precedent_steps (17))
    19 (19 work_element    mill_contour_pocket
          feature_id      12
          tool_type_id    end_mill_0.25_2_ab
          changer_slot    19
          stepover        0.125
          speed           5200
          feed_rate       13
          pass_depth      0.125
          precedent_steps (18))
    20 (20 work_element    mill_straight_groove
          feature_id      14
          tool_type_id    end_mill_0.125_2_ab
          changer_slot    11
          speed           5200
          feed_rate       3
          pass_depth      0.0625
          precedent_steps (19))
    21 (21 work_element    mill_straight_groove
          feature_id      13
          tool_type_id    end_mill_0.125_2_ab
          changer_slot    11
          speed           5200
          feed_rate       3
          pass_depth      0.0625
          precedent_steps (20))
    22 (22 work_element    mill_groove
          feature_id      11
          tool_type_id    ball_nosed_end_mill_0.25_4_bs
          changer_slot    5
          speed           3437
          feed_rate       8
          pass_depth      0.125
          precedent_steps (21))
    23 (23 work_element    drill_hole
          feature_id      10
          tool_type_id    drill_0.5_2_abs
          changer_slot    34
          speed           2100
          feed_rate       5
          pass_depth      0.5
          precedent_steps (22))
    24 (24 work_element    machine_chamfer_in
          feature_id      11
          tool_type_id    chamfer_0.375_3_abs
          changer_slot    6
          speed           5200
          feed_rate       29
          precedent_steps (23))
    25 (25 work_element    machine_chamfer_out
          feature_id      11
          tool_type_id    chamfer_0.375_3_abs
          changer_slot    6
          speed           5200
          feed_rate       29
          precedent_steps (24))
    26 (26 work_element    close_plan
          precedent_steps (25)))
  tool_requirements (
    face_mill_2.0_4_abs
    probe_0.25
    end_mill_1.0_2_ab
    ball_nosed_end_mill_0.25_4_bs
    end_mill_0.5625_2_ab
    end_mill_0.5_2_ab
    end_mill_0.125_2_ab
    drill_0.1719_2_abs
    chamfer_0.375_3_abs
    countersink_0.75_1_ab
    tap_0.19_0_abs
    end_mill_0.25_2_ab
    drill_0.5_2_abs)))
```

2. STRUCTURE OF PROCESS PLAN FILES

2.1. Overview

A process plan file is a computer-readable file or a printout of a computer-readable file. The computer-readable version is made up of ascii characters without any non-printing characters (except those in LISP format that LISP can ignore) except white space characters.

2.2. VWS-Standard Format

The process plan is a file with four sections. The sections must be in a fixed order. Each section (and the plan itself) begins and ends with a key line, so that there are ten key lines. In some cases the VWS2 system cannot handle a key line with extraneous white space at the beginning or end, even though the AMRF standard allows such white space. Blank lines, however, are completely irrelevant. The gross structure is thus as follows (the material in square brackets is commentary).

```
--PROCESS_PLAN--
--HEADER_SECTION--
   [header information goes here]
--END_HEADER_SECTION--
--PARAMETERS_SECTION--
   [parameter information goes here]
--END_PARAMETERS_SECTION--
--REQUIREMENTS_SECTION--
   [requirements information goes here]
--END_REQUIREMENTS_SECTION--
--PROCEDURE_SECTION--
   [procedure information goes here]
--END_PROCEDURE_SECTION--
--END_PROCESS_PLAN--
```

2.3. LISP-Readable Format

The process plan file is a LISP-loadable file that, when loaded, sets up the property list of the plan_id listed in the header. The property list must have at least three properties: header, steps, and tool_requirements. The properties are normally in that order, but the order is not significant. There are also normally no other properties included, but if there are, that does not matter. Thus, the gross structure is as follows (the material in square brackets is commentary but the parentheses are significant).

```
(setplist '[plan_id goes here] '(
   header [header list goes here]
   steps  [step list goes here]
   tool_requirements [tool requirements list goes here]))
```

3. HEADER SECTION

3.1. VWS-Standard Format

A series of lines goes in between the key lines that start and end the header section. The order of the lines is irrelevant. A typical line might look like the following.

```
PLAN_ID         := XYZ_PLAN;
```

A typical line starts with some optional white space. Then there is property name (PLAN_ID in the case of the line shown). Then any amount of white space. Then the symbol := Then more white space. Then the property value (XYZ_PLAN on the line shown) followed by a semicolon. If the property value has any white space in it, it must be surrounded by double quotes. White space before or after the semicolon will be ignored.

For a process plan in VWS-standard format to be readable by the VWS2 system, the following properties are required in the header: PLAN_ID, DESIGN_ID, and MATERIAL. The values of PLAN_ID and DESIGN_ID should be LISP atoms. The allowed values of MATERIAL are ALUMINUM, BRASS, STEEL, and MONEL.

Other properties that are put into the header by the VWS2 print routine for VWS-standard format plans, but are not required by the VWS2 system when a VWS-standard format plan is read in are: PLAN_VERSION (if there is no value in the LISP environment the default value of 1 is used), PLAN_TYPE (which is always INSTRUCTION_SET), and PROCESS_ENGINEER (if there is no value in the LISP environment the default value of "VWS2 AUTOMATIC PROCESS PLANNER" is used). PLAN_VERSION and PLAN_TYPE are used by the VWS2 system in connection with AMRF database reports.

3.2. LISP-Readable Format

The header is a disembodied property list. The first entry in the list should be "header". The properties plan_id, design_id, and material are required. The order of the properties is irrelevant, and, since the header is a list, any amount of white space and any number of new line characters may be thrown in. Other property-value pairs may be included in any order, but they are not expected and will be ignored if they are present. A typical header might be as follows:

```
(header plan_id xyz_plan design_id xyz material aluminum)
```

4. PARAMETERS SECTION

4.1. Overview

Only the VWS-standard format version has a parameters section. The LISP-readable version does not normally have one. No use is made of it by the VWS2 system if there is one. The print routine for VWS-standard format files creates a meaningful parameters section from the tool requirements when it is printing. The reading routine skips over the section. The parameters section is

potentially useful to a workstation or machine tool control system to aid in the selection of a specific workpiece and specific tools, but in the VWS2 system, the Data Execution module selects tools, and the user selects the workpiece.

4.2. VWS-Standard Format

Between the key lines that start and end the parameters section there may be any number of lines. Each line identifies one parameter. A typical line is as follows:

```
$$TOOL3         : TOOL;
```

First there is optional white space. Then a parameter name ($$TOOL3 on the line shown). The two dollar signs at the beginning of the name are optional indicators that the name stands for a parameter. Then white space. Then a colon. Then more white space. Then the name of the type of thing the parameter is (TOOL on the line shown) followed by a semicolon. The name for a type of parameter may be WORKPIECE, TOOL_SET, or TOOL. White space before or after the semicolon will be ignored.

5. REQUIREMENTS SECTION

5.1. General

The requirements section is called "tool_requirements" in the LISP-readable version. The formats are rather different. As far as the VWS2 system is concerned, the important information in this section is the tool type name of each tool.

The tool type names are expected to match the tool type names of tools in the database representing the tools actually present in the tool changer of the milling machine. That database contains other information about each tool. The database is discussed elsewhere. The type name is not a unique identifier of a tool in the machine. There may be several tools in the machine with the same type name. In general, the tool type name is the name of a tool type followed by the diameter of the tool, the number of flutes, and some letters representing the materials the tool can cut (where a = aluminum, b = brass, s = steel, and m = monel. The name is joined by underscore characters. For example, the type name of a quarter inch diameter end mill with two flutes which is suitable for cutting aluminum and brass would be end_mill_0.25_2_ab. The system does not require that tool names be constructed this way. The only requirement is that each tool type have a unique name.

5.2. VWS-Standard Format

Between the key lines that start and end the requirements section there is one entry for each of the parameters in the parameters section. The AMRF standard does not prescribe the attribute-value pairs to be used, but an implementation by the Process Planning Project used the following pairs, which have been incorporated into the print routine for the VWS-standard format.

An entry for a workpiece is two lines long. An entry for a tool_set is three lines long, and an entry for a tool is four lines long. The entries are numbered with consecutive positive integers, starting at one. A typical entry for a tool is a follows:

```
<<3>>  TOOL
   ( TOOL_TYPE_ID   => PROBE_0.25,
     TOOL_ID        => $$TOOL1,
     COMPONENT_OF   => 2 );
```

The first line of the entry always starts with a positive integer enclosed in double chevrons (<<3>> in the entry above). Then there is white space. Then the name of the type of parameter (TOOL in the entry above).

The rest of the entry is enclosed in parentheses. If the rest of the entry has more than one line, each line except the last ends with a comma. The entry as a whole has a semicolon following the close parenthesis at the end of the entry. Each line of the rest of the entry includes a parameter-value pair separated by the symbol => . White space before or after the semicolon will be ignored.

The VWS2 read routine ignores white space and left parentheses at the beginning of each line of this section and ignores everything else on the line unless the next item encountered is TOOL_TYPE_ID, in which case a check is made for the => symbol and then the next continuous group of characters before a white space, comma, right parenthesis, or semicolon is taken to be the value of TOOL_TYPE_ID.

The print routine always gathers all the tools together into a tool_set, for example:

```
<<2>>  TOOL_SET
   ( TOOL_SET_ID    => $$TOOL_SET,
     COMPONENTS     => (3, 4, 5, 6, 7, 8, 9, 10, 11));
```

The value of components is a set of integers in parentheses, separated by commas. The integers are the identifying numbers of all the tools in the requirements section.

5.3. LISP-Readable Format

The value of tool_requirements is a simple list of tool type names. For example:

