A Specification of Manufacturing Processes for Planning

Yolanda Gil

August 1991

CMU-CS-91-179$_2$

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

Much research is being done on the automation of manufacturing processes. The planning component in the production stage is very significant, due to the variety of alternative processes, their complexity, and their interactions. This document describes a specification of some manufacturing processes, including the machining, joining, and finishing of parts. The aim of this specification is not to be comprehensive or detailed, but to present the AI community with a model of a complex and realistic application, and to use it to demonstrate the feasibility of effective implementations of large-scale complex domains in a general-purpose architecture. This specification has been successfully demonstrated in the PRODIGY architecture, and is one of the largest domains available for general-purpose planners.

This research was sponsored by the Avionics Laboratory, Wright Research and Development Center, Aeronautical Systems Division (AFSC), U.S. Air Force, Wright-Patterson AFB, Ohio 45433-6543 under Contract F33615-90-C-1465, ARPA Order No. 7597.

The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. government.

Keywords: Artificial intelligence, planning, applications of planning, manufacturing and process planning, Prodigy

Contents

1. Introduction

The automation of the processes involved in manufacturing is receiving increasing attention. The increasing scarcity of experts, the need for lowering manufacturing costs, and the desire to make customize products widely available are some of the reasons. Figure 1 presents the traditional steps required in production manufacturing. The design of a product involves producing a specific model that satisfies some desired set of specifications. A product is typically made of several components, also called parts. When the design is completed, production continues by planning the sequences of processes to be performed on the part. The parts are then manufactured according to the production plans. Finally, the assembly of the parts yields the desired products.

Much ongoing research is concerned with the automation of each of these processes. CAD systems are being developed to provide a helpful tool in the design stages. Planning and scheduling the operations involved in manufacturing is another major focus of research. When put to work, unexpected events may cause failures in the scheduled operations. This requires re-planning and re-scheduling on the factory floor. Automating the assembly processes requires a great deal of understanding about the interactions among different parts. These interactions determine the appropriate sequences of the assembly. All these research areas contribute to the automation of manufacturing processes.

Our main focus is on the production step of production manufacturing. As we mentioned, this step requires planning to find a sequence of operations that produce the required design. The automation of process planning is becoming a need in industry. Experts are progressively more scarce, and it is becoming harder for them to keep up with a technology that changes very rapidly. There have been many attempts to automate different aspects of process planning (see [Chang and Wysk, 1985] for an overview), and some of them use AI techniques [Hayes, 1990, Descotte and Latombe, 1985, Nau, 1987]. The variety of alternative processes, their complexity, and their interactions make the planning task very complex. Plan quality is crucial, among other things it is important to minimize both resource consumption and execution time. The above references describe special-purpose systems that implement different aspects of process planning. We want to investigate the feasibility of efficient implementations of this complex task in a general-purpose system. This paper describes a specification of some manufacturing processes and their implementation in the PRODIGY system [Minton *et al.*, 1989a, Minton *et al.*, 1989b]. The combination of PRODIGY's learning methods with its powerful inference engine should demonstrate the feasibility of effective implementations of process planning tasks in general-purpose systems.

The main processes involved in manufacturing can be classified as:

- Casting or molding: These operations involve solidifying liquid metal in a mold. When the metal takes the shape of the cavity, the mold is opened and the part is removed.
- Forming: The purpose of forming is to modify the shape or the physical properties of the material. Forming processes include rolling (to reduce the thickness of a part) and bending.
- Machining: Machining operations are used to remove material from a part to obtain a given shape or finish.

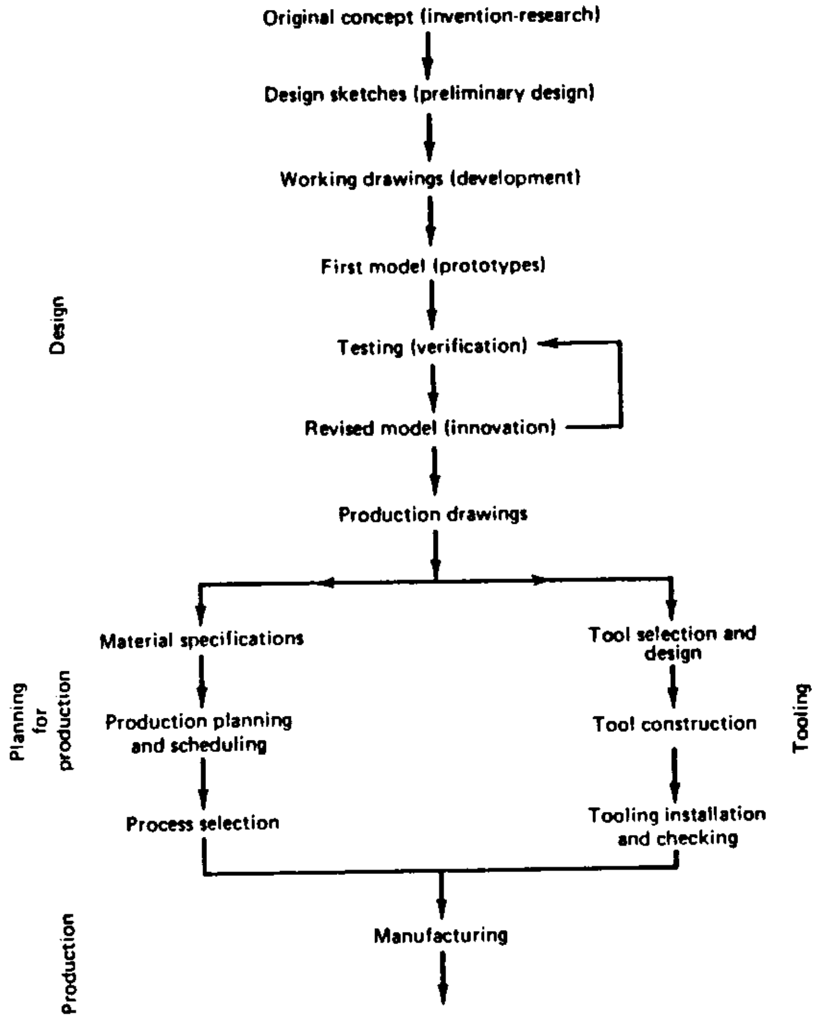

Original concept (invention-research)
Design sketches (preliminary design)
Working drawings (development)
First model (prototypes)
Testing (verification)
Revised model (innovation)
Production drawings
Material specifications
Production planning and scheduling
Process selection
Tool selection and design
Tool construction
Tooling installation and checking
Manufacturing
Design
Planning for production
Tooling
Production

Figure 1: Traditional Steps in Production Manufacturing, taken from [DeGarmo *et al.*, 1984]

- Joining and assembly: Processes for joining parts include soldering, welding, and bolting.
- Finishing: Finishing processes are employed for cleaning, removing burrs, or changing the properties of the surface on a part.

Figure 2 shows schematic representations of the first three processes.

This document describes a specification of some manufacturing processes and its implementation in a general-purpose planning system. The aim of this specification is not to be comprehensive or detailed, but to provide a starting point for the implementation. The specification concentrates on machining, joining, and finishing operations. They are performed typically in a shop where an operator produces a part that meets some given specifications. Other processes like casting and forming are usually performed elsewhere (e.g., a foundry). This makes for a natural division of the possible operations. The plans that we intend to produce specify certain sequence constraints on the operations, but we do not address scheduling issues any further [Fox, 1987]. We asked an expert machinist to assist in the construction of the domain so it would be as accurate as possible. The machinist also helped with the description of a real machine shop and sample parts for constructing problems. We begin with a description of the possible requirements for a part. Next, we show the machining, joining, and finishing processes that we consider here in more detail. Then we present our implementation in the PRODIGY architecture. We finish with some ideas about extensions for the specification and about how to use learning to achieve the desired efficiency.

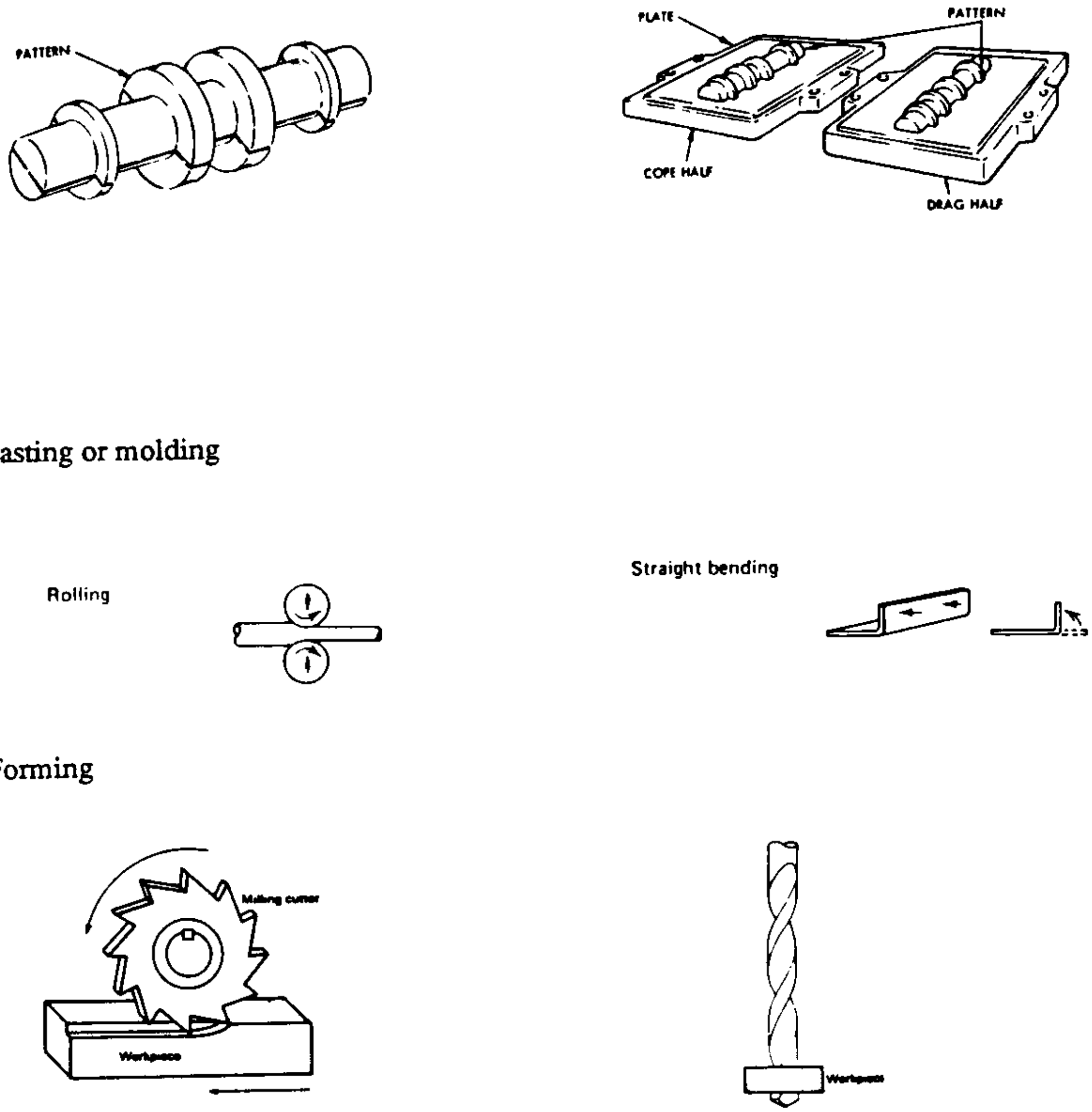
2. Specification of Requests

Requests come to the shop for specific parts. An example of a request for a part would be for a rectangular block 5" x 2" x 1" made of aluminum and with a centered hole of diameter 1/32" running through the length of the part. More generally, these requests specify:

- material
- shape
- dimensions (size)
- features that include holes, pockets, walls, etc.
- surface quality
- surface finish

Figure 3 shows an actual request. It is one of the examples included in [Hayes, 1990], selected from a job shop that serves the Mechanical Engineering Department of Carnegie Mellon University.

We now describe in more detail the possible requests along each dimension that we allow.



(a) Casting or molding

(b) Forming

(c) Machining

Figure 2: Schematic Representation of Some Manufacturing Processes (from [DeGarmo *et al.*, 1984] and [McGeough, 1988])

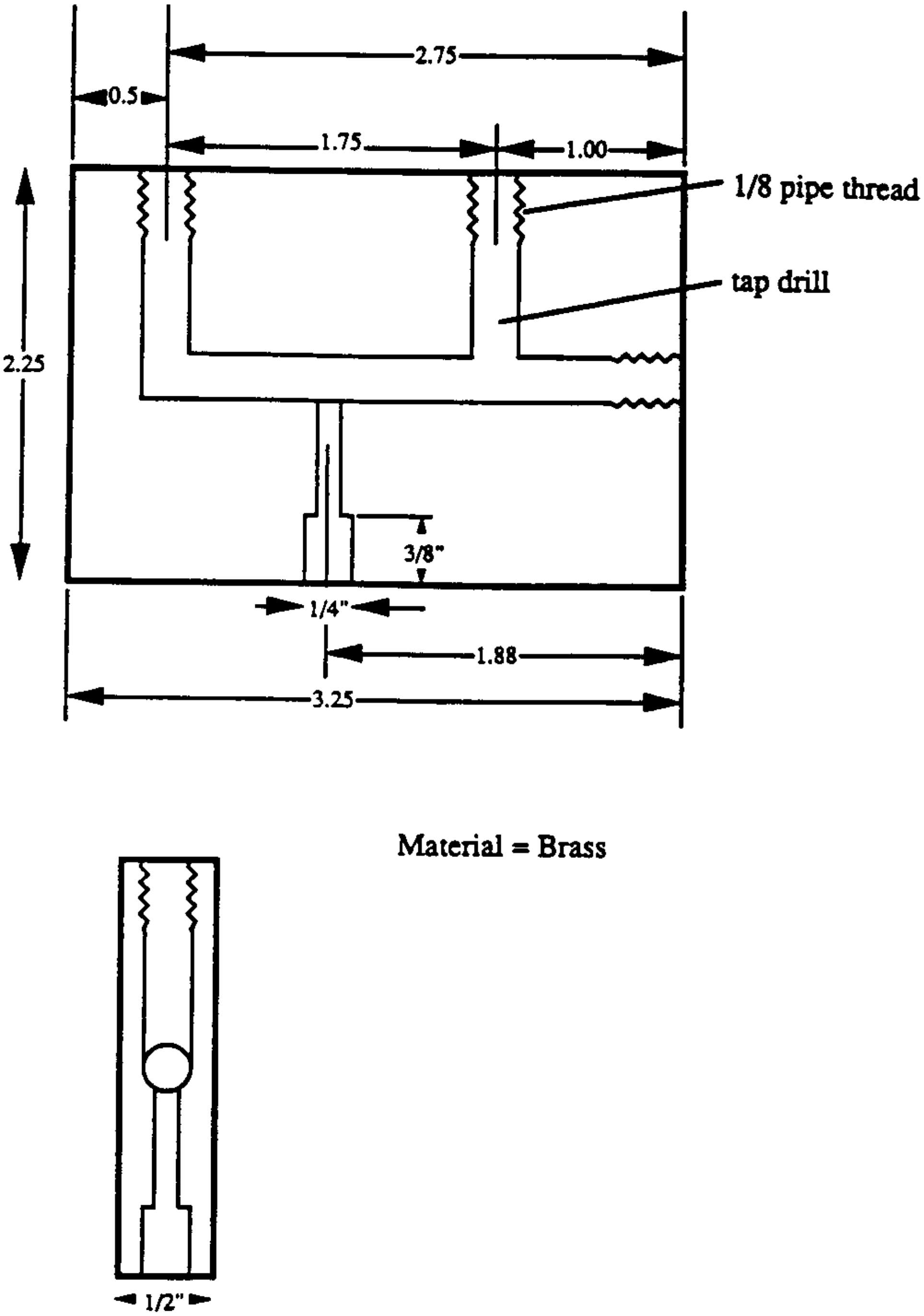


Figure 3: An Example of a Request for a Part (from [Hayes, 1990])

2.1. Geometry of the Parts

In order to simplify the process of holding various shapes, the only shapes that we consider are cylinders and rectangles. Rectangles are specified by length, width, and height. Cylinders are specified by length and diameter.

The sides of a part must be distinguishable, since they may have different properties like holes and surface conditions.

2.2. Materials of Parts

Parts can be made of different materials. The material of a part can determine which is the correct operation to be used in a process. This is why we need to specify the alloy and its hardness. Here is a list of the materials we consider, their alloy and harness.

material	*alloy*	*hardness*
steel iron	ferrous	hard
aluminum		soft
brass copper bronze plastic	non-ferrous	

2.3. Surface Quality

Different operations produce different surface finishes. A surface finish is determined by the roughness of the surface. Generally, a request specifies a range of surface finishes that are acceptable. These ranges determine the surface quality as follows:

roughness	*surface finish*	*surface quality*
500	sawcut	
250	rough mill rough turn rough shaped	machined
200	finish shaped cold rolled	
125	finish mill finish turn	
63	rough grind	ground
32	finish grind	

For example, a request for a part with machined surfaces is satisfied with any part with surfaces of roughness 250 or less (i.e. no surface has a sawcut finish).

2.4. Surface Coatings

Roughness is not the only parameter that determines the surface of a part. There may be requests for coatings of the surface. We consider here only metal coatings, which have the effect of making the surface resistant to certain factors as follows:

coating	*purpose*
stainless steel	corrosion resistance
zirconium oxide	heat resistance
aluminum oxide	wear resistance

2.5. Features

A number of features can be defined for parts. Features include holes, ribs, bosses, angles, slots (blind and through), shoulders, channels, and pockets. Features can have subfeatures, which are specifications particular to a feature. For example, a subfeature of a hole is a tap (i.e., a thread) through the hole. Once a feature is machined, the geometry of the part changes. This complicates the machining of other features in the part, and in some cases it may even render such machining impossible. These feature interactions are a major source of complication for the automation of machining processes, and we will not consider most of them in this specification.

Holes are the only feature that we will include. There are two different kinds of holes: *through* holes and *blind* holes. A hole is described by the following parameters:

- side
- x coordinate in that side
- y coordinate in that side
- diameter. The range of this parameter is determined by the set of available tools for drilling holes. (See Appendix 2).
- depth

In this specification, the following processes produce subfeatures for holes, illustrated in Figure 4:

- *Reaming* improves the finish of the interior of an existing hole.

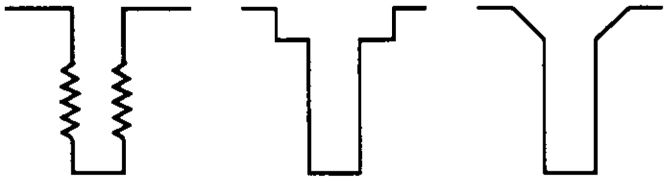


Figure 4: Possible Subfeatures

- *Tapping* an existing hole produces a thread inside.
- *Counterboring* enlarges the top of a hole for the head of a screw.
- *Countersinking* cuts an angular opening into the end of a hole.

2.6. Tolerance Levels

The requirements for the part also describe the accuracy with which the specifications should be met. The error margin for a measurement is called the *tolerance*. When the tolerance is very small, the machinist performs the operation and measures the results repeatedly until the desired tolerance is met. Tolerance levels also present additional complications. The selection of tools and machines is greatly affected by the tolerances specified. In addition, some operations may have to be repeated to improve the accuracy of a measurement.

3. Machining, Joining, and Finishing Operations

The first thing to do with the work after receiving it from the foundry is to come up with a *squaring plan*. A squaring plan is a sequence of operations that will produce a block that has perpendicular sides and accurate measurements. Squaring is necessary because some surface finishes that are very rough do not allow accurate measurements. Squaring plans also ensure that the faces are perpendicular, so the measurements will not be distorted. After the squaring plan is executed, the part is ready to be processed. Squaring plans require detailed geometric representations that we do not want to address, so we do not include them in the model.

This section describes the machining, joining, and finishing processes that the model considers.

3.1. Machining Processes

Figure 5 presents a schematic representation of machining processes. Most of these operations are used to remove material from a part.

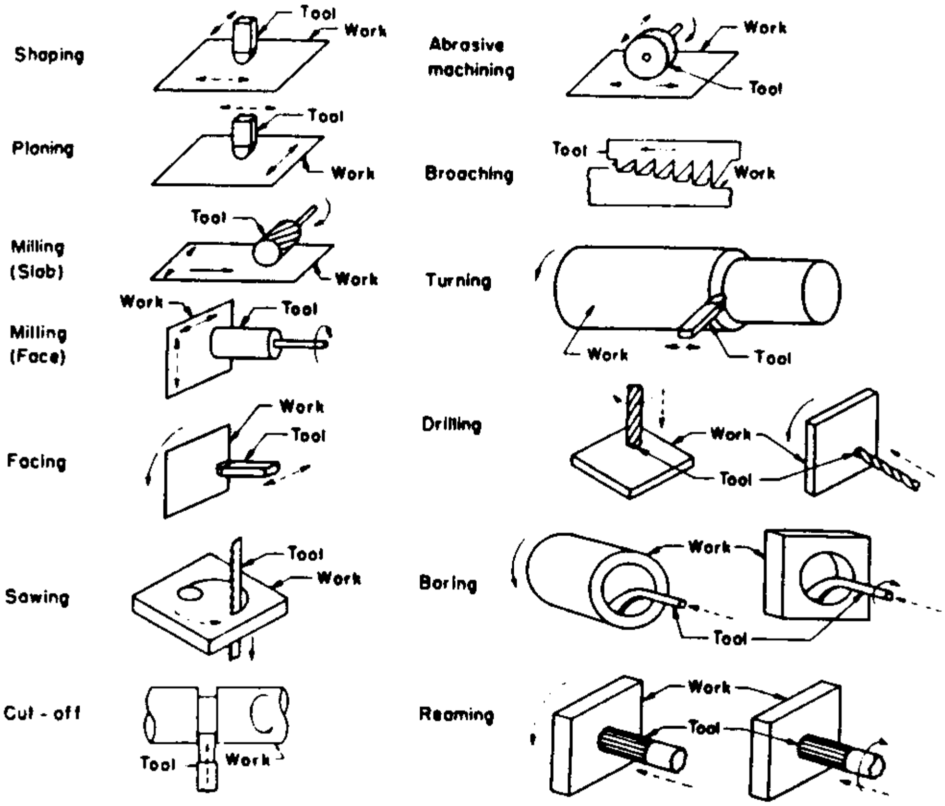


Figure 5: Schematic Representation of Some Machining Processes (from [DeGarmo *et al.*, 1984])

Each machine uses different tools, such as drill bits, grinding wheels, and saw blades. The tool used in an operation depends on the material of the part and on the surface finish that is desired. Different tools have different effects. Most of these tools can be used in several machines and for several machining operations. This makes planning for this operations very complex.

We introduce now in more detail the machining processes considered in this model.

3.1.1. Drill

Drilling machines are used to cut holes. Other drilling operations can be used to enlarge and finish holes. The tool used is called a *drill-bit*.

First, a small mark is made at the location of a desired hole. This mark is called a *spot hole*, and it guides the head of the drill bit. Spot holes are made using machines with spot drills. Spot drills come in generic diameters, and they can be used for any diameter hole.

Many types of drill bits are available for drilling holes, including twist, straight-fluted, high-helix, oil-hole, and gun drills. The drill bits used must have the same diameter of the desired hole (or smaller if the hole requires a certain finish). Twist drills are the most common kind. Straight-fluted drills work well with brass parts, and they are used of the depth of the hole is at most

2 inches. Oil-hole drills can be used for holes of a depth up to 20 inches, and they also require the use of some fluid. High-helix and gun drills require some fluid in the machine to absorb the heat.

A summary of the drilling operations is presented in the following table:

Drill bits	*Purpose*
spot-drill	drill a spot hole in any material
twist-drill	drill a hole into any material
straight-fluted	drill a hole into BRASS, holes of depth < 2
high-helix	drill a hole into any material, requires fluid
oil hole	drill a hole of less than 20" deep, requires fluid
gun	drill a hole into any material, holes of up to 20" deep, high precision holes, requires fluid
tap	produce a thread inside a hole
counterbore	enlarge the top of a hole for the head of a screw
countersink	cut an angular opening into the end of a hole
ream	enlarge a hole accurately

3.1.2. Mill

A milling machine can perform a variety of operations, but it will only be used here to to drill holes and cut parts along a certain dimension.

Milling machines are used to reduce the size of a part using a milling cutter. Two different operations are possible depending on how the tool moves with respect to the part: side mill and face mill.

Drilling with a milling machine is achieved using a drill bit as a tool. The operation is similar to using a drilling machine.

Operation	*Purpose*
face-mill	changes height of the part
	producing a ROUGH-MILL surface finish
side-mill	changes width and length of the part
	producing a ROUGH-MILL surface finish
drill with a twist drill	drill a hole into any material
drill a spot drill	drill a spot hole

3.1.3. Lathe

A lathe is used for turning operations as follows:

Operation	*Purpose*	*Tool*
rough-turn	makes any shape into a cylindrical shape, and if already cylindrical it makes diameter smaller.	rough toolbit
	It produces a surface finish ROUGH-TURN.	
finish-turn	makes diameter of a cylindrical part smaller and surface finish FINISH-TURN.	finish toolbit
make-thread	makes the surface of the cylinder TAPPED	V-thread
knurl	makes the surface of the cylinder KNURLED	knurl
file	makes the surface of the cylinder ROUGH-GRIND	lathe file

In our specification, only the lathe can change the shape of a part from rectangular to cylindrical, by turning it with a rough tool.

3.1.4. Shapers and Planers

Shapers and planers are used to reduce the size of rectangular parts. Shapers and planers do the same operations, but a planer is used for larger parts (up to 48"x 48"x 14") or for working on several parts at once. Also, a shaper is easier to use for angle shapes.

Operation	*Purpose*
rough shaping or planing	cutting producing a ROUGH surface finish
finish shaping or planing	cutting producing a FINISHED surface finish

3.1.5. Grinder

Grinding operations reduce the size of parts. Grinders are used with a tool called a *grinding wheel*. For soft parts we need to use grinding wheels of hard material, and for hard parts we need grinding wheels of soft materials. Grinders change the surface finish of the part depending on the *grit* of the wheel.

Grit of wheel	*Purpose*
fine	cutting producing a surface finish FINISH-GRIND
coarse	cutting producing a surface finish ROUGH-GRIND

Operation	*Purpose*
rough grind	cutting producing a ROUGH-GRIND surface finish
finish grind	cutting producing a FINISH-GRIND surface finish

3.1.6. Circular Saw

Circular saws are used to reduce the size of parts. Two different tools can be used: a cold saw or a friction saw. Each produces a different surface finish. Friction saws require the use of cutting fluid.

Tool	*Purpose*
cold saw	cutting producing a surface finish FINISH-MILL
friction saw	cutting producing a surface finish ROUGH-MILL

3.1.7. Band Saw

Band saws are used to reduce the size of parts. A saw band can be used to cut a part along some dimension, and it produces a sawcut finish. A band file can be used to produce a polished surface finish.

Tool	*Purpose*
saw band	cutting producing a surface finish SAWCUT
band file	cutting producing a surface finish POLISHED

3.2. Joining Processes

Welding is the only joining process considered. In order to limit the shapes to cylindrical and rectangular, welding is restricted as follows: (a) A rectangular part cannot be welded to a cylindrical part, and (b) Two cylindrical parts can only be welded at their bases.

We consider two types of welding processes:

- metal arc welding
- gas welding

3.2.1. Metal Arc Welder

A metal-arc welder uses an electrode to join two parts. The electrode is consumed in the process, and it leaves a rough edge around the union.

3.2.2. Cylinder Gas Welder

Gas welding is done using an oxygen tank, a regulator, and a torch. The operator must hold a welding rod, that is consumed in the process and leaves a rough edge around the union.

3.3. Finishing Processes

A surface can have several different finishes:

- Polishing a surface reduces its roughness to improve the appearance.
- Metal coating is used for making a surface resistant to certain factors like heat, corrosion, or wear.

3.3.1. Polished Surfaces

A part can be polished by a lathe using an emery cloth, and by a band saw using a band file.

3.3.2. Metal Coating with Spray Guns

Spray guns are used to coat surfaces. A metal wire is fed into the spray gun. The coating depends of the material of the wire:

- stainless steel makes the surface corrosion resistant
- zirconium oxide makes the surface heat resistant
- aluminum oxide makes the surface wear resistant

A surface must be prepared before it can be coated. This preparation involves cleaning dirt and grease from it, and spraying a high melting point metal (e.g., tungsten or molybdenum) that fuses to the surface and makes subsequent coats stick.

3.4. Surface Conditioning Processes

Some processes are needed for preparing the condition of the surfaces of a part:

- Cleaning is necessary to remove the dirt and grease on a part.
- Deburring is necessary after machining operations that leave sharp edges on the part. This operation is done with a brush.

3.5. Tools

Tools are installed in each machine by attaching them to a tool holder. As we have seen, the tools used by the machines are of different materials and characteristics:

1. Milling machines use milling cutters and drill bits.
2. Drills use drill bits.
3. Lathes use rough and finish toolbits, v-threads, and knurls.
4. Shapers and planers use rough and finishing cutting tools.
5. Grinders use grinding wheels.
6. Circular saws use cold saws and friction saws.
7. Band saws use band saw blades.
8. Metal arc welders use electrodes.
9. Gas welders use welding rods.
10. Spray guns use spraying metal wires.

Some operations involve human assistance with other tools. For example, when the lathe is used for filing, the machinist holds an abrasive cloth. When the lathe is polishing a part, the machinist holds a file. Also, the machinist can remove burrs with a brush.

3.6. Cutting Fluids

When a part is machined, the tool used is in direct contact with the part's surface. Cutting fluids are used to absorb the heat that the friction between the tool and the part.

The fluid used in an operation depends on the material of the part. Soluble oils are used for steel and aluminum parts. Mineral oils are used for iron parts. Any of the above types of cutting fluids can be used with parts made of brass, bronze, or copper.