```
(probe_0.25   end_mill_0.375_2_ab   end_mill_0.125_2_ab   drill_0.1406_2_abs)
```

6. PROCEDURE SECTION

6.1. Overview

The procedure section is the heart of the process plan, and the two formats for this section match very closely (although it is simply called "steps" in the LISP-readable format). The steps are numbered sequentially starting with 1. Each step has a work element name. The remainder of each step is a set of parameter-value pairs.

The VWS2 system can deal with 21 different types of work element, and a process plan may have up to 1023 steps in the procedure section. Specifications for work elements are given later in this paper.

The procedure section always begins with an initialize_plan step and always ends with a close_plan step. Neither of these two types of step ever appears in the middle of the procedure section. The steps in the middle of the section are either one of three zero-setting operations, or one of sixteen metal cutting operations.

6.2. Parameters

6.2.1. Overview

As mentioned earlier, a process plan is enhanced by the Data Execution module, which may add information to a given step, add steps, or delete steps. For each type of work element some parameter-value pairs are required in the unenhanced version, some are optional in the unenhanced version, and some are required in the enhanced version.

If a parameter is giving a size or location (which will usually be evident from the name of the parameter), the value should be a number representing a quantity expressed in inches. Such a number may be fixed or floating point, and initial or terminal decimal points are acceptable.

Some parameters-value pairs are used with many different work elements, as follows.

6.2.2. Work Element

In the VWS-standard format, the name of the work element is given on the first line of the step without any specific indication that it is a work element. In the LISP-readable format, work_element is the parameter name for one of the parameter-value pairs and must be present in every step of the procedure section.

6.2.3. Time

A time entry appears in every step of the procedure section in the VWS-standard format. The print routine for the VWS-standard format gives every time entry the value in the plan, if there is one. Otherwise, it prints the value "0000:01:00:00" (including the quotes), which is one hour. Time does not appear in the LISP-readable format and is not used in any way by the VWS2 system. The

VWS2 read routine for the VWS-standard format ignores time entries.

6.2.4. Precedent Steps

A list of precedent steps is required in every step except initialize_plan, which must have no precedent steps. The assignment of precedent steps must be such that the close_plan step cannot be executed until all other steps have been executed. The parameter name is PREC_STEPS in the VWS-standard format and precedent_steps in the LISP-readable format.

The value of this parameter is a set of step numbers enclosed in parentheses. In the VWS-standard format these step numbers must be separated by commas. The reading routine for the VWS-standard format will accept the list with or without commas. The print routine for the VWS-standard format always prints the commas. In the LISP-readable format commas may not be present.

6.2.5. Tool Type Identifier

This serves to identify the type of tool needed to carry out the step and is required in every step except initialize_plan and close_plan. The parameter name is TOOL_TYPE_ID in the VWS-standard format and tool_type_id in the LISP-readable format. Acceptable values have already been discussed.

6.2.6. Feature Identifier

For those steps which cut features, this serves to identify the feature from the design to which the operation is to be applied. This is all work elements except initialize, close, the three zero-setters, fly_cut and face_mill which are performed without regard to features. The parameter name is FEATURE_ID in the VWS-standard format and feature_id in the LISP-readable format. The value must be a positive integer which is the number of a feature from the design.

6.2.7. Speed

This is the turning rate of the spindle of the milling machine during the operation, in revolutions per minute. It is not used for initialize or close steps or the three zero-setters. It is optional in the unenhanced version of all other steps and required in the enhanced version. The parameter name is SPEED in the VWS-standard format and speed in the LISP-readable format.

6.2.8. Feed Rate

This is the rate in inches per minute at which the tool moves relative to the workpiece. It is not used for initialize or close steps or the three zero-setters. It is optional in the unenhanced version of all other steps and required in the enhanced version. The parameter name is FEED_RATE in the VWS-standard format and feed_rate in the LISP-readable format.

6.2.9. Vertical Pass Depth

If a feature is deep, it must be cut in several passes. The vertical pass depth is the incremental depth of each pass for a deep cut. It is not used for initialize or close steps, the three zero-setters, the two chamfers, tapping, center_drilling, counterboring, or countersinking. It is optional in the unenhanced version of all other steps and required in the enhanced version. The parameter name is PASS_DEPTH in the VWS-standard format and pass_depth in the LISP-readable format.

6.2.10. Stepover

If a lot of material is to be milled out of a feature (a pocket, for example) each vertical pass usually includes several back-and-forth passes of an end mill. The horizontal cut depth of one of these back-and-forth passes is called stepover. Stepover is optional in the unenhanced version of mill_pocket, mill_contour_pocket, mill_side_contour, fly_cut, and face_mill, required in the enhanced version of those five operations, and not needed in any other operation. The parameter name is STEPOVER in the VWS-standard format and stepover in the LISP-readable format.

6.2.11. Changer slot

This is the number of the changer slot on the milling machine in which the tool to be used for the step is to be found. The number is a positive integer between 1 and 40. It is not used for initialize or close steps. It should not be present in the unenhanced version of all other steps, but it is required in the enhanced version. The parameter name is CHANGER_SLOT in the VWS-standard format and changer_slot in the LISP-readable format.

6.3. VWS-Standard Format

Between the key lines that start and end the procedure section there are numbered entries, each several lines long, for example:

```
<<5>>  MILL_POCKET
   ( FEATURE_ID     => 7,
     TOOL_TYPE_ID   => END_MILL_0.375_2_AB,
     PREC_STEPS     => (4, 11, 3),
     TIME           => "0000:01:00:00" );
```

The first line starts with the number of the step in double chevrons. Then white space. Then the name of the work element. The remainder of the entry is enclosed in parentheses and ended by a semicolon after the right parenthesis. Each line of the remainder of the entry is a parameter-value pair separated by the symbol => . This symbol should have white space on both sides. After each line of the remainder, except the last one, there is a comma. The order of the lines is irrelevant. White space before or after the semicolon will be ignored.

6.4. LISP-Readable Format

The list of steps is a disembodied property list, the first element of which is "steps". The property names are the positive integers starting with 1 and going up in order. Each value in the list is itself a disembodied property list, the first element of which is the same integer that is the property name, for example:

```
(5  work_element    mill_pocket
    feature_id      7
    tool_type_id    end_mill_0.375_2_ab
    precedent_steps (4 11 3))
```

The fact that this is printed on several different lines is of no relevance. The order of the properties is also irrelevant.

7. WORK ELEMENTS

7.1. Overview

There are 21 work elements in the VWS2 system. The names of these work elements and a brief description of each follow. The first twelve of the work elements require only those parameters discussed above. All of these twelve require a feature_id among the parameters. Any parameters required by items 13 - 21 below that are not discussed above are presented below. A summary of the work elements is shown in Table 5.

Each work element description includes a discussion of the effect of carrying out the work element. The meaning of "carrying out" a work element may vary according to what is being done. The following material is phrased as though the plan was being executed in real time and the workpiece was changing as each step of the plan was executed. The VWS is not currently executing milling machine process plans in real time. Rather, process plans are fed to the Data Execution module, which emulates execution of the plan on an internal model of a workpiece. Usually NC-code is written at the same time, and it is execution of the NC-code on the milling machine at a later time that actually changes workpieces.

It will be helpful in reading this section to look at pictures of the various feature types shown in section 2 of Chapter II of the design paper [K&J2].

7.2. Drill_hole

The feature must be a hole, which must have a conical bottom if it is not a thru-hole. The drill_hole operation drills to the bottom of the hole. If the hole has subfeatures, such as a thread, this operation does NOT make the subfeatures. The tool used must be a drill.

7.3. Machine_chamfer_in

The feature must be a hole, pocket, groove, or straight_groove. The feature must have a chamfer_in_depth. This operation causes the chamfer specified in the design to be made. The feature must exist before this operation is attempted. The tool used must be a chamfer tool.

7.4. Machine_chamfer_out

The feature must be a chamfer_out or a groove. The feature must have a chamfer_out_depth. This operation causes the chamfer specified in the design to be made. On a groove a chamfer_out appears on the edge of the island left inside the groove. If the feature is a groove, it must exist before this operation is attempted. The tool used must be a chamfer tool.

7.5. Machine_countersink

The feature must be a hole with a countersink diameter. This operation causes the countersink specified in the design to be made. The hole must exist before this operation is attempted. The tool used must be a countersink.

7.6. Mill_contour_groove

The feature must be a contour_groove. This operation causes the contour_groove to be made. The tool used must be an end mill if the bottom of the contour_groove is flat or a ball nosed end mill if the bottom of the contour_groove is round.

7.7. Mill_contour_pocket

The feature must be a contour_pocket. This operation causes the contour_pocket to be made. The tool used must be an end mill.

7.8. Mill_groove

The feature must be a groove. This operation causes the groove to be made. The tool used must be an end mill if the bottom of the groove is flat or a ball nosed end mill if the bottom of the groove is round.

7.9. Mill_pocket

The feature must be a pocket. This operation causes the pocket to be made. The tool used must be an end mill.

7.10. Mill_side_contour

The feature must be a side_contour. This operation causes the side_contour to be made. The tool used must be an end mill.

7.11. Mill_straight_groove

The feature must be a straight_groove. This operation causes the straight_groove to be made. The tool used must be an end mill if the bottom of the straight_groove is flat or a ball nosed end mill if the bottom of the straight_groove is round.

7.12. Mill_text

The feature must be text. This operation causes the text to be made. The tool used must be a ball nosed end mill.

7.13. Tap_thread

The feature must be a hole with the three parameters needed to specify a thread. This operation causes the thread specified in the design to be made. The hole must exist before this operation is attempted. The tool used must be a tap.

7.14. Initialize_plan

This has only one parameter besides those discussed above. That is PROG_NAME in VWS-standard format and prog_name in LISP-readable format. This is an abbreviation of program name. The value should be a string of not more than 30 characters. If NC-code is written by the Data Execution module, this program name will appear in the first line of the NC-code.

7.15. Close_plan

This has no parameters besides those above.

7.16. Center_drill

This requires a feature_id and one extra parameter: center_drill_depth. The feature must be a hole. The hole should not have been made before this operation is used. This operation causes a conical hole to be made centered on the center of the feature. The depth of the conical hole is given by the center_drill_depth. The tool must be a center drill.

7.17. Counterbore

This requires a feature_id and one extra parameter: counterbore_depth. The feature must be a hole. The hole should have been made before this operation is used. The counterbore_depth may not be greater than the hole depth. This operation only improves an existing hole.

7.18. Face_mill

This does NOT require a feature_id because it does not make a feature. It is used only for cutting a workpiece that is too tall down to size. Conceptually, this operation face mills away a rectangular area from a certain starting height to a certain depth. There must be no material on the workpiece outside this rectangular area above the lowest extent of the face mill or the tool may crash into it. Usually the rectangular area is the entire top of the workpiece, but this is not required. The tool must be a face mill.

Six extra parameters are required, four of which define the rectangle:
i. upper_l_x -- x-coordinate of upper left hand corner of the rectangle.
ii. upper_l_y -- y-coordinate of upper left hand corner of the rectangle.
iii. lower_r_x -- x-coordinate of lower right hand corner of the rectangle.
iv. lower_r_y -- y-coordinate of lower right hand corner of the rectangle.
v. depth -- how far below the starting z-value to mill away.
vi. z_surf -- The distance above the final location of the top of the part where the face milling should start.

7.19. Fly_cut

The description of this operation and its parameters is identical to that of face_mill. The only difference between the two operations is that fly_cutting is done with a fly cutter. The fly cutter normally is larger than the face mill so it makes a wider swath, but the pass depth of fly cutter is less.

7.20. Set0_center

This operation and the next two do not cut any metal. It finds the exact x and y locations of the center of a hole on a workpiece whose approximate location and diameter are known and uses that information to establish the zero points for the x and y axes of the milling machine. This operation and the next two operations rely on the fact that two different coordinate systems may be used: i. an absolute coordinate system on the milling machine, and (ii) a system whose axes are parallel to those of the first, but whose origin is moved. In the VWS2 system, the second coordinate system has its origin at the top, front, left corner of the workpiece.

This operation and the next one are used because, when a workpiece is fixtured in the milling machine, its exact location is not usually known. In order to do correct milling, the workpiece must be correctly located. The tool must be a probe tool, and, obviously, the milling machine must have probing capability.

All three zero setting operations allow the origin of the relative coordinate system to be set at some point that is offset from the location being probed. Thus, if a part is known to have a hole whose center is two inches from the front left of the part in the x-direction, and three inches in the y-direction, it is possible to set the origin at the front left corner by finding the center of the hole.

Five extra parameters are required:

i. near_diam -- the approximate diameter of the hole.
ii. near_x -- the approximate x-value in absolute machine coordinates of the center of the hole.
iii. near_y -- the approximate y-value in absolute machine coordinates of the center of the hole.
iv. x_offset -- the amount to add to the absolute x-value of the center of the hole before setting the relative x-zero at that number.
v. y_offset -- the amount to add to the absolute y-value of the center of the hole before setting the relative y-zero at that number.

7.21. Set0_corner

This operation finds the exact x and y locations of a corner of a part. The corner must have its sides parallel to the x and y axes. The comments about zero-setting given for set0_center apply to this operation as well.

Five extra parameters are required:

i. corner -- This is 1, 2, 3, or 4. It stands for the type of corner which is being probed. There are four types, corresponding to the four corners of a rectangle. The lower left type is 1, lower right 2, upper right 3, and upper left 4.
ii. near_x -- the approximate x-value in absolute machine coordinates of the corner.
iii. near_y -- the approximate y-value in absolute machine coordinates of the corner.
iv. x_offset -- the amount to add to the absolute x-value of the corner before setting the relative x-zero at that number.
v. y_offset -- the amount to add to the absolute y-value of the corner before setting the relative y-zero at that number.

7.22. Set0_z

This operation finds the z-location of the top of a part at a given xy location. The comments about zero-setting given for set0_center apply to this operation as well.

Three extra parameters are required:

i. x_loc -- The absolute x-value at which to probe.
ii. y_loc -- The absolute y-value at which to probe.
iii. offset -- the amount to add to the absolute z-value of the top of the part before setting the relative z-zero at that number.

Table 5. Summary of Work Elements

Work Element Name	Comments	Unenhanced Plan Parameters		Enhanced Plan Parameters Required
		required	optional	
center drill	Makes a small conical starter hole.	center_drill_depth, fptw	drs	center_drill_depth, cdfprstw
close_plan	Must be last step. Ends machning.	pw		pw
counterbore	Bores existing hole to given depth.	counterbore_depth, fptw	drs	counterbore_depth, cdfprstw
drill_hole	Drills a hole with a twist drill.	fptw	drs	cdfprstw
face_mill	Reduces height of workpiece.	See footnote 2, ptw	dors	See footnote 2, cdoprstw
fly_cut	Reduces height of workpiece.	See footnote 2, ptw	dors	See footnote 2, cdoprstw
initialize_plan	Must be first step. Starts machining.	prog_name, w		prog_name, w
machine_chamfer_in	Chamfers hole, pocket, groove, etc.	fptw	drs	cdfprstw
machine_chamfer_out	Chamfers block or island in a groove.	fptw	drs	cdfprstw
machine_countersink	Countersinks an existing hole.	fptw	drs	cdfprstw
mill_contour_groove	Mills a contour_groove.	fptw	drs	cdfprstw
mill_contour_pocket	Mills a contour_pocket.	fptw	dors	cdfoprstw
mill_groove	Mills a groove.	fptw	drs	cdfprstw
mill_pocket	Mills a pocket.	fptw	dors	cdfoprstw
mill_side_contour	Mills a side_contour.	fptw	dors	cdfoprstw
mill_straight_groove	Mills a straight_groove.	fptw	drs	cdfprstw
mill_text	Mills a text string.	fptw	drs	cdfprstw
set0_center	Sets x and y zero by probing a hole.	near_x, near_y, near_diam, x_offset, y_offset, ptw		near_x, near_y, near_diam, x_offset, y_offset, cptw
set0_corner	Sets x and y zero by probing a corner.	near_x, near_y, corner, x_offset, ptw		near_x, near_y, corner, x_offset, y_offset, cptw
set0_z	sets z zero by probing top of part.	x_loc, y_loc, offset, ptw		x_loc, y_loc, offset, cptw
tap_thread	Threads an existing hole.	fptw	drs	cdfprstw

1. c=changer_slot, d=pass_depth, f=feature, o=stepover, p=precedent_steps, r=feed_rate, s=speed, t=tool_type_id, w=work_element

2. Face_mill and fly_cut require: upper_l_x, upper_l_y, lower_r_x, lower_r_y, depth, and z_surf.

IV. PROCESS PLAN GENERATION FOR THE VWS MILLING MACHINE

1. INTRODUCTION

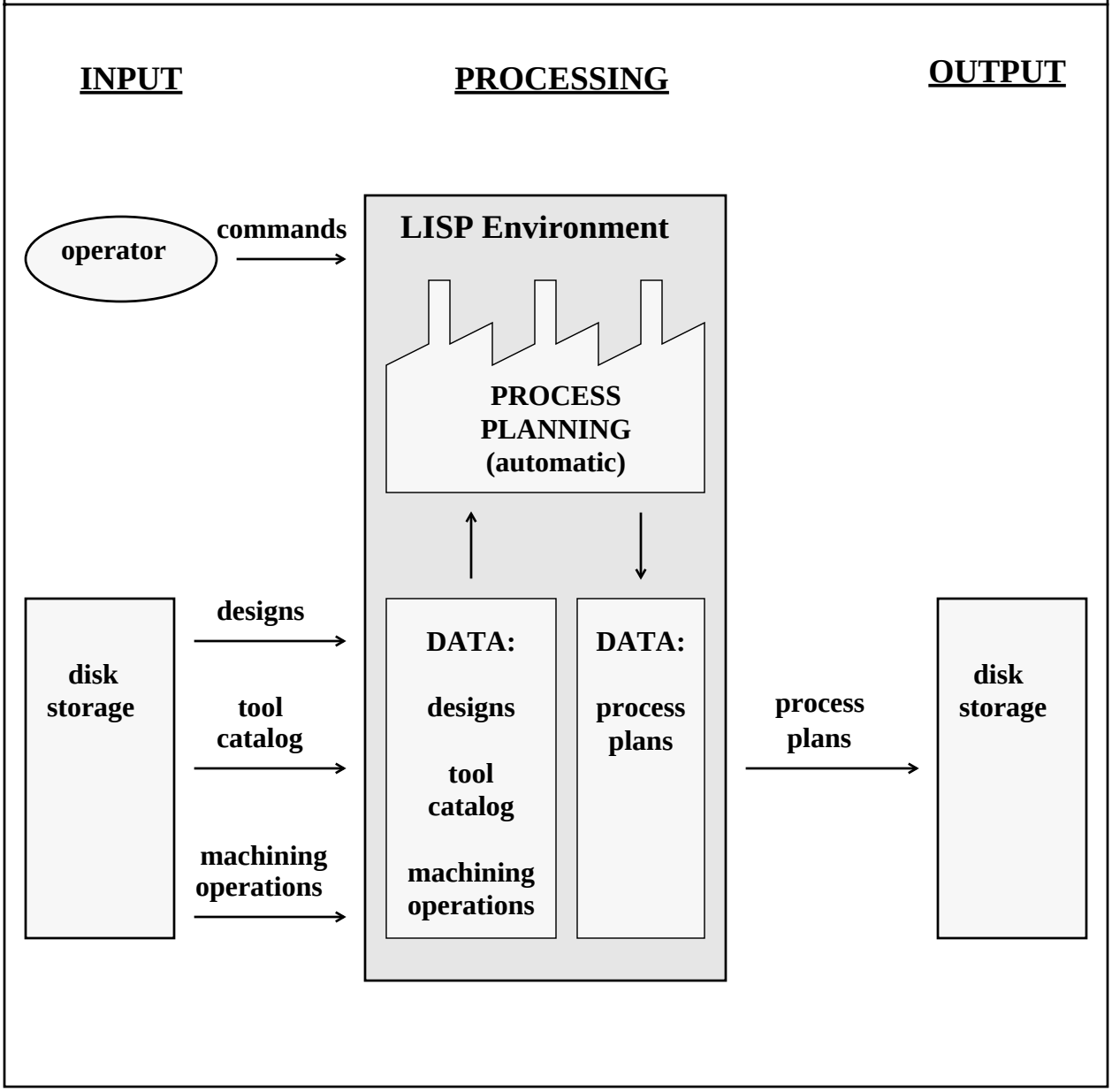
The VWS2 Automatic Process Planning module is a set of functions and data embedded in the VWS2 LISP environment. It takes the design of a part (in VWS design protocol format) as input and produces a process plan for milling a part of the given design. The process plan is stored in the LISP environment. Optionally, the module will print out a copy of the plan in either or both of the formats described in the previous chapter.

The VWS2 Automatic Process Planning module is not the final answer to the need for automatic process planning for a milling machine. It is limited in a number of ways -- most severely by the design protocol. The limitations are enumerated in section 4.

Although the Process Planning module may be called directly from the LISP environment, it is most easily used from the "vws_cadm" friendly user interface to the VWS2 system. Vws_cadm asks the user a series of questions (such as the name of the design and whether to print the plan) and then calls the Process Planning module to do the requested work. As mentioned earlier, the Process Planning module may also be called by the VWS control system.

Figure 2 shows the configuration of the Process Planning module.

Figure 2. Process Planning Module Configuration



2. DATA REQUIREMENTS

2.1. Introduction

Two databases are required by the Process Planning module: a tool catalog and a features database. These are presented briefly here and will be described further in other papers.

2.2. Tool Catalog

The tool catalog is a disembodied property list in the LISP environment. It contains descriptions of all tools which the Process Planning module may select for inclusion in process plans. Each tool has a unique name, called the "tool_id".

The catalog has two main sections: names and materials. In the names section the properties are tool_id's and the values are lists which are tool descriptions. The materials section is subdivided by tool type and each tool type is subdivided further by tool diameter. The "tap" tool type is further subdivided by threads per inch. The tool catalog currently in use contains 48 tools. The current catalog is small, but because of its structure and the way it is accessed, it is anticipated that several thousand tools could be included in the catalog without bogging the process planner down.

The need for information about tools in the VWS2 system has highlighted several research issues that require further attention. In the VWS2 system tool information must be commonly understood by the Process Planning and Data Execution modules. In a broader automated manufacturing environment, tool information must be commonly understood by materials handling and the global database, as well. Some issues are:
1. How should tools be identified? Should the name of the tool be derived from its physical characteristics?
2. What information about a tool is required in the database?
3. Should there be a fixed set of tools available in a catalog, or should it be possible to generate tool descriptions dynamically?
4. What additional information about a tool setup is needed when a tool is fixed in a tool holder and placed in the tool changer of a machine?

2.3. Features Database

The VWS2 features database is a list of feature types with several attribute-value pairs for each feature type. One of the attributes each feature has is called "oper_finder". The value of "oper_finder" is the name of a function that determines which machining operations will be needed to make a feature of that type.

3. GENERATING A PROCESS PLAN

3.1. Overview

The software which generates process plans is based on unsophisticated rules for selecting and ordering machining operations. The rules, described in this section, are built into the software and not kept in a separate database. The list manipulation done by the software is modestly sophisticated, but that is a technical side issue; clunky list manipulation would suffice and not be much slower.

The reason that a plan can be created by relatively simple routines is that the VWS2 design protocol was specifically tailored to facilitate process planning. The key characteristics of the design protocol that facilitate process planning are:
1. It is a constructive solid geometry type system in which subtraction of primitive volumes is the only operation allowed.
2. Each primitive volume (or feature) in the protocol may be produced by one or a few common machining operations.
3. The notion of a reference feature makes it easy to partially order machining operations. The reader is referred to [K&J2] for a detailed description of the protocol.

The VWS2 Process Planning Module is fast. The XYZ plan was generated in forty seconds on a Sun 3/160 computer running interpreted LISP. Preparing this plan from a previously unseen design would have taken the author several hours using a text editor, and the plan would have contained mistakes.

Process plans for the VWS milling machine are generated as property lists in the LISP environment and are in the same format as the printed LISP-readable version.

Process plans prepared automatically may be edited by hand. It is feasible to do this using a text editor such as EMACS, but tedious. It would be nice to have a friendly process plan editor in the VWS2 system analogous to the Part Design Editor, but none is currently available. Making changes to an automatically prepared plan may be desirable in the following situations:
1. A partially machined part is being machined a second time and it is not feasible to probe the front left corner of the part to establish x and y zero. In this case a corner or hole that can be probed is identified by the user, and the appropriate zero-finding routine is inserted in the plan.
2. A part is being machined on the pallet and is not centered on the pallet. Again the zero-finding routine is changed.
3. Intersecting features occur on the part and the order of machining should be changed. For example, if a pocket and a hole are to be made in the top of a part, and the hole is to be drilled right on the edge of the pocket, the VWS2 process planner (which does not look for feature intersections) will choose to make the pocket first. This is bad machining because the drill will probably break. The order of operations should be changed. This may be done simply by changing precedent step lists.

4. The user wants to reduce machining time. Speeds, feed rates, and pass depths used by the VWS2 system are generally conservative and may be safely increased in some situations.

3.2. Header

The plan header is easily assembled from information provided by the user.

3.3. List Of Steps

3.3.1. Getting Started

Almost all of the work of the Process Planning Module occurs in generating the list of steps in a plan.

The first, second, and last steps are always initialize_plan, set0_corner, and close_plan, respectively. The system assumes that the part is to be made in the vise of the milling machine, and sets the parameters of set0_corner appropriately for the vise. The parameters will be reset automatically in the Data Execution module for milling on a pallet if the user decides to mill on the pallet. If the part is not centered on the pallet, however, the parameters must be reset by hand.

3.3.2. The Heart of the List

The remaining steps are generated without step numbers, speeds, feed rates, pass depths, stepovers, or precedent steps. This is done as follows.

3.3.2.1. Divide Design in Levels

The design is divided into levels by reference feature. Level 1 includes all features that have no reference feature. Level 2 includes all features whose reference feature is in level 1. Level 3 includes all features whose reference feature is in level 2, and so on until all features are assigned to a level. There is no maximum number of levels, but of course the number of levels cannot be larger than the number of features. Note that the depth of the features is not used in determining levels. It is likely that a feature in level n may be deeper than some feature in level n+m.

In the XYZ part, the levels are as follows:
level 1 -- features 1, 2, 3, 4
level 2 -- features 5, 6, 7, 8, 9
level 3 -- features 10, 11, 12, 13, 14
Notice that feature 6 in level 2 (one of the tapped holes) extends below most of the features in level 3. It is coincidental that the levels for the XYZ part are in increasing feature order; that is irrelevant to the operation of the system.

All the features in level n (including their subfeatures) will be machined before any feature in level n+1. This is a sensible method of proceeding because every feature in level n+1 is underneath some feature in level n. Once the features in level n have been made, because the outline of a feature always fits inside the outline of a reference feature, it is certain that there will be clear access from above to every feature in level n+1.

3.3.2.2. Select Operations for Each Feature

Operations are selected for making each feature in two stages.
1. The operation selection function for that feature type named in the features database selects operations for making the main feature.
2. The feature is examined for optional parameters indicating that it has subfeatures. If these parameters are found, then the operation selection function for each subfeature type selects operations for making the subfeature.

The operation selectors are mostly rather simple. Each operation selector calls the tool selector to select a tool to perform the operation. The actions of the 12 operation selectors and the tool selector are summarized in Table 6. Operation selection is trivial except in the case of a hole feature (see footnote 1 of Table 6).

The tool selector first selects a type of tool according to the operation and feature. Next it selects a diameter for the tool. Finally, it looks in the tool catalog under the correct material and returns the name of the tool that has the right type and diameter. If there is no such tool in the catalog, an error message is returned. In the case of taps, the number of threads per inch is also checked.

Once an operation has been selected, selecting the type of tool is simple in all cases. Selecting the tool diameter is simple in all cases except for making contour_pockets and contour_grooves. The tool diameter is determined by the dimensions of the feature for: groove, contour_groove, straight_groove, text, hole (if made by a drilling operation), and thread. The largest tool in the catalog smaller than a certain size is selected for: face milling, fly cutting, and mill_pocket (for which the upper limit is determined by the corner radius of the pocket). There is only one size for a chamfer tool (0.375) and a countersink (0.75).

For contour pockets and side contours, the largest tool in the catalog that will fit into the smallest corner of the feature is tentatively selected. Then the tool path required to cut the outline of the feature is generated and a check is made of whether the tool will cut away material it should not cut when it follows this path. If the check is OK, the selection is made final. Otherwise, the next smaller tool in the catalog is tried. This goes on until a size that works is found.

3.3.2.3. Order the Operations in Each Level

Within each level, order the set of operations needed to make all the features (and their subfeatures) on that level. There are many ways to do this. The condition that a parent feature must be made before any of its subfeatures is observed. In some cases there is a preferred order for making subfeatures (countersink before tapping, for example).

The current ordering algorithm was selected to minimize tool changing. It uses tools in the following order: fly_cutter, face_mill, end_mill, ball_nosed_end_mill, drill, chamfer, countersink, tap. Within type, tools are used in decreasing diameter order.

An ordering algorithm based on operation type was implemented earlier but discarded in favor of the current algorithm.

In Table 2, the plan steps corresponding to the levels are:
level 1 -- steps 3, 4, 5, 6
level 2 -- steps 7, 8, 9, 10, 11, 12, 13, 14, 15, 16
level 3 -- steps 17, 18, 19, 20, 21, 22, 23

3.3.3. Finishing Up

If the user has asked to have speeds, feed rates, stepovers, and pass depths inserted (these four come as a package -- the option is called "extra items"), this is done for each step in the heart of the list. Only those items which are needed by a step are inserted. For example, drilling requires speed, feed rate, and pass depth, but not stepover. The calculation of these items is fairly simple and is adequate for aluminum, brass, steel, and monel. Details of these four items are given in chapter IV of [KR&W].

Step numbers and precedent steps are added to the heart of the list very simply. Step numbers are assigned to the heart of the list in order starting with 3 (1 and 2 are mentioned above). Since the plan is prepared so that it is known to be safe to carry it out in sequential order (barring any of the problems noted in section 4), the precedent steps are arranged so that the only possible order of execution that satisfies the precedence requirements is sequential order. This is done by having the list of precedent steps for step n+1 be the list (n) in all cases except step 1.

It would not be difficult to revise the method of assigning precedent steps so that only the "make reference feature first" and "make parent feature first" rules were embodied in the assignment of precedent steps. Then, to reach the current level of efficiency, the Data Execution module would have to be changed to do its ordering more intelligently in order to minimize tool changing. There seemed to be nothing to be gained by doing that, so it has not been done.

Table 6. Operation and Tool Selection

Feature or Subfeature	Operation	Tool
chamfer_in	machine_chamfer_in	chamfer
chamfer_out	machine_chamfer_out	chamfer
contour_groove	mill_contour_groove	end_mill /2 ball_nosed_end_mill
contour_pocket	mill_contour_pocket	end_mill
countersink	machine_countersink	countersink
groove	mill_groove	end_mill /2 ball_nosed_end_mill
hole	drill_hole /1 mill_pocket	drill end_mill
pocket	mill_pocket	end_mill
side_contour	mill_side_contour	end_mill
straight_groove	mill_straight_groove	end_mill /2 ball_nosed_end_mill
text	mill_text	ball_nosed_end_mill
thread	tap_thread	tap

1. In the case of a hole, if the hole is not a clean through hole, it is drilled if its bottom is conical and milled if the bottom is flat. A clean through hole is drilled if a drill of the right size is in the catalog. Otherwise, it is milled.

2. Grooves, contour_grooves, and straight_grooves are made with an end_mill if they are flat-bottomed and with a ball_nosed_end_mill if they are round-bottomed.

3.4. Tool Requirements

The tool requirements section of a process plan is generated by simply extracting the tool type identifiers from each step in the list of steps and removing duplicates.

4. LIMITATIONS

4.1. Design Protocol

Only parts whose design can be expressed in the VWS2 design protocol can have process plans generated for them

Since the design protocol assumes that all features are made in the same side of a part, a part requiring machining on more than one side requires a design for each side being machined. A separate milling machine process plan must be made for each design, and a workstation level process plan is needed to control sequencing of the milling machine plans and handling of the workpiece between cuts. It is feasible but tedious to make very complicated parts in this fashion.

4.2. Verification and Replanning

The automatic process planner will generate a process plan for any design for which tools of the right diameter (and the right number of threads per inch, in the case of taps) can be found in the tool catalog. The process planner assumes, however, that the tool will always be long enough to do the job. This is often not a correct assumption. The VWS2 process plan verification subsystem will catch errors of this sort when the Data Execution module is run. But there is no replanning capability in the VWS2 system which could try to find some other method of machining.

4.3. Feature Intersection

As described in section 1, the VWS2 system does not detect feature intersections and consider whether the order of machining should be altered to take account of feature intersections.

4.4. Multi-Operation Feature Making

Although the VWS2 system is comfortable with making each subfeature of a feature in a separate machining operation (a countersunk hole, for example requires a hole-making operation and a countersink operation), and can handle counterboring and center drilling adequately, it cannot deal with other types of making features in more than one machining operation. It would be nice to be able to make large features by first hogging then finishing. It would not be hard to add an operation like "hog_mill_pocket" to the list of machining operations, but modelling the execution of operations of this sort in the Data Execution module requires major changes to that module and has not yet been undertaken.

4.5. Tolerance Requirements

Tolerance requirements cannot currently be expressed in the VWS2 design protocol. It would be

nice to have tolerances expressed and to vary the choice of machining operations needed to make a feature according to the tolerance requirements.

4.6. Machining Forces

The system does not deal with cutting forces and how they might deform or break the part during machining.

4.7. Fixturing

The process planner does not prescribe fixturing. It assumes that the workpiece will be fixtured so that all steps of the plan can be carried out. The verification subsystem will detect interferences with fixturing, but, as noted earlier, no replanning capability is provided.

5. SOFTWARE

The VWS2 software for process planning and enhancement, outside of the database items and a few general functions for list manipulation, consists of 55 LISP functions, 45 from the "proc2" subdirectory and 10 from the "exec2" subdirectory of the "vws2" directory. For plan generation 33 of these are required, plan enhancement requires 17, and plan reading and writing requires 14. Nine of the 55 work in both generation and enhancement. The breakdown is shown in Table 7.

The process plan reading and writing (but not generation) capabilities of the VWS2 system extend beyond the milling machine. Workstation level plans are also being read with the same software. Plans for other VWS equipment could be handled by adding appropriate entries to the reading tables for acceptable parameters in the header and steps of a plan.

Table 7. Breakdown of Process Planning LISP Functions

PROC2 DIRECTORY	EXEC2 DIRECTORY
plan generation	**plan generation**
add_extra_items diameterize find_cg_ops find_chin_ops find_chout_ops find_cp_ops find_cs_ops find_csink_ops find_groove_ops find_hole_ops find_levels find_pocket_ops find_straight_groove_ops find_text_ops find_thread_ops generate_plan order_ops pass_depth plan_level plan_ops pull_tool_req select_cont_diam select_max_diam select_tool select_tool_diameter select_tool_type	feed_rate radial stepover spindle_speed spindle_speed_al spindle_speed_brass spindle_speed_steel spindle_speed_monel
plan enhancement	**plan enhancement**
add_extra_items cull_steps delete_step enhance_step insert_face_mill insert_step pass_depth	enhance1_plan enhance_tool_parms feed_rate init_exec_plan radial_stepover spindle_speed spindle_speed_al spindle_speed_brass spindle_speed_steel spindle_speed_monel
plan reading and writing	
headel letter_down letter_up print_plan print_plan_flat read_header read_item read_one_step read_parms read_plan_flat read_reqs read_steps skip_chars workel	

V. PROCESS PLAN ENHANCEMENT FOR THE VWS MILLING MACHINE

1. INTRODUCTION

As mentioned earlier, process plans are part of the input to the VWS2 Data Execution module, and the module enhances process plans. This paper does not describe the module in detail, but only its process plan enhancement functions. A detailed description of the module is given in [KR&W].

The enhancement software in the module is discussed immediately above at the end of chapter IV.

2. OPTIONS OF THE DATA EXECUTION MODULE

The Data Execution module has five independent processing and output options, any combination of which can be carried out simultaneously:
1. writing NC-code for the VWS milling machine,
2. drawing the part resulting from carrying out a process plan,
3. making a data model of that part,
4. verifying the process plan, and
5. printing an enhanced version of the input process plan.

The input options in the module are:
1. which process plan to execute,
2. which workpiece to use, and
3. whether to use an enhanced process plan as input. If the plan is already enhanced, no further enhancement is carried out before the plan is executed.

The machining options in the module are:
1. machine in the vise or on a pallet, and
2. establish z zero either by working down from the top of the part or up from the bottom.

3. DATA REQUIREMENTS

In addition to stored data about the milling machine, the Data Execution module requires a part design and a process plan as input. If a description of the workpiece selected by the user already exists in the LISP environment, it will be used by the module. If not, the module will assume that the workpiece is a featureless block of the size specified in the design and an appropriate description will be created.

The source of the input data is irrelevant as long as the data is in the right format. In particular, process plans may be received either from the VWS2 Process Planning module or from some other process planning system. The format of designs required by the Data Execution module is described in [K&J2].

The data about the milling machine that is required includes a list of the current tooling (with detailed information about each tool), parameters regarding fixturing geometry, and lists giving geometric information about fixturing obstacles. This data is semi-static. Input documents must

be edited by hand to change it.

4. ENHANCING THE PLAN

4.1. Introduction

The Data Execution module enhances an unenhanced plan in two stages. If the first stage is unsuccessful, the user is notified and the module quits. If the first stage is successful, the module proceeds to the second stage. If this is unsuccessful, the user is notified and the module quits. If the second stage is also successful, the module continues its work (which is not discussed further in this paper).

During enhancement, if a step is inserted or deleted, the module renumbers the steps of the plan and changes the precedent steps of each step to match the new numbering. This is a bit tricky.

If a step is inserted requiring a new tool, the tool is put into the list of tool requirements. If a step is deleted, and the tool used in that step is not used in any other step, the tool is removed from the list of tool requirements.

4.2. Stage 1

In stage 1 steps may be inserted in the following cases. Any machining step that is inserted omits speeds, feed rates, stepovers, and pass depths.

1. The top of the part is to be used to establish the z zero point.

In this case a set0_z step is inserted as step 3 if step 3 is not already a set0_z.

2. A workpiece has a "slab" in the top.

In this case step 2 is modified if it is a set0_corner step, a face milling step is inserted to remove the slab, a set0_corner step is inserted to re-establish x and y zero, and, if the workpiece is still too tall, another face milling step is inserted to bring it down to size.

3. A workpiece is too tall.

In this case a face milling step is inserted as step 3 to bring it down to size.

4. A workpiece without a "slab" is to be machined in the pallet area, and the process plan assumes (as an automatically generated process plan will) that the workpiece is to be machined in the vise.

In this case step2, set0_corner, is modified by changing near_x and near_y.

4.3. Stage 2

In stage 2 the workpiece data is examined for existing features. If there are existing features, steps in the plan required to make the features are removed from the plan. The subfeatures of existing features are handled similarly.

Then the current tooling is examined to make sure the tool requirements of the altered version of the plan are met. If not, the module notifies the user and quits.

Then the module selects a changer slot for each step and inserts that information in the step.

Finally, speeds, feeds, stepovers, and pass depths are added to any step which requires but is missing any of these four items. If any of the four items is present, the existing values are used. Only missing values are inserted.

APPENDIX A. NC-CODE FOR THE XYZ PART