3.7. Holding Devices

Before we can perform an operation with a machine, the part has to be securely held by some holding device. First, the machinist always makes sure that the part is clean and that it has no burrs remaining from previous operations. This is so that the part can be held tightly. There are different holding devices. Each of them is appropriate for holding certain shapes, and each can be used on certain types of machines.

Holding a part is a very tricky process and can be impossible for some shapes. So the order of the operations performed by the machinist is affected by the shapes produced by those operations. For example, if we want a triangle with a hole in one side, we should drill the hole before cutting the part to shape, because a triangular shape is harder to hold.

Here is a list of which shapes every holding device can hold:

	holding device						
					chucks		
shape	*vise*	*v-block*	*toe-clamp*	*centers*	*4-jaw*	*collet*	*magnetic*
rectangular	*		*		*		*
cylindrical	*	*	*	*	*	*	

Cylindrical parts are held by vises weakly. To hold them securely, a toe clamp must be used in combination with the vise. A toe clamp should also be used when holding a cylindrical part to machine its curved side

Each machine has a work area similar to a table. A holding device can be mounted to a machine table. There are certain restrictions that are determined by the operations that each machine performs, and by the shapes of the parts that they can handle. Here is a list of which machines use which holding devices:

	holding device						
					chucks		
machine	*vise*	*v-block*	*toe-clamp*	*centers*	*4-jaw*	*collet*	*magnetic*
mill	*	*	*		*	*	
drill	*	*	*		*		
lathe				*	*	*	
shaper	*						
planer			*				
grinder	*	*					*
circular saw	*	*					
band saw							
welder	*		*				

Band saws do not need a holding device; the cutting force holds the work to the table.

Spray guns do not support holding devices. Other machines hold the part while it is being sprayed.

The centers require that the part have center holes. Center holes are drilled with a special drill bit called the center drill. The drilling machine then countersinks the hole with an angle of 60 degrees. A part needs a center hole in each of two opposite sides in order to be held by the centers.

4. An Implementation in the PRODIGY Architecture

This section begins with a brief description of PRODIGY. Then we describe how this paper's specification of process planning has been implemented in this architecture. We then give an example of how to create a plan to produce a part.

4.1. The PRODIGY Architecture

The PRODIGY [Minton *et al.*, 1989a, Minton *et al.*, 1989b] system is a general-purpose problem solver that integrates several learning mechanisms to improve performance. A problem is given by an internal state, representing the current state of the world, and a goal state. Domain knowledge is represented in a set of operators and inference rules. The operators are models of the available actions and they specify the effects of the actions under different conditions. Inference rules are used to deduce additional information from the state. PRODIGY searches for a solution using backward chaining means-ends analysis by default, but it can be configured to perform other search strategies, including breadth-first search, best-first search, and depth-first iterative deepening. PRODIGY uses a casual commitment strategy for every decision in the search process. Control rules are applied at each decision point (to choose goals, operators, etc.) to guide the search. They may express definitive selections or heuristic recommendations. If no control rules are available for a certain decision, the choice is made randomly. The problem solver is nonlinear and has a very powerful language to express both domain and control knowledge.

4.2. PRODIGY's Process Planning Domain

Appendix 1 contains the complete domain. We describe the domain here briefly.

Most operators correspond to the machining operations described in the last section. Consider, for example, an operator for face milling a part. We need to represent the fact that if we use a milling cutter on a milling machine, the height of the part will change. We can write the following operator:

```
(FACE-MILL ;; first attempt to write operator FACE-MILL
 (params (<machine> <part> <milling-cutter>))
 (preconds (and
   (is-a <machine> MILLING-MACHINE)
   (is-of-type <milling-cutter> MILLING-CUTTER)
   (holding-tool <machine> <milling-cutter>)))
 (effects (
   (add (size-of <part> HEIGHT <value>))
   (del (size-of <part> HEIGHT <value-old>)))))
```

The notation means that if the preconditions are true in the current state then we can perform the milling operation, which changes the state according to the effects listed. PRODIGY will find a way to instantiate the variables in the operator, for example `(FACE-MILL milling-machine1 part10 milling-cutter1)`. Another operator will be used whose effect will be that the milling machine is holding the milling cutter.

A more complete description of the operator would specify that the part must be held by a holding device in such a way that the HEIGHT can be machined, and that the new size of the part must be smaller than the current size. Also, any surface properties of the side being machined will disappear, and the part will have dirt and burrs. The operator is now:

```
(FACE-MILL
 (params (<machine> <part> <milling-cutter> <holding-device>
          <side> <dim> <value>))
 (preconds (and
   (is-a <machine> MILLING-MACHINE)
   (is-of-type <milling-cutter> MILLING-CUTTER)
   (same <dim> HEIGHT)
   (size-of <part> <dim> <value-old>)
   (smaller <value> <value-old>)
   (side-up-for-machining <dim> <side>)
   (holding-tool <machine> <milling-cutter>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (surface-coating-side <part> <side> <*surface-coating>))
   (del (surface-finish-side <part> <side> <s-q>))
   (add (surface-finish-side <part> <side> ROUGH-MILL))
   (add (size-of <part> <dim> <value>))
   (del (size-of <part> <dim> <value-old>)))))
```

The operators for each operation are build similarly.

There are operators to put tools in tool holders for each type of machine. There is one single operator to remove tools from tool holders.

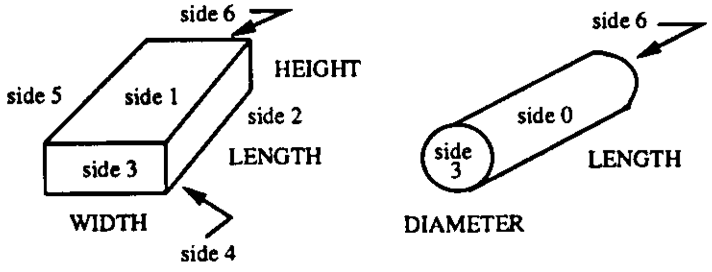
Other operators put holding devices on machine tables. There is one for each type of machine. A machine can only have one holding device on its table at a particular time. One more operator removes any holding device from any machine.

Parts have to be on the machine table before a holding device can hold them. There are operators for holding parts in each type of holding device. Only one operator is needed for releasing parts.

There are inference rules for availability of machines, parts, tools, tool holders, and holding devices. A machine is available if there is no part on the work area. A part is available if it is not being held by any holding device. Tools and holding devices are available when they are not mounted on any machine. Other inference rules are used for defining the surface finish and coating, type of material, and shapes of parts. Another set of inference rules encode a type hierarchy (e.g., the fact that a twist drill is a type of drill bit).

Another set of inference rules determines which sides should be used to hold a part. As we mentioned before, the sides of a part must be distinguishable. Hole locations are determined by the x and y coordinates of a given side. Also, for machining a part along its width it must be held so that the sides to be machined are accessible. In this specification, the sides are named as depicted in Figure 6. The inference rules mentioned would suggest, for example, that for making the height of a rectangular part smaller it must be held with its side 1 facing up.

Control rules can guide the search along the more promising paths. For example, suppose that we have the following two goals: a certain coating and a certain finish for a surface. If the coating



(a) Rectangular part

(b) Cylindrical part

Figure 6: Specifying the Sides and Surfaces of Parts

is done first, then it will be removed by the operation to do the finish so we would have to coat the surface again. Thus, the finish should be done first, and then the coating. The following control rule recommends to prefer the goal of giving a finish to a surface to the goal of coating:

```
(PREFER-FINISH-TO-COATING
     (lhs (and (current-node <node>)
               (candidate-goal <node>
                    (surface-coating-side <part> <side> <s-c>))
               (candidate-goal <node>
                    (surface-finish-side <part> <side> <s-f>))))
     (rhs (prefer goal
                    (surface-finish-side <part> <side> <s-f>)
                    (surface-coating-side <part> <side> <s-c>))))
```

Appendix 1 includes some other useful control rules.

4.3. Problem Specifications

A description of a shop with machines, tools, and parts should be the initial state of any problem. One example of such a machine shop is included in Appendix 2. It includes a wide variety of machines, tools, holding devices, and parts.

We have tested this domain in many different problems. We include here as an example a problem that corresponds to the specification of the part depicted in Figure 3. The following conjunction of conditions built from the specification constitutes the goal for this particular problem:

```
(exists (<part>) (is-a <part> PART)
   (and
      (material-of <part> BRASS)
      (size-of <part> LENGTH 3.25)
      (size-of <part> WIDTH .5)
      (size-of <part> HEIGHT 2.25)

      (is-tapped <part> hole1 side1 1 1/8 .5 .25)
(is-tapped <part> hole2 side1 1 1/8 2.25 .25)
(is-tapped <part> hole3 side6  3 1/8 .25 1.12)

(has-hole <part> hole4 side4 1 1/4  2.25 .25)
(is-counterbored <part> hole4 side4 1 1/4  2.25 .25 3/8)))
```

Here is the solution found for this problem:

```
Solution Length: 157 (81 operators)
Operator Sequence:

  put-tool-on-milling-machine milling-machine1 milling-cutter2
  put-holding-device-in-milling-machine milling-machine1 vise1
  clean part51
  put-on-machine-table milling-machine1 part51
  hold-with-vise milling-machine1 vise1 part51 side3
  side-mill milling-machine1 part51 milling-cutter2 vise1 side3 length 3.25
  release-from-holding-device milling-machine1 vise1 part51 side3
  remove-burrs part51 brush1
  clean part51
  hold-with-vise milling-machine1 vise1 part51 side2
  side-mill milling-machine1 part51 milling-cutter2 vise1 side2 width 0.5
  release-from-holding-device milling-machine1 vise1 part51 side2
  remove-burrs part51 brush1
  clean part51
  hold-with-vise milling-machine1 vise1 part51 side1
  face-mill milling-machine1 part51 milling-cutter2 vise1 side1 height 2.25
  put-in-drill-spindle drill1 spot-drill1
  release-from-holding-device milling-machine1 vise1 part51 side1
  remove-holding-device-from-machine milling-machine1 vise1
  put-holding-device-in-drill drill1 vise1
  remove-burrs part51 brush1
  clean part51
  put-on-machine-table drill1 part51
  hold-with-vise drill1 vise1 part51 side1
  drill-with-spot-drill drill1 spot-drill1 vise1 part51 hole1 side1
  remove-tool-from-machine drill1 spot-drill1
  put-in-drill-spindle drill1 twist-drill5
  drill-with-twist-drill drill1 twist-drill5 vise1 part51 hole1 side1 1 1/8
  remove-tool-from-machine drill1 twist-drill5
  put-in-drill-spindle drill1 tap6
  release-from-holding-device drill1 vise1 part51 side1
  remove-burrs part51 brush1
  clean part51
  hold-with-vise drill1 vise1 part51 side1
  tap drill1 tap6 vise1 part51 hole1
  remove-tool-from-machine drill1 tap6
  put-in-drill-spindle drill1 spot-drill1
  drill-with-spot-drill drill1 spot-drill1 vise1 part51 hole2 side1
  remove-tool-from-machine drill1 spot-drill1
  put-in-drill-spindle drill1 twist-drill5
  drill-with-twist-drill drill1 twist-drill5 vise1 part51 hole2 side1 1 1/8
  remove-tool-from-machine drill1 twist-drill5
  put-in-drill-spindle drill1 tap6
  release-from-holding-device drill1 vise1 part51 side1
  remove-burrs part51 brush1
  clean part51
  hold-with-vise drill1 vise1 part51 side1
  tap drill1 tap6 vise1 part51 hole2
  remove-tool-from-machine drill1 tap6
  put-in-drill-spindle drill1 spot-drill1
  release-from-holding-device drill1 vise1 part51 side1
  remove-burrs part51 brush1
  clean part51
  hold-with-vise drill1 vise1 part51 side6
```

```
drill-with-spot-drill drill1 spot-drill1 vise1 part51 hole3 side6
remove-tool-from-machine drill1 spot-drill1
put-in-drill-spindle drill1 twist-drill5
drill-with-twist-drill drill1 twist-drill5 vise1 part51 hole3 side6 3 1/8
remove-tool-from-machine drill1 twist-drill5
put-in-drill-spindle drill1 tap6
release-from-holding-device drill1 vise1 part51 side6
remove-burrs part51 brush1
clean part51
hold-with-vise drill1 vise1 part51 side6
tap drill1 tap6 vise1 part51 hole3
remove-tool-from-machine drill1 tap6
put-in-drill-spindle drill1 spot-drill1
release-from-holding-device drill1 vise1 part51 side6
remove-burrs part51 brush1
clean part51
hold-with-vise drill1 vise1 part51 side4
drill-with-spot-drill drill1 spot-drill1 vise1 part51 hole4 side4
remove-tool-from-machine drill1 spot-drill1
put-in-drill-spindle drill1 twist-drill13
drill-with-twist-drill drill1 twist-drill13 vise1 part51 hole4 side4 1 1/4
remove-tool-from-machine drill1 twist-drill13
put-in-drill-spindle drill1 counterbore4
release-from-holding-device drill1 vise1 part51 side4
remove-burrs part51 brush1
clean part51
hold-with-vise drill1 vise1 part51 side4
counterbore drill1 counterbore4 vise1 part51 hole4
```

5. Future Work

Our ideas for future work are divided into two subsections. First, we describe possible extensions to the domain that would make the current specification more precise. Then we describe how PRODIGY's learning mechanisms could improve the current performance of the system.

5.1. Possible Extensions of the Specification

We describe now possible extensions to the specification presented in this paper. [Hayes, 1987] and [Hayes, 1990] present alternative models that address some of these issues.

Many more machines, tools, and operations are available and can be easily added to the current domain.

This specification assumes that all the parts have been already squared. The domain could be extended to produce squaring plans.

The domain does not take into account the tolerances provided in the specifications of the parts. Accuracy of the operations is a very important factor when selecting machines that perform the same operations. For example, milling machines and drills can both make holes, but milling machines are much more accurate.

The cost associated with the different operations is a very important factor when trying to decide which operation to use. This version of the domain does not include costs.

The shape of the parts can only be cylindrical or rectangular in the current version of the domain. If more irregular shapes were allowed, the operators for holding parts would have to be changed. The first step would be to generate angle shapes with the shaper and the planer using a corner cutting tool. The milling machine can produce all kinds of complicated shapes.

Another extension to the domain could be positioning the tools. For example, one could position the drill spindle that holds a drill bit exactly at the same x- and y-coordinates of the hole required.

The specification can also be extended to acquire data about a part by collecting observations. There are several measuring devices that can be used to determine the size of a part when this particular data is not given to the system. Micrometers can be used to measure an unknown size or to check it after an operation. Telescopic gauges are used to measure a hole. First, the gauge measures the hole, and then the gauge is measured by a micrometer.

5.2. Improving Performance

PRODIGY was developed both as a general-purpose problem solving architecture and as a testbed for machine learning research. Many different learning strategies have been implemented to improve the performance of the problem solver. The combinatorial explosion of the search can be reduced in PRODIGY using several control mechanisms that include the use of control knowledge, hierarchical planning, and analogical reasoning. We describe now briefly how learning is combined with each of these mechanisms, and their use for process planning applications.

The control rules included in Appendix 1 were hand-coded but they can be acquired automatically. The STATIC module [Etzioni, 1990] constructs control rules by analyzing the domain description prior to problem solving. Then, PRODIGY/EBL [Minton, 1988] examines the problem-solving traces and acquires additional rules using explanation-based learning. All the rules are subject to a dynamic utility analysis that recommends which rules are useful and should be retained.

PRODIGY can also be used as a hierarchical problem solver. The domain knowledge is examined and multiple abstraction levels are learned automatically. To solve a problem, PRODIGY searches for a solution in the most abstract space, which contains the aspects of the domain that add the most complexity to the search. The abstract solution is then used to guide the search in the lower abstraction levels. Abstraction planning is a good approach to tame the complexity of planning domains in general [Knoblock, 1991], and it seems promising for both process planning and scheduling domains [Fox and Smith, 1984, Nau, 1987].

Another method that is being investigated is how to use similar previously solved problems to solve new ones [Carbonell and Veloso, 1988]. PRODIGY records solved problems along with the justifications for each decision taken in the process. A similarity metric is used to retrieve old episodes and a derivational analogy engine reconstructs a solution reusing the decisions where equivalent justifications hold true. Currently, PRODIGY finds solutions for process planning problems in a generative fashion, i.e., by constructing plans given a set of possible operators. PRODIGY's analogical engine could be used to implement a "variant" approach [Chang and Wysk, 1985] using predefined plans associated with families of parts, modifying them for the particular part wanted.

An orthogonal issue in learning and problem solving is the acquisition of domain knowledge. PRODIGY can engage in an apprentice-like dialogue that enables the user to specify domain knowledge, monitor the search process, and provide useful advice on line [Joseph, 1989]. Domain knowledge can also be acquired by interaction with the environment [Carbonell and Gil, 1990]. Given some initial body of knowledge and the possibility to execute actions and collect observations, autonomous acquisition of domain knowledge is possible. The system has expectations that emerge from its current knowledge. Plan execution is monitored, and when the expectations and the observations diverge, learning is triggered. PRODIGY uses experimentation strategies to correct the domain knowledge in order to prevent future failures. In fact, this research is being applied to the process planning application described in this paper [Gil, 1991].

In summary, learning in PRODIGY is combined with problem solving through the automatic acquisition of episodes useful for analogical reasoning, producing abstraction hierarchies, and learning control rules. The acquisition of domain knowledge from the environment is possible both from the user through an apprentice system and from the environment through autonomous learning by experimentation. We hope that these learning methods together with PRODIGY's powerful inference engine will produce an efficient process-planning system.

Plan optimization is desirable in many planning applications. A general-purpose planner must have mechanisms for producing acceptable resource consumption, execution time, or other constraints on the quality of plans. Their use in process planning is necessary, and we would like to investigate this issue further. Combining the operations required to produce several parts into a schedule would be one step further in optimizing the use of the resources.

6. Conclusions

We have described a specification of process planning in manufacturing that can be used by AI systems. This specification has been successfully demonstrated in the PRODIGY architecture, and is one of the largest domains available for general-purpose planners. By using this task we hope to demonstrate the feasibility of effective implementations of large-scale complex domains in a general-purpose architecture.

Acknowledgements

I would like to thank Jim Dillinger for all his help and patience showing me what is involved in a machinist's job. Caroline Hayes was most helpful for showing me the light at the end of all the tunnels I ventured into. Thanks also to the other members of the PRODIGY project for their suggestions on the implementation. Susan Finger, Kevin Knight, and Alicia Perez provided very helpful comments on drafts of this document.

References

[Carbonell and Gil, 1990] J. G. Carbonell and Y. Gil. Learning by experimentation: The operator refinement method. In *Machine Learning, An Artificial Intelligence Approach, Volume III*. Morgan Kaufmann, Irvine, CA, 1990.

[Carbonell and Veloso, 1988] J. G. Carbonell and M. M. Veloso. Integrating derivational analogy into a general problem-solving architecture. In *Proceedings of the First Workshop on Case-Based Reasoning*, Tampa, FL, May 1988. Morgan Kaufmann.

[Chang and Wysk, 1985] T. C. Chang and R. A. Wysk. *An Introduction to Automated Process Planning Systems*. Prentice-Hall, Englewood Cliffs, NJ, 1985.

[CLEVELAND, 1984] *Cleveland Cutting Tools: Catalog 1984*. Cleveland Twist Drill: an ACME Cleveland Company, Cleveland, OH, 1984.

[DeGarmo *et al.*, 1984] E. P. DeGarmo, J. T. Black, and R. A. Kohser. *Materials and Processes in Manufacturing*. Macmillan Publishing Company, New York, 1984.

[Descotte and Latombe, 1985] Y. Descotte and J. Latombe. Making compromises among antagonist constraints in a planner. *Artificial Intelligence*, 27(2), 1985.

[Etzioni, 1990] O. Etzioni. *A Structural Theory of Search Control*. PhD thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 1990.

[Fox and Smith, 1984] M. Fox and S. Smith. ISIS: A knowledge-based system for factory scheduling. *International Journal of Expert Systems*, 1(1), 1984.

[Fox, 1987] M. Fox. *Constraint-Directed Search: A Case Study of Job-Shop Scheduling*. Morgan Kaufmann, 1987.

[Gil, 1991] Y. Gil. A domain-independent framework for effective experimentation in planning. In *Proceedings of the Eight International Workshop on Machine Leaning*, Evanston, IL, 1991. Morgan Kaufmann.

[Hayes, 1987] C. Hayes. *Planning in the Machining Domain: Using Goal Interactions to Guide Search*. PhD thesis, The Robotics Institute, Carnegie Mellon University, 1987.

[Hayes, 1990] C. Hayes. *Machining Planning: a Model of an Expert Level Planning Process*. PhD thesis, The Robotics Institute, Carnegie Mellon University, 1990.

[Joseph, 1989] R. L. Joseph. Graphical knowledge acquisition. In *Proceedings of the 4th Workshop on Knowledge Acquisition For Knowledge-Based Systems*, Banff, Canada, 1989.

[Knoblock, 1991] C. A. Knoblock. *Automatically Generating Abstractions for Problem Solving*. PhD thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 1991.

[McGeough, 1988] J. A. McGeough. *Advanced Methods of Machining*. Chapman and Hall, New York, NY, 1988.

[Minton *et al.*, 1989a] S. Minton, J. G. Carbonell, C. A. Knoblock, D. R. Kuokka, O. Etzioni, and Y. Gil. Explanation-based learning: A problem solving perspective. *Artificial Intelligence*, 40(1-3):63–118, 1989.

[Minton *et al.*, 1989b] S. Minton, C. A. Knoblock, D. R. Kuokka, Y. Gil, R. L. Joseph, and J. G. Carbonell. PRODIGY 2.0: The manual and tutorial. Technical Report CMU-CS-89-146, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 1989.

[Minton, 1988] S. Minton. *Learning Search Control Knowledge: An Explanation-based Approach.* Kluwer Academic Publishers, Boston, Massachusetts, 1988.

[Nau, 1987] D. Nau. Automated process planning using hierarchical abstraction. In *1987 Texas Instruments call for papers on AI for Industrial Automation*, 1987.

[Shell and Carbonell, 1991] P. Shell and J. G. Carbonell. Empirical and analytical performance of iterative operators. In *The 13th Annual Conference of The Cognitive Science Society*. Lawrence Erlbaum Associates, 1991.

A An Implementation in PRODIGY

This appendix contains a listing of the specification described in this document as it is implemented in the PRODIGY architecture.

The implementation of this domain as well as the PRODIGY system are available upon request. Send e-mail to *prodigy@cs.cmu.edu* or write to:

The Prodigy Project
School of Computer Science
Carnegie Mellon University
Pittsburgh PA 15213