```
n0001 (ID,PROG,xyznc1,demo part,1)
n0002 g53
n0003 p69 = +1.45
n0004 p68 = +0.0
n0005 g90 g0 w(p69+(p68-10.5)) m6
n0006 p91 = 1.5
n0007 p12 = 91 m950
n0008 p90=50 p88=-.25 p89=40
n0009 p83=+17.3 p84=+7.45 p85=1
n0010 p70=0
n0011 g53 m9
n0012 g0 g90 m5 m6
n0013 g90 g0 x+36.5 y+15.0
n0014 ! Changing tool to probe for setting x_zero and y_zero
n0015 t(p89) m28 m67 m6
n0016 x(p83) y(p84)
n0017 (GSUB,OUTVWS)
n0018 p66=(p97+0.0) p67=(p98+0.0)
n0019 g56 g90 x(p66) y(p67)
n0020 p90=50 p89=40
n0021 p77=+20.3 p78=+8.925
n0022 g53 m9
n0023 g0 g90 m5 m6
n0024 g90 g0 x+36.5 y+15.0
n0025 g90 t(p89) m28 m67 m6
n0026 ! Changing tool to probe for setting z_zero
n0027 p70=0 x(p77) y(p78)
n0028 (GSUB,INTVWS)
n0029 p91 = (p92+4.424-0.05)
n0030 p12 = 91 m950
n0031 g56 g90 x(p66) y(p67)
n0032 ! Changing tool to 2.0 inch diameter face_mill
n0033 g90 g0 m6 m9
n0034 g53
n0035 g90 g0 x+36.5 y+15.0
n0036 g90 g0 s2291 f18 t15 d15 m3 m6
n0037 g56 g90 x(p66) y(p67)
n0038 x-1.1 y+0.0
n0039 m8
n0040 z+0.0
n0041 g1 x+7.1
n0042 g0 y+1.0
n0043 g1 x-1.1
n0044 g0 y+2.05
n0045 g1 x+7.1
n0046 ! 0.2 by 0.5 by 2.6 side_contour
n0047 ! Changing tool to 1.0 inch diameter end_mill
n0048 g90 g0 m6 m9
n0049 g53
n0050 g90 g0 x+36.5 y+15.0
n0051 g90 g0 s1718 t21 d21 m3 m6
n0052 g56 g90 x(p66) y(p67)
n0053 m8
n0054 x+4.74 y+2.8
n0055 g90 g0 z+0.1
n0056 g1 z+0.0 f5
n0057 g2 x+5.25 y+3.31 r+0.51 z-0.017
n0058 g1 x+5.75 y+3.31 z-0.0277
n0059 g2 x+6.26 y+2.8 r+0.51 z-0.0447
n0060 g1 x+6.26 y+0.2 z-0.1
n0061 g2 x+5.75 y-0.31 r+0.51 z-0.117
n0062 g1 x+5.25 y-0.31 z-0.1277
n0063 g2 x+4.74 y+0.2 r+0.51 z-0.1447
n0064 g1 x+4.74 y+2.8 z-0.2
n0065 g2 x+5.25 y+3.31 r+0.51
n0066 g1 x+5.75 y+3.31
n0067 g2 x+6.26 y+2.8 r+0.51
n0068 g1 x+6.26 y+0.2
n0069 g2 x+5.75 y-0.31 r+0.51
n0070 g1 x+5.25 y-0.31
n0071 g2 x+4.74 y+0.2 r+0.51
n0072 g1 x+4.74 y+2.8
n0073 g0 z+1.0
n0074 x+4.8113 y+3.06
n0075 g1 z-0.2 f40
n0076 g1 x+0.0 f17
n0077 g0 z+1.0
n0078 x+4.74 y+2.56
n0079 g1 z-0.2 f40
n0080 g1 x+0.0 f17
n0081 g0 z+1.0
n0082 x+4.74 y+2.06
n0083 g1 z-0.2 f40
n0084 g1 x+0.0 f17
n0085 g0 z+1.0
n0086 x+4.74 y+1.56
n0087 g1 z-0.2 f40
n0088 g1 x+0.0 f17
n0089 g0 z+1.0
n0090 x+4.74 y+1.06
n0091 g1 z-0.2 f40
n0092 g1 x+0.0 f17
n0093 g0 z+1.0
n0094 x+4.74 y+0.56
n0095 g1 z-0.2 f40
n0096 g1 x+0.0 f17
n0097 g0 z+1.0
n0098 x+4.7596 y+0.06
n0099 g1 z-0.2 f40
n0100 g1 x+0.0 f17
n0101 g0 z+1.0
n0102 x+4.75 y+2.8
n0103 z+0.1
n0104 g1 z-0.2 f8
n0105 f17
n0106 g2 x+5.25 y+3.3 r+0.5
n0107 g1 x+5.75 y+3.3
n0108 g2 x+6.25 y+2.8 r+0.5
n0109 g1 x+6.25 y+0.2
n0110 g2 x+5.75 y-0.3 r+0.5
n0111 g1 x+5.25 y-0.3
n0112 g2 x+4.75 y+0.2 r+0.5
n0113 g1 x+4.75 y+2.8
n0114 ! 0.015 deep text Z
n0115 ! Changing tool to 0.25 inch diameter ball_nosed_end_mill
n0116 g90 g0 m6 m9
n0117 g53
n0118 g90 g0 x+36.5 y+15.0
n0119 g90 g0 s3437 t5 d5 m3 m6
n0120 g56 g90 x(p66) y(p67)
n0121 m8
n0122 x+5.405 y+0.9
n0123 z+0.1
n0124 g1 z-0.015 f4 ! Z
n0125 g1 x+5.605 y+0.9 f8
n0126 x+5.38 y+0.5
n0127 x+5.63
n0128 ! 0.015 deep text Y
```