The domain can be easily implemented in other problem solvers, as an implementation in FRULEKIT shows [Shell and Carbonell, 1991]. FRULEKIT is an augmented implementation of the OPS Rete pattern matcher, and is quite different from PRODIGY. In particular, PRODIGY's inference engine works with backward chaining, while FRULEKIT works with forward chaining.

```
;;;*********************************************************
;;;*********************************************************
;;;*********************************************************
;;;  Operators


(setq *OPERATORS* '(

;;;*********************************************************
; MACHINE: DRILL

; operators for making holes

(DRILL-WITH-SPOT-DRILL
 (params (<machine> <drill-bit> <holding-device>
          <part> <hole> <side>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> DRILL)
   (is-a <drill-bit> SPOT-DRILL)
   (holding-tool <machine> <drill-bit>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (add (has-spot <part> <hole> <side> <loc-x> <loc-y>)))))

(DRILL-WITH-TWIST-DRILL
 (params (<machine> <drill-bit> <holding-device>
          <part> <hole> <side> <hole-depth> <hole-diameter>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> DRILL)
   (same <drill-bit-diameter> <hole-diameter>)
   (diameter-of-drill-bit <drill-bit> <drill-bit-diameter>)
   (is-a <drill-bit> TWIST-DRILL)
   (has-spot <part> <hole> <side> <loc-x> <loc-y>)
   (holding-tool <machine> <drill-bit>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (has-spot <part> <hole> <side> <loc-x> <loc-y>))
   (add (has-hole <part> <hole> <side> <hole-depth>
                  <hole-diameter> <loc-x> <loc-y>)))))

(DRILL-WITH-HIGH-HELIX-DRILL
 (params (<machine> <drill-bit> <holding-device>
          <part> <hole> <side> <hole-depth> <hole-diameter>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> DRILL)
   (same <drill-bit-diameter> <hole-diameter>)
   (diameter-of-drill-bit <drill-bit> <drill-bit-diameter>)
   (is-a <drill-bit> HIGH-HELIX-DRILL)
   (has-fluid <machine> <fluid> <part>)
   (has-spot <part> <hole> <side> <loc-x> <loc-y>)
   (holding-tool <machine> <drill-bit>)
```

```
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (has-spot <part> <hole> <side> <loc-x> <loc-y>))
   (add (has-hole <part> <hole> <side> <hole-depth>
                  <hole-diameter> <loc-x> <loc-y>)))))

(DRILL-WITH-STRAIGHT-FLUTED-DRILL
 (params (<machine> <drill-bit> <holding-device>
          <part> <hole> <side> <hole-depth> <hole-diameter>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> DRILL)
   (same <drill-bit-diameter> <hole-diameter>)
   (diameter-of-drill-bit <drill-bit> <drill-bit-diameter>)
   (is-a <drill-bit> STRAIGHT-FLUTED-DRILL)
   (smaller <hole-depth> 2)
   (material-of <part> BRASS)
   (has-spot <part> <hole> <side> <loc-x> <loc-y>)
   (holding-tool <machine> <drill-bit>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (has-spot <part> <hole> <side> <loc-x> <loc-y>))
   (add (has-hole <part> <hole> <side> <hole-depth>
                  <hole-diameter> <loc-x> <loc-y>)))))

(DRILL-WITH-OIL-HOLE-DRILL
 (params (<machine> <drill-bit> <holding-device>
          <part> <hole> <side> <hole-depth> <hole-diameter>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> DRILL)
   (same <drill-bit-diameter> <hole-diameter>)
   (diameter-of-drill-bit <drill-bit> <drill-bit-diameter>)
   (is-a <drill-bit> OIL-HOLE-DRILL)
   (smaller <hole-depth> 20)
   (has-fluid <machine> <fluid> <part>)
   (has-spot <part> <hole> <side> <loc-x> <loc-y>)
   (holding-tool <machine> <drill-bit>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (has-spot <part> <hole> <side> <loc-x> <loc-y>))
   (add (has-hole <part> <hole> <side> <hole-depth>
                  <hole-diameter> <loc-x> <loc-y>)))))

(DRILL-WITH-GUN-DRILL
 (params (<machine> <drill-bit> <holding-device>
          <part> <hole> <side> <hole-depth> <hole-diameter>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> DRILL)
   (same <drill-bit-diameter> <hole-diameter>)
   (diameter-of-drill-bit <drill-bit> <drill-bit-diameter>)
   (is-a <drill-bit> GUN-DRILL)
   (has-fluid <machine> <fluid> <part>)
   (has-spot <part> <hole> <side> <loc-x> <loc-y>)
   (holding-tool <machine> <drill-bit>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (has-spot <part> <hole> <side> <loc-x> <loc-y>))
   (add (has-hole <part> <hole> <side> <hole-depth>
                  <hole-diameter> <loc-x> <loc-y>)))))


(DRILL-WITH-CENTER-DRILL
 (params (<machine> <drill-bit> <holding-device> <part>
          <hole> <side> <drill-bit-diameter> <loc-x> <loc-y>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> DRILL)
   (diameter-of-drill-bit <drill-bit> <drill-bit-diameter>)
   (same <drill-bit-diameter> <hole-diameter>)
   (is-a <drill-bit> CENTER-DRILL)
   (has-spot <part> <hole> <side> <loc-x> <loc-y>)
   (holding-tool <machine> <drill-bit>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (has-spot <part> <hole> <side> <loc-x> <loc-y>))
   (add (has-hole <part> <hole> <side> 1/8
                  <hole-diameter> <loc-x> <loc-y>))
   (add (has-center-hole <part> <hole> <side> <loc-x> <loc-y>)))))

;; operators for finishing holes

(TAP
 (params (<machine> <drill-bit> <holding-device> <part> <hole>))
 (preconds (and
   (is-a <part> PAPT)
   (is-a <machine> DRILL)
   (same <drill-bit-diameter> <hole-diameter>)
   (diameter-of-drill-bit <drill-bit> <drill-bit-diameter>)
   (is-a <drill-bit> TAP)
   (has-hole <part> <hole> <side> <hole-depth> <hole-diameter>
             <loc-x> <loc-y>)
   (holding-tool <machine> <drill-bit>)
   (~ (has-burrs <part>))
   (is-clean <part>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (if (is-reamed <part> <hole> <side> <hole-depth>
                  <hole-diameter> <loc-x> <loc-y>)
       (del (is-reamed <part> <hole> <side> <hole-depth>
                       <hole-diameter> <loc-x> <loc-y>)))
   (add (is-tapped <part> <hole> <side> <hole-depth>
                   <hole-diameter> <loc-x> <loc-y>)))))

(COUNTERSINK
 (params (<machine> <drill-bit> <holding-device> <part> <hole>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> DRILL)
   (angle-of-drill-bit <drill-bit> <angle>)
   (is-a <drill-bit> COUNTERSINK)
   (has-hole <part> <hole> <side> <hole-depth> <hole-diameter>
             <loc-x> <loc-y>)
   (holding-tool <machine> <drill-bit>)
   (~ (has-burrs <part>))
   (is-clean <part>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (add (is-countersinked <part> <hole> <side> <hole-depth>
                   <hole-diameter> <loc-x> <loc-y> <angle>)))))

(COUNTERBORE
 (params (<machine> <drill-bit> <holding-device> <part> <hole>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> DRILL)
   (size-of-drill-bit <drill-bit> <counterbore-size>)
   (is-a <drill-bit> COUNTERBORE)
   (has-hole <part> <hole> <side> <hole-depth> <hole-diameter>
             <loc-x> <loc-y>)
   (holding-tool <machine> <drill-bit>)
   (~ (has-burrs <part>))
   (is-clean <part>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
```

```
   (add (is-counterbored <part> <hole> <side> <hole-depth>
           <hole-diameter> <loc-x> <loc-y> <counterbore-size>)))))

(REAM
 (params (<machine> <drill-bit> <holding-device> <part> <hole>
          <side> <hole-depth> <hole-diameter>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> DRILL)
   (same <drill-bit-diameter> <hole-diameter>)
   (diameter-of-drill-bit <drill-bit> <drill-bit-diameter>)
   (is-a <drill-bit> REAMER)
   (smaller <hole-depth> 2)
   (has-fluid <machine> <fluid> <part>)
   (has-hole <part> <hole> <side> <hole-depth> <hole-diameter>
             <loc-x> <loc-y>)
   (holding-tool <machine> <drill-bit>)
   (~ (has-burrs <part>))
   (is-clean <part>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (if (is-tapped <part> <hole> <side> <hole-depth>
                  <hole-diameter> <loc-x> <loc-y>)
       (del (is-tapped <part> <hole> <side> <hole-depth>
                       <hole-diameter> <loc-x> <loc-y>)))
   (add (is-reamed <part> <hole> <side> <hole-depth>
                   <hole-diameter> <loc-x> <loc-y>)))))


;;;**************************************************************
; MACHINE: MILLING MACHINE

(SIDE-MILL
 (params (<machine> <part> <milling-cutter> <holding-device>
          <side> <dim> <value>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> MILLING-MACHINE)
   (is-of-type <milling-cutter> MILLING-CUTTER)
   (or (same <dim> WIDTH)
       (same <dim> LENGTH))
   (size-of <part> <dim> <value-old>)
   (smaller <value> <value-old>)
   (smaller-than-2in  <value-old> <value>)
   (side-up-for-machining <dim> <side>)
   (holding-tool <machine> <milling-cutter>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (surface-coating-side <part> <side> <*surface-coating>))
   (del (surface-finish-side <part> <side> <*s-q>))
   (add (surface-finish-side <part> <side> ROUGH-MILL))
   (add (size-of <part> <dim> <value>))
   (del (size-of <part> <dim> <value-old>)))))

(FACE-MILL
 (params (<machine> <part> <milling-cutter> <holding-device>
          <side> <dim> <value>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> MILLING-MACHINE)
   (is-of-type <milling-cutter> MILLING-CUTTER)
   (same <dim> HEIGHT)
   (size-of <part> <dim> <value-old>)
   (smaller <value> <value-old>)
   (side-up-for-machining <dim> <side>)
   (holding-tool <machine> <milling-cutter>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (surface-coating-side <part> <side> <*surface-coating>))
   (del (surface-finish-side <part> <side> <*s-q>))
   (add (surface-finish-side <part> <side> ROUGH-MILL))
   (add (size-of <part> <dim> <value>))
   (del (size-of <part> <dim> <value-old>)))))

(DRILL-WITH-SPOT-DRILL-IN-MILLING-MACHINE
 (params (<machine> <drill-bit> <holding-device>
          <part> <hole> <side>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> MILLING-MACHINE)
   (is-a <drill-bit> SPOT-DRILL)
   (holding-tool <machine> <drill-bit>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (add (has-spot <part> <hole> <side> <loc-x> <loc-y>)))))
```

```
(DRILL-WITH-TWIST-DRILL-IN-MILLING-MACHINE
 (params (<machine> <drill-bit> <holding-device>
          <part> <hole> <side> <hole-depth> <hole-diameter>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> MILLING-MACHINE)
   (same <drill-bit-diameter> <hole-diameter>)
   (diameter-of-drill-bit <drill-bit> <drill-bit-diameter>)
   (is-a <drill-bit> TWIST-DRILL)
   (has-spot <part> <hole> <side> <loc-x> <loc-y>)
   (holding-tool <machine> <drill-bit>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (has-spot <part> <hole> <side> <loc-x> <loc-y>))
   (add (has-hole <part> <hole> <side> <hole-depth>
                  <hole-diameter> <loc-x> <loc-y>)))))

;;;************************************************************
; MACHINE: LATHE

(ROUGH-TURN-RECTANGULAR-PART
 (params (<machine> <part> <toolbit> <holding-device>
          <diameter-new>))
 (preconds (and
   (is-a <machine> LATHE)
   (is-a <toolbit> ROUGH-TOOLBIT)
   (shape-of <part> RECTANGULAR)
   (size-of <part> HEIGHT <h>)
   (size-of <part> WIDTH <w>)
   (smaller <diameter-new> <h>)
   (smaller <diameter-new> <w>)
   (holding-tool <machine> <toolbit>)
   (side-up-for-machining DIAMETER <side>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (size-of <part> HEIGHT <h>))
   (del (size-of <part> WIDTH <w>))
   (add (size-of <part> DIAMETER <diameter-new>))
   (del (surface-coating-side <part> SIDE1 <*surface-coating>))
   (del (surface-coating-side <part> SIDE2 <*surface-coating>))
   (del (surface-coating-side <part> SIDE4 <*surface-coating>))
   (del (surface-coating-side <part> SIDE5 <*surface-coating>))
   (del (surface-coating-side <part> SIDE0 <*surface-coating>))
   (del (surface-finish-side <part> SIDE1 <sf1>))
   (del (surface-finish-side <part> SIDE2 <sf2>))
   (del (surface-finish-side <part> SIDE4 <sf4>))
   (del (surface-finish-side <part> SIDE5 <sf5>))
   (add (surface-finish-side <part> SIDE0 ROUGH-TURN)))))

(ROUGH-TURN-CYLINDRICAL-PART
 (params (<machine> <part> <toolbit> <holding-device>
          <diameter-new>))
 (preconds (and
   (is-a <machine> LATHE)
   (is-a <toolbit> ROUGH-TOOLBIT)
   (shape-of <part> CYLINDRICAL)
   (size-of <part> DIAMETER <diameter>)
   (smaller <diameter-new> <diameter>)
   (holding-tool <machine> <toolbit>)
   (side-up-for-machining DIAMETER <side>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (size-of <part> DIAMETER <diameter>))
   (add (size-of <part> DIAMETER <diameter-new>))
   (del (surface-coating-side <part> SIDE0 <*surface-coating>))
   (del (surface-finish-side <part> SIDE0 <sf>))
   (add (surface-finish-side <part> SIDE0 ROUGH-TURN)))))

(FINISH-TURN
 (params (<machine> <part> <toolbit> <holding-device>
          <diameter-new>))
 (preconds (and
   (is-a <machine> LATHE)
   (is-a <toolbit> FINISH-TOOLBIT)
   (shape-of <part> CYLINDRICAL)
   (size-of <part> DIAMETER <diameter>)
   (finishing-size <diameter> <diameter-new>)
   (holding-tool <machine> <toolbit>)
   (~ (has-burrs <part>))
   (is-clean <part>)
   (holding <machine> <holding-device> <part> SIDE0)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (size-of <part> DIAMETER <diameter>))
   (add (size-of <part> DIAMETER <diameter-new>))
   (del (surface-coating-side <part> SIDE0 <*surface-coating>))
   (del (surface-finish-side <part> SIDE0 <sf>))
   (add (surface-finish-side <part> SIDE0 FINISH-TURN)))))

(MAKE-THREAD-WITH-LATHE
 (params (<machine> <part> <holding-device> <side>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> LATHE)
   (is-a <toolbit> V-THREAD)
   (shape-of <part> CYLINDRICAL)
   (holding-tool <machine> <toolbit>)
   (~ (has-burrs <part>))
   (is-clean <part>)
   (holding <machine> <holding-device> <part> SIDE0)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (surface-coating-side <part> SIDE0 <*surface-coating>))
   (del (surface-finish-side <part> SIDE0 <sf>))
   (add (surface-finish-side <part> SIDE0 TAPPED)))))

(MAKE-KNURL-WITH-LATHE
 (params (<machine> <part> <holding-device> <side>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> LATHE)
   (is-a <toolbit> KNURL)
   (shape-of <part> CYLINDRICAL)
   (holding-tool <machine> <toolbit>)
   (~ (has-burrs <part>))
   (is-clean <part>)
   (holding <machine> <holding-device> <part> SIDE0)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (surface-coating-side <part> SIDE0 <*surface-coating>))
   (del (surface-finish-side <part> SIDE0 <sf>))
   (add (surface-finish-side <part> SIDE0 KNURLED)))))

(FILE-WITH-LATHE
 (params (<machine> <part> <holding-device> <lathe-file>
          <diameter-new>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> LATHE)
   (is-a <lathe-file> LATHE-FILE)
   (shape-of <part> CYLINDRICAL)
   (size-of <part> DIAMETER <diameter>)
   (finishing-size <diameter> <diameter-new>)
   (~ (has-burrs <part>))
   (is-clean <part>)