```
n0129 g0 z+1.0
n0130 x+5.38 y+1.7
n0131 z+0.1
n0132 g1 z-0.015 f4 ! Y
n0133 g1 x+5.505 y+1.5 f8
n0134 g0 z+1.0
n0135 x+5.63 y+1.7
n0136 z+0.1
n0137 g1 z-0.015 f4
n0138 x+5.505 y+1.5 f8
n0139 y+1.3
n0140 ! 0.015 deep text X
n0141 g0 z+1.0
n0142 x+5.38 y+2.1
n0143 z+0.1
n0144 g1 z-0.015 f4 ! X
n0145 g1 x+5.605 y+2.5 f8
n0146 g0 z+1.0
n0147 x+5.405 y+2.5
n0148 z+0.1
n0149 g1 z-0.015 f4
n0150 x+5.63 y+2.1 f8
n0151 ! 0.3 by 3.05 by 1 pocket
n0152 ! Changing tool to 0.5625 inch diameter end_mill
n0153 g90 g0 m6 m9
n0154 g53
n0155 g90 g0 x+36.5 y+15.0
n0156 g90 g0 s3055 t18 d18 m3 m6
n0157 g56 g90 x(p66) y(p67)
n0158 m8
n0159 x+4.0 y+1.5
n0160 g0 z-0.1
n0161 g1 z-0.2 f5
n0162 x+1.95 y+1.5 z-0.4813
n0163 x+4.0 y+1.5 z-0.4813
n0164 x+1.95 y+1.5 z-0.5
n0165 x+4.0 y+1.5
n0166 p64=0
n0167 p65=(p64+1)
n0168 p64=p65
n0169 p65=(-0.2 - (p64*0.15))
n0170 g0 z+1.0
n0171 x+4.0 y+1.5
n0172 z-0.1
n0173 g1 z(p65) f40 m8 m72
n0174 f17
n0175 g1 x+4.2 y+1.2913
n0176 g1 x+1.75
n0177 g2 x+1.7413 y+1.3 r+0.0088
n0178 g1 y+1.7
n0179 g2 x+1.75 y+1.7088 r+0.0088
n0180 g1 x+4.2
n0181 g2 x+4.2088 y+1.7 r+0.0088
n0182 g1 y+1.3
n0183 g2 x+4.2 y+1.2913 r+0.0088
n0184 (IF (p64 < 2) GOTO n0167)
n0185 g0 z+1.0
n0186 x+4.2 y+1.2813
n0187 z-0.1
n0188 g1 z-0.5 f8
n0189 f17
n0190 g3 x+4.2187 y+1.3 r+0.0187
n0191 g1 y+1.7
n0192 g3 x+4.2 y+1.7188 r+0.0187
n0193 g1 x+1.75
n0194 g3 x+1.7313 y+1.7 r+0.0187
n0195 g1 y+1.3
n0196 g3 x+1.75 y+1.2813 r+0.0187
n0197 g1 x+4.2
n0198 ! 0.75 by 5.2946 by 2.6 side_contour
n0199 ! Changing tool to 0.5 inch diameter end_mill
n0200 g90 g0 m6 m9
n0201 g53
n0202 g90 g0 x+36.5 y+15.0
n0203 g90 g0 s3437 t2 d2 m3 m6
n0204 g56 g90 x(p66) y(p67)
n0205 m8
n0206 x+0.4855 y+1.8638
n0207 g90 g0 z-0.1
n0208 g1 z-0.2 f5
n0209 g2 x+0.9277 y+3.0523 r+0.6515 z-0.218
n0210 g1 x+4.9788 y+2.5459 z-0.2602
n0211 g3 x+4.99 y+2.5559 r+0.01 z-0.2604
n0212 g1 x+4.99 y+2.8 z-0.2629
n0213 g2 x+5.25 y+3.06 r+0.26 z-0.2671
n0214 g1 x+5.75 y+3.06 z-0.2723
n0215 g2 x+6.01 y+2.8 r+0.26 z-0.2765
n0216 g1 x+6.01 y+0.2 z-0.3034
n0217 g2 x+5.75 y-0.06 r+0.26 z-0.3076
n0218 g1 x+5.25 y-0.06 z-0.3128
n0219 g2 x+4.99 y+0.2 r+0.26 z-0.317
n0220 g1 x+4.99 y+0.4441 z-0.3195
n0221 g3 x+4.9788 y+0.4541 r+0.01 z-0.3197
n0222 g1 x+0.9277 y-0.0523 z-0.3619
n0223 g2 x+0.4855 y+1.1362 r+0.6515 z-0.3799
n0224 g3 x+0.49 y+1.1445 r+0.01 z-0.38
n0225 g1 x+0.49 y+1.8555 z-0.3874
n0226 g3 x+0.4855 y+1.8638 r+0.01 z-0.3875
n0227 g2 x+0.9277 y+3.0523 r+0.6515 z-0.4055
n0228 g1 x+4.9788 y+2.5459 z-0.4477
n0229 g3 x+4.99 y+2.5559 r+0.01 z-0.4479
n0230 g1 x+4.99 y+2.8 z-0.4504
n0231 g2 x+5.25 y+3.06 r+0.26 z-0.4546
n0232 g1 x+5.75 y+3.06 z-0.4598
n0233 g2 x+6.01 y+2.8 r+0.26 z-0.464
n0234 g1 x+6.01 y+0.2 z-0.4909
n0235 g2 x+5.75 y-0.06 r+0.26 z-0.4951
n0236 g1 x+5.25 y-0.06 z-0.5003
n0237 g2 x+4.99 y+0.2 r+0.26 z-0.5045
n0238 g1 x+4.99 y+0.4441 z-0.507
n0239 g3 x+4.9788 y+0.4541 r+0.01 z-0.5072
n0240 g1 x+0.9277 y-0.0523 z-0.5494
n0241 g2 x+0.4855 y+1.1362 r+0.6515 z-0.5674
n0242 g3 x+0.49 y+1.1445 r+0.01 z-0.5675
n0243 g1 x+0.49 y+1.8555 z-0.5749
n0244 g3 x+0.4855 y+1.8638 r+0.01 z-0.575
n0245 g2 x+0.9277 y+3.0523 r+0.6515 z-0.593
n0246 g1 x+4.9788 y+2.5459 z-0.6352
n0247 g3 x+4.99 y+2.5559 r+0.01 z-0.6354
n0248 g1 x+4.99 y+2.8 z-0.6379
n0249 g2 x+5.25 y+3.06 r+0.26 z-0.6421
n0250 g1 x+5.75 y+3.06 z-0.6473
n0251 g2 x+6.01 y+2.8 r+0.26 z-0.6515
n0252 g1 x+6.01 y+0.2 z-0.6784
n0253 g2 x+5.75 y-0.06 r+0.26 z-0.6826
n0254 g1 x+5.25 y-0.06 z-0.6878
n0255 g2 x+4.99 y+0.2 r+0.26 z-0.692
n0256 g1 x+4.99 y+0.4441 z-0.6945
n0257 g3 x+4.9788 y+0.4541 r+0.01 z-0.6947
n0258 g1 x+0.9277 y-0.0523 z-0.7369
n0259 g2 x+0.4855 y+1.1362 r+0.6515 z-0.7549
n0260 g3 x+0.49 y+1.1445 r+0.01 z-0.755
n0261 g1 x+0.49 y+1.8555 z-0.7624
n0262 g3 x+0.4855 y+1.8638 r+0.01 z-0.7625
```