   (holding <machine> <holding-device> <part> SIDE0)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (size-of <part> DIAMETER <diameter>))
   (add (size-of <part> DIAMETER <diameter-new>))
   (del (surface-coating-side <part> SIDE0 <*surface-coating>))
   (del (surface-finish-side <part> SIDE0 <sf>))
   (add (surface-finish-side <part> SIDE0 ROUGH-GRIND)))))

(POLISH-WITH-LATHE
 (params (<machine> <part> <holding-device> <cloth>
          <diameter-new>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> LATHE)
   (is-a <cloth> ABRASIVE-CLOTH)
   (material-of-abrasive-cloth <cloth> EMERY)
   (shape-of <part> CYLINDRICAL)
   (~ (has-burrs <part>))
   (is-clean <part>)
   (holding <machine> <holding-device> <part> SIDE0)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (surface-coating-side <part> SIDE0 <*surface-coating>))
   (del (surface-finish-side <part> SIDE0 <*s-q>))
   (add (surface-finish-side <part> SIDE0 POLISHED)))))

;;;************************************************************
; MACHINE: SHAPER

(ROUGH-SHAPE
 (params (<machine> <part> <cutting-tool> <holding-device>
          <side> <dim> <value>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> SHAPER)
   (is-a <cutting-tool> ROUGHING-CUTTING-TOOL)
   (size-of <part> <dim> <value-old>)
   (smaller <value> <value-old>)
```

```
  (~ (same <dim> DIAMETER)) ; this is necessary, otherwise
          ;  the op is used to make parts cylindrical
  (side-up-for-machining <dim> <side>)
  (holding-tool <machine> <cutting-tool>)
  (holding <machine> <holding-device> <part> <side>)))
 (effects (
  (del (is-clean <part>))
  (add (has-burrs <part>))
  (del (surface-coating-side <part> <side> <*surface-coating>))
  (del (surface-finish-side <part> <side> <*s-q>))
  (add (surface-finish-side <part> <side> ROUGH-SHAPED))
  (add (size-of <part> <dim> <value>))
  (del (size-of <part> <dim> <value-old>)))))

(FINISH-SHAPE
 (params (<machine> <part> <cutting-tool> <holding-device>
        <side> <dim> <value>))
  (preconds (and
   (is-a <part> PART)
   (is-a <machine> SHAPER)
   (is-a <cutting-tool> FINISHING-CUTTING-TOOL)
   (size-of <part> <dim> <value-old>)
   (finishing-size <value-old> <value>)
   (~ (same <dim> DIAMETER)) ; this is necessary, otherwise
          ;  the op is used to make parts cylindrical
   (side-up-for-machining <dim> <side>)
   (holding-tool <machine> <cutting-tool>)
   (~ (has-burrs <part>))
   (is-clean <part>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (surface-coating-side <part> <side> <*surface-coating>))
   (del (surface-finish-side <part> <side> <*s-q>))
   (add (surface-finish-side <part> <side> FINISH-SHAPED))
   (add (size-of <part> <dim> <value>))
   (del (size-of <part> <dim> <value-old>)))))

;;;**********************************************************
; MACHINE: PLANER

(ROUGH-SHAPE-WITH-PLANER
 (params (<machine> <part> <cutting-tool> <holding-device>
        <side> <dim> <value>))
  (preconds (and
   (is-a <part> PART)
   (is-a <machine> PLANER)
   (is-a <cutting-tool> ROUGHING-CUTTING-TOOL)
   (size-of <part> <dim> <value-old>)
   (smaller <value> <value-old>)
   (~ (same <dim> DIAMETER)) ; this is necessary, otherwise
          ;  the op is used to make parts cylindrical
   (side-up-for-machining <dim> <side>)
   (holding-tool <machine> <cutting-tool>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (surface-coating-side <part> <side> <*surface-coating>))
   (del (surface-finish-side <part> <side> <*s-q>))
   (add (surface-finish-side <part> <side> ROUGH-PLANED))
   (add (size-of <part> <dim> <value>))
   (del (size-of <part> <dim> <value-old>)))))

(FINISH-SHAPE-WITH-PLANER
 (params (<machine> <part> <cutting-tool> <holding-device>
        <side> <dim> <value>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> PLANER)
   (is-a <cutting-tool> FINISHING-CUTTING-TOOL)
   (size-of <part> <dim> <value-old>)
   (finishing-size <value-old> <value>)
   (~ (same <dim> DIAMETER)) ; this is necessary, otherwise
          ;  the op is used to make parts cylindrical
   (side-up-for-machining <dim> <side>)
   (holding-tool <machine> <cutting-tool>)
   (~ (has-burrs <part>))
   (is-clean <part>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (surface-coating-side <part> <side> <*surface-coating>))
   (del (surface-finish-side <part> <side> <*s-q>))
   (add (surface-finish-side <part> <side> FINISH-PLANED))
   (add (size-of <part> <dim> <value>))
   (del (size-of <part> <dim> <value-old>)))))

;;;**********************************************************
; MACHINE: GRINDER
```

```
(ROUGH-GRIND-WITH-HARD-WHEEL
 (params (<machine> <part> <wheel> <holding-device>
        <side> <dim> <value>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> GRINDER)
   (is-a <wheel> GRINDING-WHEEL)
   (has-fluid <machine> <fluid> <part>)
   (hardness-of-wheel <wheel> HARD)
   (hardness-of <part> SOFT)
   (~ (material-of <part> BRONZE))
   (~ (material-of <part> COPPER))
   (grit-of-wheel <wheel> COARSE-GRIT)
   (size-of <part> <dim> <value-old>)
   (smaller <value> <value-old>)
   (~ (same <dim> DIAMETER)) ; this is necessary, otherwise
          ;  the op is used to make parts cylindrical
   (side-up-for-machining <dim> <side>)
   (holding-tool <machine> <wheel>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (surface-coating-side <part> <side> <*surface-coating>))
   (del (surface-finish-side <part> <side> <*s-q>))
   (add (surface-finish-side <part> <side> ROUGH-GRIND))
   (add (size-of <part> <dim> <value>))
   (del (size-of <part> <dim> <value-old>)))))

(ROUGH-GRIND-WITH-SOFT-WHEEL
 (params (<machine> <part> <wheel> <holding-device>
        <side> <dim> <value>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> GRINDER)
   (is-a <wheel> GRINDING-WHEEL)
   (has-fluid <machine> <fluid> <part>)
   (hardness-of-wheel <wheel> SOFT)
   (hardness-of <part> HARD)
   (grit-of-wheel <wheel> COARSE-GRIT)
   (size-of <part> <dim> <value-old>)
   (smaller <value> <value-old>)
   (~ (same <dim> DIAMETER)) ; this is necessary, otherwise
          ;  the op is used to make parts cylindrical
   (side-up-for-machining <dim> <side>)
   (holding-tool <machine> <wheel>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (surface-coating-side <part> <side> <*surface-coating>))
   (del (surface-finish-side <part> <side> <*s-q>))
   (add (surface-finish-side <part> <side> ROUGH-GRIND))
   (add (size-of <part> <dim> <value>))
   (del (size-of <part> <dim> <value-old>)))))

(FINISH-GRIND-WITH-HARD-WHEEL
 (params (<machine> <part> <wheel> <holding-device>
        <side> <dim> <value>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> GRINDER)
   (is-a <wheel> GRINDING-WHEEL)
   (has-fluid <machine> <fluid> <part>)
   (hardness-of-wheel <wheel> HARD)
   (hardness-of <part> SOFT)
   (~ (material-of <part> BRONZE))
   (~ (material-of <part> COPPER))
   (grit-of-wheel <wheel> FINE-GRIT)
   (size-of <part> <dim> <value-old>)
   (finishing-size <value-old> <value>)
   (~ (same <dim> DIAMETER)) ; this is necessary, otherwise
          ;  the op is used to make parts cylindrical
   (side-up-for-machining <dim> <side>)
   (holding-tool <machine> <wheel>)
   (~ (has-burrs <part>))
   (is-clean <part>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (surface-coating-side <part> <side> <*surface-coating>))
   (del (surface-finish-side <part> <side> <*s-q>))
   (add (surface-finish-side <part> <side> FINISH-GRIND))
   (add (size-of <part> <dim> <value>))
   (del (size-of <part> <dim> <value-old>)))))


(FINISH-GRIND-WITH-SOFT-WHEEL
 (params (<machine> <part> <wheel> <holding-device>
        <side> <dim> <value>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> GRINDER)
   (is-a <wheel> GRINDING-WHEEL)
   (has-fluid <machine> <fluid> <part>)
   (hardness-of-wheel <wheel> SOFT)
   (hardness-of <part> HARD)
   (grit-of-wheel <wheel> FINE-GRIT)
   (size-of <part> <dim> <value-old>)
   (finishing-size <value-old> <value>)
   (~ (same <dim> DIAMETER)) ; this is necessary, otherwise
             ;  the op is used to make parts cylindrical
   (side-up-for-machining <dim> <side>)
   (holding-tool <machine> <wheel>)
   (~ (has-burrs <part>))
   (is-clean <part>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (surface-coating-side <part> <side> <*surface-coating>))
   (del (surface-finish-side <part> <side> <*s-q>))
   (add (surface-finish-side <part> <side> FINISH-GRIND))
   (add (size-of <part> <dim> <value>))
   (del (size-of <part> <dim> <value-old>)))))

;;;************************************************************
; MACHINE: CIRCULAR-SAW

(CUT-WITH-CIRCULAR-COLD-SAW
 (params (<machine> <part> <attachment> <holding-device> <dim>
          <value>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> CIRCULAR-SAW)
   (is-a <attachment> COLD-SAW)
   (size-of <part> <dim> <value-old>)
   (smaller <value> <value-old>)
   (~ (same <dim> DIAMETER)) ; this is necessary, otherwise
             ;  the op is used to make parts cylindrical
   (side-up-for-machining <dim> <side>)
   (holding-tool <machine> <attachment>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (surface-coating-side <part> <side> <*surface-coating>))
   (del (surface-finish-side <part> <side> <*s-q>))
   (add (surface-finish-side <part> <side> FINISH-MILL))
   (del (size-of <part> <dim> <value-old>))
   (add (size-of <part> <dim> <value>)))))

(CUT-WITH-CIRCULAR-FRICTION-SAW
 (params (<machine> <part> <attachment> <holding-device> <dim>
          <value>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> CIRCULAR-SAW)
   (is-a <attachment> FRICTION-SAW)
   (has-fluid <machine> <fluid> <part>)
   (size-of <part> <dim> <value-old>)
   (smaller <value> <value-old>)
   (~ (same <dim> DIAMETER)) ; this is necessary, otherwise
             ;  the op is used to make parts cylindrical
   (side-up-for-machining <dim> <side>)
   (holding-tool <machine> <attachment>)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (surface-coating-side <part> <side> <*surface-coating>))
   (del (surface-finish-side <part> <side> <*s-q>))
   (add (surface-finish-side <part> <side> ROUGH-MILL))
   (del (size-of <part> <dim> <value-old>))
   (add (size-of <part> <dim> <value>)))))

;;;************************************************************
; MACHINE: BAND-SAW

(CUT-WITH-BAND-SAW
 (params (<machine> <part> <attachment> <dim> <value>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> BAND-SAW)
   (is-a <attachment> BAND-FILE)
   (size-of <part> <dim> <value-old>)
   (smaller <value> <value-old>)
   (~ (same <dim> DIAMETER)) ; this is necessary, otherwise
             ;  the op is used to make parts cylindrical
  (side-up-for-machining <dim> <side>)
  (holding-tool <machine> <attachment>)
  (~ (has-burrs <part>))
  (is-clean <part>)
  (on-table <machine> <part>)))
 (effects (
```

```
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (surface-coating-side <part> <side> <*surface-coating>))
   (del (surface-finish-side <part> <side> <*s-q>))
   (add (surface-finish-side <part> <side> SAWCUT))
   (del (size-of <part> <dim> <value-old>))
   (add (size-of <part> <dim> <value>)))))

(POLISH-WITH-BAND-SAW
 (params (<machine> <part> <attachment> <side>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> BAND-SAW)
   (is-a <attachment> SAW-BAND)
   (side-up-for-machining <dim> <side>)
   (holding-tool <machine> <attachment>)
   (~ (has-burrs <part>))
   (is-clean <part>)
   (on-table <machine> <part>)))
 (effects (
   (del (is-clean <part>))
   (add (has-burrs <part>))
   (del (surface-coating-side <part> <side> <*surface-coating>))
   (del (surface-finish-side <part> <side> <*old-sf-cond>))
   (add (surface-finish-side <part> <side> POLISHED)))))


;;;************************************************************
; MACHINE: WELDER

(WELD-CYLINDERS-METAL-ARC
 (params (<machine> <part1> <part2> <part> <electrode>
          <holding-device> <length>))
 (preconds (and
   (is-a <part1> PART)
   (is-a <part2> PART)
   (~ (same <part1> <part2>))
   (is-a <machine> METAL-ARC-WELDER)
   (is-a <electrode> ELECTRODE)
   (material-of <part1> <material1>)
   (material-of <part2> <material2>)
   (shape-of <part1> CYLINDRICAL)
   (shape-of <part2> CYLINDRICAL)
   (~ (exists (<hole>)
         (has-hole <part1> <hole> <*side> <*depth> <*diameter>
                   <*loc-x> <*loc-y>)))
   (~ (exists (<hole>)
         (has-hole <part2> <hole> <*side> <*depth> <*diameter>
                   <*loc-x> <*loc-y>)))
   (size-of <part1> DIAMETER <diameter1>)
   (size-of <part2> DIAMETER <diameter2>)
   (same <diameter1> <diameter2>)
   (size-of <part1> LENGTH <length1>)
   (size-of <part2> LENGTH <length2>)
   (new-size <length1> <length2> <length>)
   (new-part <part> <part1> <part2>)
   (new-material <material> <material1> <material2>)
   (holding-tool <machine> <electrode>)
   (holding <machine> <holding-device> <part2> SIDE3)))
 (effects (
   (del (is-a <part1> PART))
   (del (is-a <part2> PART))
   (add (is-a <part> PART))
   (add (material-of <part> <material>))
   (add (size-of <part> DIAMETER <diameter1>))
   (add (size-of <part> LENGTH <length>))
   (add (surface-finish-side <part> SIDE0 SAWCUT))
   (if (surface-finish-side <part1> SIDE3 <sf31>)
       (add (surface-finish-side <part> SIDE3 <sf31>)))
   (if (surface-finish-side <part2> SIDE6 <sf62>)
       (add (surface-finish-side <part> SIDE6 <sf62>)))
   (del (holding <machine> <holding-device> <part2> SIDE3))
   (add (holding <machine> <holding-device> <part> SIDE3))
   (del (size-of <part1> DIAMETER <diameter>))
   (del (size-of <part1> LENGTH <length1>))
   (del (size-of <part2> DIAMETER <diameter>))
   (del (size-of <part2> LENGTH <length2>))
   (del (material-of <part1> <material1>))
   (del (material-of <part2> <material2>))
   (del (is-clean <part1>))
   (del (is-clean <part2>))
   (del (surface-coating-side <part1> <*sidea> <*surf-coatinga>))
   (del (surface-coating-side <part2> <*sideb> <*surf-coatingb>))
   (del (surface-finish-side <part1> <*sidec> <*sfc>))
   (del (surface-finish-side <part2> <*sided> <*sfd>)))))

(WELD-CYLINDERS-GAS
 (params (<machine> <part1> <part2> <part> <rod>
          <holding-device> <length>))
 (preconds (and
   (is-a <part1> PART)
   (is-a <part2> PART)
   (~ (same <part1> <part2>))
```

```
  (is-a <machine> GAS-WELDER)
  (is-a <rod> WELDING-ROD)
  (is-a <torch> TORCH)
  (material-of <part1> <material1>)
  (material-of <part2> <material2>)
  (same <material1> <material2>)
  (shape-of <part1> CYLINDRICAL)
  (shape-of <part2> CYLINDRICAL)
  (~ (exists (<hole>)
       (has-hole <part1> <hole> <*side> <*depth> <*diameter>
                 <*loc-x> <*loc-y>)))
  (~ (exists (<hole>)
       (has-hole <part2> <hole> <*side> <*depth> <*diameter>
                 <*loc-x> <*loc-y>)))
  (size-of <part1> DIAMETER <diameter1>)
  (size-of <part2> DIAMETER <diameter2>)
  (same <diameter1> <diameter2>)
  (size-of <part1> LENGTB <length1>)
  (size-of <part2> LENGTB <length2>)
  (new-size <length1> <length2> <length>)
  (new-part <part> <part1> <part2>)
  (holding <machine> <holding-device> <part2> SIDE3)))
 (effects (
  (del (is-a <part1> PART))
  (del (is-a <part2> PART))
  (add (is-a <part> PART))
  (add (material-of <part> <material1>))
  (add (size-of <part> DIAMETER <diameter1>))
  (add (size-of <part> LENGTB <length>))
  (add (surface-finish-side <part> SIDE0 SAWCUT))
  (if (surface-finish-side <part1> SIDE3 <sf31>)
      (add (surface-finish-side <part> SIDE3 <sf31>)))
  (if (surface-finish-side <part2> SIDE6 <sf62>)
      (add (surface-finish-side <part> SIDE6 <sf62>)))
  (del (holding <machine> <holding-device> <part2> SIDE3))
  (add (holding <machine> <holding-device> <part> SIDE3))
  (del (size-of <part1> DIAMETER <diameter>))
  (del (size-of <part1> LENGTB <length1>))
  (del (size-of <part2> DIAMETER <diameter>))
  (del (size-of <part2> LENGTB <length2>))
  (del (material-of <part1> <material1>))
  (del (material-of <part2> <material2>))
  (del (is-clean <part1>))
  (del (is-clean <part2>))
  (del (surface-coating-side <part1> <*sidea> <*surf-coatinga>))
  (del (surface-coating-side <part2> <*sideb> <*surf-coatingb>))
  (del (surface-finish-side <part1> <*sidec> <*sfc>))
  (del (surface-finish-side <part2> <*sided> <*sfd>)))))

;;;************************************************************
; METAL-COATING

(METAL-SPRAY-COATING
 (params (<machine> <wire> <part> <side> <another-machine>
          <holding-device>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> ELECTRIC-ARC-SPRAY-GUN)
   (is-a <wire> SPRAYING-METAL-WIRE)
   (~ (material-of <wire> TUNGSTEN))
   (- (material-of <wire> MOLYBDENUM))
   (is-clean <part>)
   (~ (has-burrs <part>))
   (surface-coating-side <part> <side> FUSED-METAL)
   (is-of-type <another-machine> MACHINE)
   (holding <another-machine> <holding-device> <part> <side>)))
 (effects (
   (if (material-of <wire> STAINLESS-STEEL)
       (add (surface-coating-side <part> <side>
   CORROSION-RESISTANT)))
   (if (material-of <wire> ZIRCONIUM-OXIDE)
       (add (surface-coating-side <part> <side> HEAT-RESISTANT)))
   (if (material-of <wire> ALUMINUM-OXIDE)
       (add (surface-coating-side <part> <side> WEAR-RESISTANT)))
   (del (surface-coating-side <part> <side> FUSED-METAL)))))

(METAL-SPRAY-PREPARE
 (params (<machine> <wire> <part> <side> <another-machine>
          <holding-device>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> ELECTRIC-ARC-SPRAY-GUN)
   (is-a <wire> SPRAYING-METAL-WIRE)
   (has-high-melting-point <wire>)
   (is-clean <part>)
   (~ (has-burrs <part>))
   (is-of-type <another-machine> MACHINE)
   (holding <another-machine> <holding-device> <part> <side>)))
 (effects (
   (add (surface-coating-side <part> <side> FUSED-METAL)))))

;;;************************************************************
; OTHER OPERATIONS
```

```
(CLEAN
 (params (<part>))
 (preconds (and
   (is-a <part> PART)
   (is-available-part <part>)))
 (effects (
   (add (is-clean <part>)))))

(REMOVE-BURRS
 (params (<part> <brush>))
 (preconds (and
   (is-a <part> PART)
   (is-a <brush> BRUSE)
   (is-available-part <part>)))
 (effects (
   (del (is-clean <part>))
   (del (has-burrs <part>)))))


;;;************************************************************
;;;************************************************************
; operators for preparing the machines

;;;************************************************************
; tools in machines

(PUT-TOOL-ON-MILLING-MACHINE
 (params (<machine> <attachment>))
 (preconds (and
   (is-a <machine> MILLING-MACHINE)
   (or (is-of-type <attachment> MILLING-CUTTER)
       (is-of-type <attachment> DRILL-BIT))
   (is-available-tool-holder <machine>)
   (is-available-tool <attachment>)))
 (effects (
   (add (holding-tool <machine> <attachment>)))))

(PUT-IN-DRILL-SPINDLE
 (params (<machine> <drill-bit>))
 (preconds (and
   (is-a <machine> DRILL)
   (is-of-type <drill-bit> DRILL-BIT)
   (is-available-tool-holder <machine>)
   (is-available-tool <drill-bit>)))
 (effects (
   (add (holding-tool <machine> <drill-bit>)))))

(PUT-TOOLBIT-IN-LATHE
 (params (<machine> <toolbit>))
 (preconds (and
   (is-a <machine> LATHE)
   (is-of-type <toolbit> LATHE-TOOLBIT)
   (is-available-tool-holder <machine>)
   (is-available-tool <toolbit>)))
 (effects (
   (add (holding-tool <machine> <toolbit>)))))

(PUT-CUTTING-TOOL-IN-SHAPER-OR-PLANER
 (params (<machine> <cutting-tool>))
 (preconds (and
   (or (is-a <machine> SHAPER)
       (is-a <machine> PLANER))
   (is-of-type <cutting-tool> CUTTING-TOOL)
   (is-available-tool-holder <machine>)
   (is-available-tool <cutting-tool>)))
 (effects (
   (add (holding-tool <machine> <cutting-tool>)))))

(PUT-WHEEL-IN-GRINDER
 (params (<machine> <wheel>))
 (preconds (and
   (is-a <machine> GRINDER)
   (is-a <wheel> GRINDING-WHEEL)
   (is-available-tool-holder <machine>)
   (is-available-tool <wheel>)))
 (effects (
   (add (holding-tool <machine> <wheel>)))))

(PUT-CIRCULAR-SAW-ATTACHMENT-IN-CIRCULAR-SAW
 (params (<machine> <attachment>))
 (preconds (and
   (is-a <machine> CIRCULAR-SAW)
   (is-of-type <attachment> CIRCULAR-SAW-ATTACHMENT)
   (is-available-tool-holder <machine>)
   (is-available-tool <attachment>)))
 (effects (
   (add (holding-tool <machine> <attachment>)))))

(PUT-BAND-SAW-ATTACHMENT-IN-BAND-SAW
 (params (<machine> <attachment>))
 (preconds (and
   (is-a <machine> BAND-SAW)
```

```
    (is-of-type <attachment> BAND-SAW-ATTACHMENT)
    (is-available-tool-holder <machine>)
    (is-available-tool <attachment>)))
  (effects (
    (add (holding-tool <machine> <attachment>)))))

 (PUT-ELECTRODE-IN-WELDER
  (params (<machine> <electrode>))
  (preconds (and
    (is-a <machine> METAL-ARC-WELDER)
    (is-a <electrode> ELECTRODE)
    (is-available-tool-holder <machine>)
    (is-available-tool <electrode>)))
  (effects (
    (add (holding-tool <machine> <electrode>)))))

 (REMOVE-TOOL-FROM-MACHINE
  (params (<machine> <tool>))
  (preconds (and
    (is-of-type <machine> MACHINE)
    (is-of-type <tool> MACHINE-TOOL)
    (holding-tool <machine> <tool>)))
  (effects (
    (del (holding-tool <machine> <tool>)))))

;;;************************************************************
; holding devices in machines

 (PUT-HOLDING-DEVICE-IN-MILLING-MACHINE
  (params (<machine> <holding-device>))
  (preconds (and
    (is-a <machine> MILLING-MACHINE)
    (or
         (is-a <holding-device> 4-JAW-CHUCK)
         (is-a <holding-device> V-BLOCK)
         (is-a <holding-device> VISE)
         (is-a <holding-device> COLLET-CHUCK)
         (is-a <holding-device> TOE-CLAMP))
    (is-available-table <machine> <holding-device>)
    (is-available-holding-device <holding-device>)))
  (effects (
    (add (has-device <machine> <holding-device>)))))

 (PUT-HOLDING-DEVICE-IN-DRILL
  (params (<machine> <holding-device>))
  (preconds (and
    (is-a <machine> DRILL)
    (or
         (is-a <holding-device> 4-JAW-CHUCK)
         (is-a <holding-device> V-BLOCK)
         (is-a <holding-device> VISE)
         (is-a <holding-device> TOE-CLAMP))
    (is-available-table <machine> <holding-device>)
    (is-available-holding-device <holding-device>)))
  (effects (
    (add (has-device <machine> <holding-device>)))))

 (PUT-HOLDING-DEVICE-IN-LATHE
  (params (<machine> <holding-device>))
  (preconds (and
    (is-a <machine> LATHE)
    (or (is-a <holding-device> CENTERS)
        (is-a <holding-device> 4-JAW-CHUCK)
        (is-a <holding-device> COLLET-CHUCK))
    (is-available-table <machine> <holding-device>)
    (is-available-holding-device <holding-device>)))
  (effects (
    (add (has-device <machine> <holding-device>)))))

 (PUT-HOLDING-DEVICE-IN-SHAPER
  (params (<machine> <holding-device>))
  (preconds (and
    (is-a <machine> SHAPER)
    (is-a <holding-device> VISE)
    (is-available-table <machine> <holding-device>)
    (is-available-holding-device <holding-device>)))
  (effects (
    (add (has-device <machine> <holding-device>)))))

 (PUT-HOLDING-DEVICE-IN-PLANER
  (params (<machine> <holding-device>))
  (preconds (and
    (is-a <machine> PLANER)
    (is-a <holding-device> TOE-CLAMP)
    (is-available-table <machine> <holding-device>)
    (is-available-holding-device <holding-device>)))
  (effects (
    (add (has-device <machine> <holding-device>)))))

 (PUT-HOLDING-DEVICE-IN-GRINDER
  (params (<machine> <holding-device>))
  (preconds (and
    (is-a <machine> GRINDER)
```

```
    (or (is-a <holding-device> MAGNETIC-CHUCK)
        (is-a <holding-device> V-BLOCK)
        (is-a <holding-device> VISE))
    (is-available-table <machine> <holding-device>)
    (is-available-holding-device <holding-device>)))
  (effects (
    (add (has-device <machine> <holding-device>)))))

 (PUT-HOLDING-DEVICE-IN-CIRCULAR-SAW
  (params (<machine> <holding-device>))
  (preconds (and
    (is-a <machine> CIRCULAR-SAW)
    (or (is-a <holding-device> VISE)
        (is-a <holding-device> V-BLOCK))
    (is-available-table <machine> <holding-device>)
    (is-available-holding-device <holding-device>)))
  (effects (
    (add (has-device <machine> <holding-device>)))))

 (PUT-HOLDING-DEVICE-IN-WELDER
  (params (<machine> <holding-device>))
  (preconds (and
    (is-of-type <machine> WELDER)
    (or (is-a <holding-device> VISE)
          (is-a <holding-device> TOE-CLAMP))
    (is-available-table <machine> <holding-device>)
    (is-available-holding-device <holding-device>)))
  (effects (
    (add (has-device <machine> <holding-device>)))))


 (REMOVE-HOLDING-DEVICE-FROM-MACHINE
  (params (<machine> <holding-device>))
  (preconds (and
    (is-of-type <machine> MACHINE)
    (is-of-type <holding-device> HOLDING-DEVICE)
    (has-device <machine> <holding-device>)
    (is-empty-holding-device <holding-device> <machine>)))
  (effects (
    (del (has-device <machine> <holding-device>)))))

;;;************************************************************
; cutting fluid in machines

 (ADD-SOLUBLE-OIL
  (params (<machine> <fluid>))
  (preconds (and
    (is-of-type <machine> MACHINE)
    (is-a <part> PART)
    (or (material-of <part> STEEL)
        (material-of <part> ALUMINUM))
    (is-a <fluid> SOLUBLE-OIL)))
  (effects (
    (add (has-fluid <machine> <fluid> <part>)))))

 (ADD-MINERAL-OIL
  (params (<machine> <fluid>))
  (preconds (and
    (is-of-type <machine> MACHINE)
    (is-a <part> PART)
    (is-a <fluid> MINERAL-OIL)
    (material-of <part> IRON)))
  (effects (
    (add (has-fluid <machine> <fluid> <part>)))))

 (ADD-ANY-CUTTING-FLUID
  (params (<machine> <fluid>))
  (preconds (and
    (is-of-type <machine> MACHINE)
    (is-a <part> PART)
    (or (material-of <part> BRASS)
        (material-of <part> BRONZE)
        (material-of <part> COPPER))
    (is-of-type <fluid> CUTTING-FLUID)))
  (effects (
    (add (has-fluid <machine> <fluid> <part>)))))


;;;************************************************************
;;;************************************************************
; operators for holding parts with a device in a machine

 (PUT-ON-MACHINE-TABLE
  (params (<machine> <part>))
  (preconds (and
    (is-a <part> PART)
    (is-of-type <machine> MACHINE)
    (~ (is-a <machine> SHAPER))
    (is-available-part <part>)
    (is-available-machine <machine>)))
  (effects (
    (del (on-table <another-machine> <part>))
    (add (on-table <machine> <part>)))))
```

```
(PUT-ON-SHAPER-TABLE
 (params (<machine> <part>))
 (preconds (and
   (is-a <part> PART)
   (is-a <machine> SHAPER)
   (size-of-machine <machine> <shaper-size>)
   (size-of <part> LENGTH <part-size>)
   (smaller <part-size> <shaper-size>)
   (is-available-part <part>)
   (is-available-machine <machine>)))
 (effects (
   (del (on-table <another-machine> <part>))
   (add (on-table <machine> <part>)))))


(HOLD-WITH-V-BLOCK
 (params (<machine> <holding-device> <part> <side>))
 (preconds
   (and
   (is-of-type <machine> MACHINE)
   (is-a <part> PART)
   (is-a <holding-device> V-BLOCK)
   (has-device <machine> <holding-device>)
   (~ (has-burrs <part>))
   (is-clean <part>)
   (on-table <machine> <part>)
   (shape-of <part> CYLINDRICAL)
   (same <side> SIDE0)
   (is-empty-holding-device <holding-device> <machine>)
   (is-available-part <part>)))
 (effects (
   (del (on-table <machine> <part>))
   (add (holding-weakly <machine> <holding-device>
                        <part> <side>)))))

(HOLD-WITH-VISE
 (params (<machine> <holding-device> <part> <side>))
 (preconds (and
   (is-of-type <machine> MACHINE)
   (is-a <part> PART)
   (is-a <holding-device> VISE)
   (has-device <machine> <holding-device>)
   (~ (has-burrs <part>))
   (is-clean <part>)
   (on-table <machine> <part>)
   (is-empty-holding-device <holding-device> <machine>)
   (is-available-part <part>)))
 (effects (
   (del (on-table <machine> <part>))
   (if (shape-of <part> CYLINDRICAL)
   (add (holding-weakly <machine> <holding-device>
        <part> <side>)))
   (if (shape-of <part> RECTANGULAR)
   (add (holding <machine> <holding-device> <part> <side>))))))

(HOLD-WITH-TOE-CLAMP
 (params (<machine> <holding-device> <part> <side>))
 (preconds (and
   (is-of-type <machine> MACHINE)
   (is-a <part> PART)
   (is-a <holding-device> TOE-CLAMP)
   (has-device <machine> <holding-device>)
   (~ (has-burrs <part>))
   (is-clean <part>)
   (or (shape-of <part> RECTANGULAR)
       (same <side> SIDE3)   ; can hold cylindrical parts
       (same <side> SIDE6))  ; only with toe clamp for sides 3 &6
   (on-table <machine> <part>)
   (is-empty-holding-device <holding-device> <machine>)
   (is-available-part <part>)))
 (effects (
   (del (on-table <machine> <part>))
   (add (holding <machine> <holding-device> <part> <side>)))))

(SECURE-WITH-TOE-CLAMP
 (params (<machine> <holding-device> <part> <side>))
 (preconds (and
   (is-of-type <machine> MACHINE)
   (is-a <part> PART)
   (is-a <holding-device> TOE-CLAMP)
   (has-device <machine> <holding-device>)
   (~ (has-burrs <part>))
   (is-clean <part>)
   (shape-of <part> CYLINDRICAL)
   (holding-weakly <machine> <another-holding-device>
   <part> <side>)
   (is-empty-holding-device <holding-device> <machine>)))
 (effects (
   (del (on-table <machine> <part>))
   (add (holding <machine> <holding-device> <part> <side>)))))
```

```
(HOLD-WITH-CENTERS
 (params (<machine> <holding-device> <part> <side>))
 (preconds (and
   (is-of-type <machine> MACHINE)
   (is-a <part> PART)
   (is-a <holding-device> CENTERS)
   (has-device <machine> <holding-device>)
   (has-center-holes <part>)
   (- (has-burrs <part>))
   (is-clean <part>)
   (on-table <machine> <part>)
   (shape-of <part> CYLINDRICAL)
   (is-empty-holding-device <holding-device> <machine>)
   (is-available-part <part>)))
 (effects (
   (del (on-table <machine> <part>))
   (add (holding <machine> <holding-device> <part> <side>)))))

(HOLD-WITH-4-JAW-CHUCK
 (params (<machine> <holding-device> <part> <side>))
 (preconds (and
   (is-of-type <machine> MACHINE)
   (is-a <part> PART)
   (is-a <holding-device> 4-JAW-CHUCK)