```
n0263 g2 x+0.9277 y+3.0523 r+0.6515 z-0.7805
n0264 g1 x+4.9788 y+2.5459 z-0.8227
n0265 g3 x+4.99 y+2.5559 r+0.01 z-0.8229
n0266 g1 x+4.99 y+2.8 z-0.8254
n0267 g2 x+5.25 y+3.06 r+0.26 z-0.8296
n0268 g1 x+5.75 y+3.06 z-0.8348
n0269 g2 x+6.01 y+2.8 r+0.26 z-0.839
n0270 g1 x+6.01 y+0.2 z-0.8659
n0271 g2 x+5.75 y-0.06 r+0.26 z-0.8701
n0272 g1 x+5.25 y-0.06 z-0.8753
n0273 g2 x+4.99 y+0.2 r+0.26 z-0.8795
n0274 g1 x+4.99 y+0.4441 z-0.882
n0275 g3 x+4.9788 y+0.4541 r+0.01 z-0.8822
n0276 g1 x+0.9277 y-0.0523 z-0.9244
n0277 g2 x+0.4855 y+1.1362 r+0.6515 z-0.9424
n0278 g3 x+0.49 y+1.1445 r+0.01 z-0.9425
n0279 g1 x+0.49 y+1.8555 z-0.9499
n0280 g3 x+0.4855 y+1.8638 r+0.01 z-0.95
n0281 g2 x+0.9277 y+3.0523 r+0.6515
n0282 g1 x+4.9788 y+2.5459
n0283 g3 x+4.99 y+2.5559 r+0.01
n0284 g1 x+4.99 y+2.8
n0285 g2 x+5.25 y+3.06 r+0.26
n0286 g1 x+5.75 y+3.06
n0287 g2 x+6.01 y+2.8 r+0.26
n0288 g1 x+6.01 y+0.2
n0289 g2 x+5.75 y-0.06 r+0.26
n0290 g1 x+5.25 y-0.06
n0291 g2 x+4.99 y+0.2 r+0.26
n0292 g1 x+4.99 y+0.4441
n0293 g3 x+4.9788 y+0.4541 r+0.01
n0294 g1 x+0.9277 y-0.0523
n0295 g2 x+0.4855 y+1.1362 r+0.6515
n0296 g3 x+0.49 y+1.1445 r+0.01
n0297 g1 x+0.49 y+1.8555
n0298 g3 x+0.4855 y+1.8638 r+0.01
n0299 p64=0
n0300 p65=(p64+1)
n0301 p64=p65
n0302 p65=(-0.2 - (p64*0.25))
n0303 g0 z+1.0
n0304 x+6.35 y+3.185
n0305 g1 z(p65) f40
n0306 g1 x+0.0 f17
n0307 g0 z+1.0
n0308 x+5.0278 y+2.935
n0309 g1 z(p65) f40
n0310 g1 x+1.8662 f17
n0311 g0 z+1.0
n0312 x+0.4669 y+2.935
n0313 g1 z(p65) f40
n0314 g1 x+0.0 f17
n0315 g0 z+1.0
n0316 x+4.99 y+2.685
n0317 g1 z(p65) f40
n0318 g1 x+3.8662 f17
n0319 g0 z+1.0
n0320 x+0.2583 y+2.685
n0321 g1 z(p65) f40
n0322 g1 x+0.0 f17
n0323 g0 z+1.0
n0324 x+0.3967 y+1.935
n0325 g1 z(p65) f40
n0326 g1 x+0.0 f17
n0327 g0 z+1.0
n0328 x+0.49 y+1.685
n0329 g1 z(p65) f40
n0330 g1 x+0.0 f17
n0331 g0 z+1.0
n0332 x+0.49 y+1.435
n0333 g1 z(p65) f40
n0334 g1 x+0.0 f17
n0335 g0 z+1.0
n0336 x+0.49 y+1.185
n0337 g1 z(p65) f40
n0338 g1 x+0.0 f17
n0339 g0 z+1.0
n0340 x+0.2917 y+0.935
n0341 g1 z(p65) f40
n0342 g1 x+0.0 f17
n0343 g0 z+1.0
n0344 x+4.9904 y+0.185
n0345 g1 z(p65) f40
n0346 g1 x+2.8262 f17
n0347 g0 z+1.0
n0348 x+0.3399 y+0.185
n0349 g1 z(p65) f40
n0350 g1 x+0.0 f17
n0351 (IF (p64 < 3) GOTO n0300)
n0352 g0 z+1.0
n0353 x+0.4911 y+1.8721
n0354 z-0.1
n0355 g1 z-0.95 f8
n0356 f17
n0357 g2 x+0.9265 y+3.0424 r+0.6415
n0358 g1 x+4.9775 y+2.536
n0359 g3 x+5.0 y+2.5559 r+0.02
n0360 g1 x+5.0 y+2.8
n0361 g2 x+5.25 y+3.05 r+0.25
n0362 g1 x+5.75 y+3.05
n0363 g2 x+6.0 y+2.8 r+0.25
n0364 g1 x+6.0 y+0.2
n0365 g2 x+5.75 y-0.05 r+0.25
n0366 g1 x+5.25 y-0.05
n0367 g2 x+5.0 y+0.2 r+0.25
n0368 g1 x+5.0 y+0.4441
n0369 g3 x+4.9775 y+0.464 r+0.02
n0370 g1 x+0.9265 y-0.0424
n0371 g2 x+0.4911 y+1.1279 r+0.6415
n0372 g3 x+0.5 y+1.1445 r+0.02
n0373 g1 x+0.5 y+1.8555
n0374 g3 x+0.4911 y+1.8721 r+0.02
n0375 ! 0.1 by 3.875 by 2.2141 contour_groove of 0.125 width
n0376 ! Changing tool to 0.125 inch diameter end_mill
n0377 g90 g0 m6 m9
n0378 g53
n0379 g90 g0 x+36.5 y+15.0
n0380 g90 g0 s5200 t11 d11 m3 m6
n0381 g56 g90 x(p66) y(p67)
n0382 m8
n0383 x+1.25 y+2.344
n0384 g90 g0 z-0.1
n0385 g1 z-0.2 f3
n0386 g2 x+1.4744 y+2.5425 r+0.2 z-0.2016
n0387 g1 x+4.8244 y+2.1315 z-0.2176
n0388 g2 x+5.0 y+1.933 r+0.2 z-0.2189
n0389 g1 x+5.0 y+1.0479 z-0.2231
n0390 g2 x+4.8233 y+0.8493 r+0.2 z-0.2245
n0391 g1 x+1.4733 y+0.4562 z-0.2404
n0392 g2 x+1.25 y+0.6548 r+0.2 z-0.242
n0393 g1 x+1.25 y+2.344 z-0.25
n0394 g2 x+1.4744 y+2.5425 r+0.2 z-0.2516
n0395 g1 x+4.8244 y+2.1315 z-0.2676
n0396 g2 x+5.0 y+1.933 r+0.2 z-0.2689
```

```
n0397 g1 x+5.0 y+1.0479 z-0.2731
n0398 g2 x+4.8233 y+0.8493 r+0.2 z-0.2745
n0399 g1 x+1.4733 y+0.4562 z-0.2904
n0400 g2 x+1.25 y+0.6548 r+0.2 z-0.292
n0401 g1 x+1.25 y+2.344 z-0.3
n0402 g2 x+1.4744 y+2.5425 r+0.2 z-0.3
n0403 g1 x+4.8244 y+2.1315
n0404 g2 x+5.0 y+1.933 r+0.2
n0405 g1 x+5.0 y+1.0479
n0406 g2 x+4.8233 y+0.8493 r+0.2
n0407 g1 x+1.4733 y+0.4562
n0408 g2 x+1.25 y+0.6548 r+0.2
n0409 g1 x+1.25 y+2.344
n0410 ! Changing tool to 0.1719 inch diameter drill
n0411 g90 g0 m6 m9
n0412 g53
n0413 g90 g0 x+36.5 y+15.0
n0414 g90 g0 f4 t25 d25 m3 m6
n0415 g56 g90 x(p66) y(p67)
n0416 x+0.875 y+2.375
n0417 m8
n0418 g83 r-0.1 z-1.0 d25 p1=0.1719 ! 0.8 deep hole
n0419 g0 z+1.0
n0420 x+0.875 y+0.625
n0421 g83 r-0.1 z-1.0 f4 d25 m8 p1=0.1719 ! 0.8 deep hole
n0422 ! Changing tool to 0.375 inch diameter chamfer
n0423 g90 g0 m6 m9
n0424 g53
n0425 g90 g0 x+36.5 y+15.0
n0426 g90 g0 f29 t6 d6 m3 m6
n0427 g56 g90 x(p66) y(p67)
n0428 x+1.75 y+1.0938
n0429 m8
n0430 g0 z-0.1 ! 0.04 wide chamfer
n0431 g1 z-0.3338
n0432 g1 x+4.2
n0433 g3 x+4.4062 y+1.3 r+0.2062
n0434 g1 y+1.7
n0435 g3 x+4.2 y+1.9062 r+0.2062
n0436 g1 x+1.75
n0437 g3 x+1.5438 y+1.7 r+0.2062
n0438 g1 y+1.3
n0439 g3 x+1.75 y+1.0938 r+0.2062
n0440 ! Changing tool to 0.75 inch diameter countersink
n0441 g90 g0 m6 m9
n0442 g53
n0443 g90 g0 x+36.5 y+15.0
n0444 g90 g0 s2291 f5 t4 d4 m3 m6
n0445 g56 g90 x(p66) y(p67)
n0446 x+0.875 y+0.625
n0447 m8
n0448 g82 r-0.1 z-0.3921 d4 p3=.5 ! 0.35 dia
n0449 g0 z+1.0
n0450 x+0.875 y+2.375
n0451 g82 r-0.1 z-0.3921 f5 d4 p3=.5 ! 0.35 dia
n0452 ! Changing tool to 0.19 inch diameter tap with 24 tpi
n0453 g90 g0 m6 m9
n0454 g53
n0455 g90 g0 x+36.5 y+15.0
n0456 g90 g0 s376 t26 d26 m3 m6
n0457 g56 g90 x(p66) y(p67)
n0458 x+0.875 y+0.625
n0459 m8
n0460 g84 r-0.1 z-0.8 f300 d26 ! 0.6 deep thread
n0461 g90 g0 z+1.0
n0462 x+0.875 y+2.375
n0463 g84 r-0.1 z-0.8 f300 d26 ! 0.6 deep thread
n0464 ! 0.3 by 0.66 by 0.8 contour_pocket
n0465 ! Changing tool to 0.25 inch diameter end_mill
n0466 g90 g0 m6 m9
n0467 g53
n0468 g90 g0 x+36.5 y+15.0
n0469 g90 g0 s5200 t19 d19 m3 m6
n0470 g56 g90 x(p66) y(p67)
n0471 m8
n0472 x+1.81 y+1.765
n0473 g90 g0 z-0.4
n0474 g1 z-0.5 f4
n0475 g3 x+1.8076 y+1.7556 r+0.005 z-0.5006
n0476 g2 x+1.8076 y+1.2444 r+0.2897 z-0.531
n0477 g3 x+1.81 y+1.235 r+0.005 z-0.5316
n0478 g1 x+2.19 y+1.235 z-0.55
n0479 g3 x+2.1924 y+1.2444 r+0.005 z-0.5506
n0480 g2 x+2.1924 y+1.7556 r+0.2897 z-0.581
n0481 g3 x+2.19 y+1.765 r+0.005 z-0.5816
n0482 g1 x+1.81 y+1.765 z-0.6
n0483 g3 x+1.8076 y+1.7556 r+0.005 z-0.6006
n0484 g2 x+1.8076 y+1.2444 r+0.2897 z-0.631
n0485 g3 x+1.81 y+1.235 r+0.005 z-0.6316
n0486 g1 x+2.19 y+1.235 z-0.65
n0487 g3 x+2.1924 y+1.2444 r+0.005 z-0.6506
n0488 g2 x+2.1924 y+1.7556 r+0.2897 z-0.681
n0489 g3 x+2.19 y+1.765 r+0.005 z-0.6816
n0490 g1 x+1.81 y+1.765 z-0.7
n0491 g3 x+1.8076 y+1.7556 r+0.005 z-0.7006
n0492 g2 x+1.8076 y+1.2444 r+0.2897 z-0.731
n0493 g3 x+1.81 y+1.235 r+0.005 z-0.7316
n0494 g1 x+2.19 y+1.235 z-0.75
n0495 g3 x+2.1924 y+1.2444 r+0.005 z-0.7506
n0496 g2 x+2.1924 y+1.7556 r+0.2897 z-0.781
n0497 g3 x+2.19 y+1.765 r+0.005 z-0.7816
n0498 g1 x+1.81 y+1.765 z-0.8
n0499 g3 x+1.8076 y+1.7556 r+0.005 z-0.8
n0500 g2 x+1.8076 y+1.2444 r+0.2897
n0501 g3 x+1.81 y+1.235 r+0.005
n0502 g1 x+2.19 y+1.235
n0503 g3 x+2.1924 y+1.2444 r+0.005
n0504 g2 x+2.1924 y+1.7556 r+0.2897
n0505 g3 x+2.19 y+1.765 r+0.005
n0506 g1 x+1.81 y+1.765
n0507 p64=0
n0508 p65=(p64+1)
n0509 p64=p65
n0510 p65=(-0.5 - (p64*0.1))
n0511 g0 z+1.0
n0512 x+2.1215 y+1.7025
n0513 g1 z(p65) f40
n0514 g1 x+1.8785 f13
n0515 g0 z+1.0
n0516 x+2.096 y+1.3275
n0517 g1 z(p65) f40
n0518 g1 x+1.904 f13
n0519 (IF (p64 < 3) GOTO n0508)
n0520 g0 z+1.0
n0521 x+1.81 y+1.775
n0522 z-0.4
n0523 g1 z-0.8 f6
n0524 f13
n0525 g3 x+1.8029 y+1.7468 r+0.015
n0526 g2 x+1.8029 y+1.2532 r+0.2797
n0527 g3 x+1.81 y+1.225 r+0.015
n0528 g1 x+2.19 y+1.225
n0529 g3 x+2.1971 y+1.2532 r+0.015
n0530 g2 x+2.1971 y+1.7468 r+0.2797
```

```
n0531 g3 x+2.19 y+1.775 r+0.015
n0532 g1 x+1.81 y+1.775
n0533 ! 0.1 by 0.125 by 0.8544 straight_groove
n0534 ! Changing tool to 0.125 inch diameter end_mill
n0535 g90 g0 m6 m9
n0536 g53
n0537 g90 g0 x+36.5 y+15.0
n0538 g90 g0 t11 d11 m3 m6
n0539 g56 g90 x(p66) y(p67)
n0540 x+2.9415 y+1.4781
n0541 m8
n0542 g0 z-0.4
n0543 g1 z-0.5 f3
n0544 x+2.2585 y+1.2219 z-0.5625
n0545 x+2.9415 y+1.4781 z-0.5625
n0546 x+2.2585 y+1.2219 z-0.6
n0547 x+2.9415 y+1.4781
n0548 ! 0.1 by 0.125 by 0.8544 straight_groove
n0549 g90 g0 z+1.0
n0550 x+2.9415 y+1.5219
n0551 g0 z-0.4
n0552 g1 z-0.5 f3
n0553 x+2.2585 y+1.7781 z-0.5625
n0554 x+2.9415 y+1.5219 z-0.5625
n0555 x+2.2585 y+1.7781 z-0.6
n0556 x+2.9415 y+1.5219
n0557 ! 0.3 by 1.375 by 0.7 groove
n0558 ! Changing tool to 0.25 inch diameter ball_nosed_end_mill
n0559 g90 g0 m6 m9
n0560 g53
n0561 g90 g0 x+36.5 y+15.0
n0562 g90 g0 s3437 t5 d5 m3 m6
n0563 g56 g90 x(p66) y(p67)
n0564 m8
n0565 x+4.05 y+1.275
n0566 g90 g0 z-0.4
n0567 g1 z-0.5 f8
n0568 g3 x+4.125 y+1.35 r+0.075 z-0.5039
n0569 g1 x+4.125 y+1.65 z-0.5138
n0570 g3 x+4.05 y+1.725 r+0.075 z-0.5177
n0571 g1 x+3.075 y+1.725 z-0.55
n0572 g3 x+3.0 y+1.65 r+0.075 z-0.5539
n0573 g1 x+3.0 y+1.35 z-0.5638
n0574 g3 x+3.075 y+1.275 r+0.075 z-0.5677
n0575 g1 x+4.05 y+1.275 z-0.6
n0576 g3 x+4.125 y+1.35 r+0.075 z-0.6039
n0577 g1 x+4.125 y+1.65 z-0.6138
n0578 g3 x+4.05 y+1.725 r+0.075 z-0.6177
n0579 g1 x+3.075 y+1.725 z-0.65
n0580 g3 x+3.0 y+1.65 r+0.075 z-0.6539
n0581 g1 x+3.0 y+1.35 z-0.6638
n0582 g3 x+3.075 y+1.275 r+0.075 z-0.6677
n0583 g1 x+4.05 y+1.275 z-0.7
n0584 g3 x+4.125 y+1.35 r+0.075 z-0.7039
n0585 g1 x+4.125 y+1.65 z-0.7138
n0586 g3 x+4.05 y+1.725 r+0.075 z-0.7177
n0587 g1 x+3.075 y+1.725 z-0.75
n0588 g3 x+3.0 y+1.65 r+0.075 z-0.7539
n0589 g1 x+3.0 y+1.35 z-0.7638
n0590 g3 x+3.075 y+1.275 r+0.075 z-0.7677
n0591 g1 x+4.05 y+1.275 z-0.8
n0592 g3 x+4.125 y+1.35 r+0.075 z-0.8
n0593 g1 x+4.125 y+1.65
n0594 g3 x+4.05 y+1.725 r+0.075
n0595 g1 x+3.075 y+1.725
n0596 g3 x+3.0 y+1.65 r+0.075
n0597 g1 x+3.0 y+1.35
n0598 g3 x+3.075 y+1.275 r+0.075
n0599 g1 x+4.05 y+1.275
n0600 ! Changing tool to 0.5 inch diameter drill
n0601 g90 g0 m6 m9
n0602 g53
n0603 g90 g0 x+36.5 y+15.0
n0604 g90 g0 s2100 f5 t34 d34 m3 m6
n0605 g56 g90 x(p66) y(p67)
n0606 x+3.0 y+1.5
n0607 m8
n0608 g83 r-0.4 z-1.62 d34 p1=0.5 ! 1.12 deep hole
n0609 ! Changing tool to 0.375 inch diameter chamfer
n0610 g90 g0 m6 m9
n0611 g53
n0612 g90 g0 x+36.5 y+15.0
n0613 g90 g0 s5200 f29 t6 d6 m3 m6
n0614 g56 g90 x(p66) y(p67)
n0615 x+3.075 y+1.2438
n0616 m8
n0617 g0 z-0.4 ! 0.05 wide chamfer
n0618 g1 z-0.6438
n0619 g1 x+4.05
n0620 g3 x+4.1562 y+1.35 r+0.1062
n0621 g1 y+1.65
n0622 g3 x+4.05 y+1.7562 r+0.1062
n0623 g1 x+3.075
n0624 g3 x+2.9688 y+1.65 r+0.1062
n0625 g1 y+1.35
n0626 g3 x+3.075 y+1.2438 r+0.1062
n0627 g90 g0 z+1.0 f29
n0628 x+3.125 y+1.3063
n0629 g0 z-0.4 ! 0.04 wide chamfer
n0630 g1 z-0.6338
n0631 g2 x+3.0312 y+1.4 r+0.0937
n0632 g1 y+1.6
n0633 g2 x+3.125 y+1.6938 r+0.0937
n0634 g1 x+4.0
n0635 g2 x+4.0937 y+1.6 r+0.0937
n0636 g1 y+1.4
n0637 g2 x+4.0 y+1.3063 r+0.0937
n0638 g1 x+3.125
n0639 g53 m9 m5
n0640 g90 g0 w-9.0 m6
n0641 p91 = 0.0
n0642 p12 = 91 m950
n0643 g90 g0 x+0.5 y+19.5
n0644 (END,PROG)
```

REFERENCES

[ANSI]
American National Standards Institute; "Industrial Engineering Terminology, Production Planning and Control"; ANSI; 1973; p. 16.

[BR&M]
Brown, Peter F.; and Mclean, Charles R.; "Interactive Process Planning in the AMRF"; Proceedings of a Conference on Knowledge-Based Expert Systems for Manufacturing; December 1986; Anaheim, California; ASME; 1986; pp. 245 - 262.

[BR&R]
Brown, Peter F.; and Ray, Steven R.; "Research Issues in Process Planning at the National Bureau of Standards"; Proceedings of the 19th CIRP International Seminar on Manufacturing Systems; June 1987; Pennsylvania State University; not yet in print.

[CH&W]
Chang, Tien-Chien; and Wysk, Richard A.; An Introduction to Automated Process Planning Systems; New Jersey; Prentice-Hall; 1985; p. 15.

[JUN]
Jun, Jau-Shi; "The Vertical Machining Workstation Systems"; NISTIR 88-3890; National Institute of Standards and Technology; 1988; 65 pages.

[KRA1]
Kramer, Thomas R.; "Process Planning for a Milling Machine from a Feature-Based Design"; Proceedings of Manufacturing International Meeting; Atlanta, Georgia; April 1988; ASME; 1988; Vol. III, pp. 179 -189.

[KRA2]
Kramer, Thomas R.; "The Graphics Subsystem of the Vertical Workstation of the Automated Manufacturing Research Facility at the National Bureau of Standards; NBSIR 88-3783; National Bureau of Standards; 1988; 27 pages.

[KRA3]
Kramer, Thomas R.; "Data Handling in the Vertical Workstation of the Automated Manufacturing Research Facility at the National Bureau of Standards"; NBSIR 88-3763; National Bureau of Standards; 1988; 62 pages.

[KRA4]
Kramer, Thomas R.; "The vws_cadm User Interface in the Vertical Workstation of the Automated Manufacturing Research Facility at the National Bureau of Standards"; NBSIR 88-3738; National Bureau of Standards; 1988; 110 pages.

[K&J1]
Kramer, Thomas R.; and Jun, Jau-Shi; "Software for an Automated Machining Workstation"; Proceedings of the 1986 International Machine Tool Technical Conference; September 1986; Chicago, Illinois; National Machine Tool Builders Association; 1986; pp. 12-9 through 12-44.

[K&J2]
Kramer, Thomas R.; and Jun, Jau-Shi; "The Design Protocol, Part Design Editor, and Geometry Library of the Vertical Workstation of the Automated Manufacturing Research Facility at the National Bureau of Standards"; NBSIR 88-3717; National Bureau of Standards; 1988; 101 pages.

[K&S1]
Kramer, Thomas R.; and Strayer, W. Timothy; "Error Prevention in Data Preparation for a Numerically Controlled Milling Machine"; Proceedings of 1987 ASME annual meeting; ASME; 1987; PED-Vol. 25; pp. 195 - 213.

[K&S2]
Kramer, Thomas R.; and Strayer, W. Timothy; "Error Prevention and Detection in Data Preparation for the Vertical Workstation Milling Machine in the Automated Manufacturing Research Facility at the National Bureau of Standards"; NBSIR 87-3677; National Bureau of Standards; 1987; 61 pages.

[KR&W]
Kramer, Thomas R., and Weaver, Rebecca E.; "The Data Execution Module of the Vertical Workstation of the Automated Manufacturing Research Facility at the National Bureau of Standards"; NBSIR 88-3704; National Bureau of Standards; 1988; 58 pages.

[LOVE]
Lovett, Denver; "The Vertical Workstation's Equipment Controllers"; NBSIR 88-3769; National Bureau of Standards; 1988; 59 pages.

[McLE]
McLean, Charles R.; "An Architecture for Intelligent Manufacturing Control"; Proceedings of Summer 1985 ASME Conference; August 1985; Boston, Massachusetts; ASME.

[NA&J]
Nakpalohpo, Ibrahim; and Jun, Jau-Shi; "Automated Equipment Program Generator and Execution System of the AMRF Vertical Workstation"; not published; 1987; 17 pages.

[NAU]
Nau, Dana S.; "Hierarchical Abstraction of Problem-Solving Knowledge"; January, 1987.

[RA&M]
Ray, Steven R.; and McLean, Charles R.; "Process Plan File Format"; not yet published; 1987.

[RUDD]
Rudder, Frederick; "Operations Manual for the Automatic Operation of the Vertical Workstation"; NISTIR 89-4031; National Institute of Standards and Technology; 1989; 33 pages.