   (has-device <machine> <holding-device>)
   (~ (has-burrs <part>))
   (is-clean <part>)
   (on-table <machine> <part>)
   (is-empty-holding-device <holding-device> <machine>)
   (is-available-part <part>)))
 (effects (
   (del (on-table <machine> <part>))
   (add (holding <machine> <holding-device> <part> <side>)))))

(HOLD-WITH-COLLET-CHUCK
 (params (<machine> <holding-device> <part> <side>))
 (preconds (and
   (is-of-type <machine> MACHINE)
   (is-a <part> PART)
   (is-a <holding-device> COLLET-CHUCK)
   (has-device <machine> <holding-device>)
   (- (has-burrs <part>))
   (is-clean <part>)
   (on-table <machine> <part>)
   (shape-of <part> CYLINDRICAL)
   (is-empty-holding-device <holding-device> <machine>)
   (is-available-part <part>)))
 (effects (
   (del (on-table <machine> <part>))
   (add (holding <machine> <holding-device> <part> <side>)))))

(HOLD-WITH-MAGNETIC-CHUCK
 (params (<machine> <holding-device> <part> <side>))
 (preconds (and
   (is-of-type <machine> MACHINE)
   (is-a <part> PART)
   (is-a <holding-device> MAGNETIC-CHUCK)
   (has-device <machine> <holding-device>)
   (~ (has-burrs <part>))
   (is-clean <part>)
   (on-table <machine> <part>)
   (is-empty-holding-device <holding-device> <machine>)
   (is-available-part <part>)))
 (effects (
   (del (on-table <machine> <part>))
   (add (holding <machine> <holding-device> <part> <side>)))))

(RELEASE-FROM-HOLDING-DEVICE
 (params (<machine> <holding-device> <part> <side>))
 (preconds (and
   (is-of-type <machine> MACHINE)
   (is-a <part> PART)
   (is-of-type <holding-device> HOLDING-DEVICE)
   (holding <machine> <holding-device> <part> <side>)))
 (effects (
   (del (holding <machine> <holding-device> <part> <side>))
   (add (on-table <machine> <part>)))))

(RELEASE-FROM-HOLDING-DEVICE-WEAK
 (params (<machine> <holding-device> <part> <side>))
 (preconds (and
   (is-of-type <machine> MACHINE)
   (is-a <part> PART)
   (is-of-type <holding-device> HOLDING-DEVICE)
   (holding-weakly <machine> <holding-device> <part> <side>)))
 (effects (
   (del (holding-weakly <machine> <holding-device>
                        <part> <side>))
   (add (on-table <machine> <part>)))))

))

;;;************************************************************
```

```
;;;************************************************************
;;;************************************************************

(setq *INFERENCE-RULES* '(


(HAS-CENTER-HOLES
 (params (<part> <x2> <y2>))
 (preconds (and
   (is-a <part> PART)
   (or (and
          (shape-of <part> RECTANGULAR)
          (size-of <part> WIDTH <x>)
          (size-of <part> HEIGHT <y>))
     (and
          (shape-of <part> CYLINDRICAL)
          (size-of <part> DIAMETER <x>)
          (size-of <part> DIAMETER <y>)))
   (half-of <x> <x2>)
   (half-of <y> <y2>)
   (has-center-hole <part> CENTER-HOLE-SIDE3 SIDE3 <x2> <y2>)
   (is-countersinked <part> CENTER-HOLE-SIDE3 SIDE3 1/8 1/16
                     <x2> <y2> 60)
   (has-center-hole <part> CENTER-HOLE-SIDE6 SIDE6 <x2> <y2>)
   (is-countersinked <part> CENTER-HOLE-SIDE6 SIDE6 1/8 1/16
                     <x2> <y2> 60)))
 (effects (
   (add (has-center-holes <part>)))))

;;;************************************************************

(SIDE-UP-FOR-MACHINING-LENGTH
 (params (<side>))
 (preconds (and
   (same <dim> LENGTH)
   (or (same <side> SIDE3)
     (same <side> SIDE6))))
 (effects (
   (add (side-up-for-machining <dim> <side>)))))

(SIDE-UP-FOR-MACHINING-WIDTH
 (params (<side>))
 (preconds (and
   (same <dim> WIDTH)
   (or (same <side> SIDE2)
     (same <side> SIDE5))))
 (effects (
   (add (side-up-for-machining <dim> <side>)))))

(SIDE-UP-FOR-MACHINING-HEIGHT
 (params (<side>))
 (preconds (and
   (same <dim> HEIGHT)
   (or (same <side> SIDE1)
     (same <side> SIDE4))))
 (effects (
   (add (side-up-for-machining <dim> <side>)))))

(SIDE-UP-FOR-MACHINING-DIAMETER
 (params (<side>))
 (preconds (and
   (same <dim> DIAMETER)
   (or (and
          (shape-of <part> RECTANGULAR)
          (same <side> SIDE1))
     (and
          (shape-of <part> CYLINDRICAL)
          (same <side> SIDE0)))))
 (effects (
   (add (side-up-for-machining <dim> <side>)))))

;;;************************************************************
; inference rules for availability

(MACHINE-AVAILABLE
 (params (<machine>))
 (preconds (and
   (is-of-type <machine> MACHINE)
   (~ (exists (<other-part>)
          (on-table <machine> <other-part>)))))
 (effects (
   (add (is-available-machine <machine>)))))

(TOOL-HOLDER-AVAILABLE
 (params (<machine>))
 (preconds (and
   (is-of-type <machine> MACHINE)
   (~ (exists (<tool>)
          (holding-tool <machine> <tool>)))))
 (effects (
   (add (is-available-tool-holder <machine>)))))

(TOOL-AVAILABLE
 (params (<tool>))
 (preconds (and
   (is-of-type <tool> MACHINE-TOOL)
   (~ (exists (<machine>)
          (holding-tool <machine> <tool>)))))
 (effects ( (add (is-available-tool <tool>)))))

(TABLE-AVAILABLE
 (params (<machine>))
 (preconds (and
   (is-of-type <machine> MACHINE)
   (is-of-type <holding-device> HOLDING-DEVICE)
   (or
     (~ (exists (<another-holding-device>)
          (has-device <machine> <another-holding-device>)))
     (is-a <holding-device> TOE-CLAMP))))
 (effects (
   (add (is-available-table <machine> <holding-device>)))))

(HOLDING-DEVICE-AVAILABLE
 (params (<machine> <holding-device>))
 (preconds (and
   (is-of-type <holding-device> HOLDING-DEVICE)
   (~ (exists (<machine>)
          (has-device <machine> <holding-device>)))))
 (effects (
   (add (is-available-holding-device <holding-device>)))))

(PART-AVAILABLE
 (params (<part>))
 (preconds (and
   (is-a <part> PART)
   (~ (exists (<machine>)
          (holding-weakly <machine> <*holding-device>
                          <part> <*side>)))
   (~ (exists (<machine>)
          (holding <machine> <*another-holding-device>
                   <part> <*side>)))))
 (effects (
   (add (is-available-part <part>)))))

(HOLDING-DEVICE-EMPTY
 (params (<machine> <holding-device>))
 (preconds (and
   (is-of-type <machine> MACHINE)
   (is-of-type <holding-device> HOLDING-DEVICE)
   (~ (exists (<part>)
          (holding-weakly <machine> <holding-device>
                          <part> <side>)))
   (~ (exists (<another-part>)
          (holding <machine> <holding-device>
                   <another-part> <side>)))))
 (effects (
   (add (is-empty-holding-device <holding-device> <machine>)))))

;;;************************************************************
; inference rules for shape

(IS-RECTANGULAR
 (params (<part>))
 (preconds (and
   (is-a <part> PART)
   (size-of <part> LENGTH <l>)
   (size-of <part> WIDTH <w>)
   (size-of <part> HEIGHT <h>)))
 (effects (
   (add (shape-of <part> RECTANGULAR)))))

(IS-CYLINDRICAL
 (params (<part>))
 (preconds (and
   (is-a <part> PART)
   (size-of <part> LENGTH <l>)
   (size-of <part> DIAMETER <d>)))
 (effects (
   (add (shape-of <part> CYLINDRICAL)))))

(ARE-SIDES-OF-RECTANGULAR-PART
 (params (<part>))
 (preconds (and
   (is-a <part> PART)
   (shape-of <part> RECTANGULAR)))
 (effects (
   (add (side-of <part> SIDE1))
   (add (side-of <part> SIDE2))
   (add (side-of <part> SIDE3))
   (add (side-of <part> SIDE4))
   (add (side-of <part> SIDE5))
   (add (side-of <part> SIDE6)))))


(ARE-SIDES-OF-CYLINDRICAL-PART
 (params (<part>))
 (preconds (and
   (is-a <part> PART)
   (shape-of <part> CYLINDRICAL)))
 (effects (
   (add (side-of <part> SIDE0))
   (add (side-of <part> SIDE3))
   (add (side-of <part> SIDE6)))))


;;;************************************************************
; inference rules for surface finish

(IS-MACHINED-SURFACE-QUALITY
 (params (<part> <side>))
 (preconds (and
   (is-a <part> PART)
   (or
      (surface-finish-side <part> <side> ROUGH-MILL)
      (surface-finish-side <part> <side> ROUGH-TURN)
      (surface-finish-side <part> <side> ROUGH-SHAPED)
      (surface-finish-side <part> <side> ROUGH-PLANED)
      (surface-finish-side <part> <side> FINISH-PLANED)
      (surface-finish-side <part> <side> COLD-ROLLED)
      (surface-finish-side <part> <side> FINISH-MILL)
      (surface-finish-side <part> <side> FINISH-TURN)
      (surface-finish-quality-side <part> <side> GROUND))))
 (effects (
   (add (surface-finish-quality-side <part> <side> MACHINED)))))

(IS-GROUND-SURFACE-QUALITY
 (params (<part> <side>))
 (preconds (and
   (is-a <part> PART)
   (or (surface-finish-side <part> <side> ROUGH-GRIND)
      (surface-finish-side <part> <side> FINISH-GRIND))))
 (effects (
   (add (surface-finish-quality-side <part> <side> GROUND)))))


(HAS-SURFACE-FINISH-RECTANGULAR-PART
 (params (<part>))
 (preconds (and
   (is-a <part> PART)
   (shape-of <part> RECTANGULAR)
   (surface-finish-side <part> SIDE1 <surface-finish>)
   (surface-finish-side <part> SIDE2 <surface-finish>)
   (surface-finish-side <part> SIDE3 <surface-finish>)
   (surface-finish-side <part> SIDE4 <surface-finish>)
   (surface-finish-side <part> SIDE5 <surface-finish>)
   (surface-finish-side <part> SIDE6 <surface-finish>)))
 (effects (
   (add (surface-finish <part> <surface-finish>)))))

(HAS-SURFACE-FINISH-CYLINDRICAL-PART
 (params (<part>))
 (preconds (and
   (is-a <part> PART)
   (shape-of <part> CYLINDRICAL)
   (surface-finish-side <part> SIDE0 <surface-finish>)
   (surface-finish-side <part> SIDE3 <surface-finish>)
   (surface-finish-side <part> SIDE6 <surface-finish>)))
 (effects (
   (add (surface-finish <part> <surface-finish>)))))


(HAVE-SURFACE-FINISH-RECTANGULAR-PART-SIDES
 (params (<part>))
 (preconds (and
   (is-a <part> PART)
   (shape-of <part> RECTANGULAR)
   (surface-finish <part> <surface-finish>)))
 (effects (
   (add (surface-finish-side <part> SIDE1 <surface-finish>))
   (add (surface-finish-side <part> SIDE2 <surface-finish>))
   (add (surface-finish-side <part> SIDE3 <surface-finish>))
   (add (surface-finish-side <part> SIDE4 <surface-finish>))
   (add (surface-finish-side <part> SIDE5 <surface-finish>))
   (add (surface-finish-side <part> SIDE6 <surface-finish>)))))

(HAVE-SURFACE-FINISH-CYLINDRICAL-PART-SIDES
 (params (<part>))
 (preconds (and
   (is-a <part> PART)
   (shape-of <part> CYLINDRICAL)
   (surface-finish <part> <surface-finish>)))
 (effects (
   (add (surface-finish-side <part> SIDE0 <surface-finish>))
   (add (surface-finish-side <part> SIDE3 <surface-finish>))
   (add (surface-finish-side <part> SIDE6 <surface-finish>)))))

;;;************************************************************
;  inference rules for surface-coating
```

```
(HAS-SURFACE-COATING-RECTANGULAR-PART
 (params (<part>))
 (preconds (and
   (is-a <part> PART)
   (shape-of <part> RECTANGULAR)
   (surface-coating-side <part> SIDE1 <surface-coating>)
   (surface-coating-side <part> SIDE2 <surface-coating>)
   (surface-coating-side <part> SIDE3 <surface-coating>)
   (surface-coating-side <part> SIDE4 <surface-coating>)
   (surface-coating-side <part> SIDE5 <surface-coating>)
   (surface-coating-side <part> SIDE6 <surface-coating>)))
 (effects (
   (add (surface-coating <part> <surface-coating>)))))

(HAS-SURFACE-COATING-CYLINDRICAL-PART
 (params (<part>))
 (preconds (and
   (is-a <part> PART)
   (shape-of <part> CYLINDRICAL)
   (surface-coating-side <part> SIDE0 <surface-coating>)
   (surface-coating-side <part> SIDE3 <surface-coating>)
   (surface-coating-side <part> SIDE6 <surface-coating>)))
 (effects (
   (add (surface-coating <part> <surface-coating>)))))


(HAVE-SURFACE-COATING-RECTANGULAR-PART-SIDES
 (params (<part>))
 (preconds (and
   (is-a <part> PART)
   (shape-of <part> RECTANGULAR)
   (surface-coating <part> <surface-coating>)))
 (effects (
   (add (surface-coating-side <part> SIDE1 <surface-coating>))
   (add (surface-coating-side <part> SIDE2 <surface-coating>))
   (add (surface-coating-side <part> SIDE3 <surface-coating>))
   (add (surface-coating-side <part> SIDE4 <surface-coating>))
   (add (surface-coating-side <part> SIDE5 <surface-coating>))
   (add (surface-coating-side <part> SIDE6 <surface-coating>)))))

(HAVE-SURFACE-COATING-CYLINDRICAL-PART-SIDES
 (params (<part>))
 (preconds (and
   (is-a <part> PART)
   (shape-of <part> CYLINDRICAL)
   (surface-coating <part> <surface-coating>)))
 (effects (
   (add (surface-coating-side <part> SIDE0 <surface-coating>))
   (add (surface-coating-side <part> SIDE3 <surface-coating>))
   (add (surface-coating-side <part> SIDE6 <surface-coating>)))))

;;;************************************************************

(MATERIAL-FERROUS
 (params (<part>))
 (preconds (and
   (is-a <part> PART)
   (or
      (material-of <part> STEEL)
      (material-of <part> IRON))))
 (effects (
   (add (alloy-of <part> FERROUS)))))

(MATERIAL-NON-FERROUS
 (params (<part>))
 (preconds (and
   (is-a <part> PART)
   (or
      (material-of <part> BRASS)
      (material-of <part> COPPER)
      (material-of <part> BRONZE))))
 (effects (
   (add (alloy-of <part> NON-FERROUS)))))

(HARDNESS-OF-MATERIAL-SOFT
 (params (<part>))
 (preconds (and
   (is-a <part> PART)
   (or
      (material-of <part> ALUMINUM)
      (alloy-of <part> NON-FERROUS))))
 (effects (
   (add (hardness-of <part> SOFT)))))

(HARDNESS-OF-MATERIAL-HARD
 (params (<part>))
 (preconds (and
   (is-a <part> PART)
   (alloy-of <part> FERROUS)))
 (effects (
   (add (hardness-of <part> HARD)))))
 (HIGH-MELTING-POINT
  (params (<wire>))
  (preconds (and
    (is-a <wire> SPRAYING-METAL-WIRE)
    (or
       (material-of <wire> TUNGSTEN)
       (material-of <wire> MOLYBDENUM))))
  (effects (
    (add (has-high-melting-point <wire>)))))


;;;************************************************************
; inference rules for types

 (IS-MACHINE
  (params (<machine>))
  (preconds
    (or
        (is-a <machine> DRILL)
        (is-a <machine> LATHE)
        (is-a <machine> SHAPER)
        (is-a <machine> PLANER)
        (is-a <machine> GRINDER)
        (is-a <machine> BAND-SAW)
        (is-a <machine> CIRCULAR-SAW)
        (is-a <machine> MILLING-MACHINE)
        (is-of-type <machine> WELDER)))
  (effects (
    (add (is-of-type <machine> MACHINE)))))

 (IS-WELDER
  (params (<machine>))
  (preconds
    (or
        (is-a <machine> METAL-ARC-WELDER)
        (is-a <machine> GAS-WELDER)))
  (effects (
    (add (is-of-type <machine> WELDER)))))

 (IS-TOOL
  (params (<tool>))
  (preconds
    (or
        (is-of-type <tool> MACHINE-TOOL)
        (is-of-type <tool> OPERATOR-TOOL)))
  (effects (
    (add (is-of-type <tool> TOOL)))))

 (IS-MACHINE-TOOL
  (params (<attachment>))
  (preconds
     (or
         (is-of-type <attachment> DRILL-BIT)
         (is-of-type <attachment> LATHE-TOOLBIT)
         (is-of-type <attachment> CUTTING-TOOL)
         (is-a <attachment> GRINDING-WHEEL)
         (is-of-type <attachment> BAND-SAW-ATTACHMENT)
         (is-of-type <attachment> CIRCULAR-SAW-ATTACHMENT)
         (is-of-type <attachment> MILLING-CUTTER)
         (is-a <attachment> ELECTRODE)))
  (effects (
    (add (is-of-type <attachment> MACHINE-TOOL)))))

 (IS-DRILL-BIT
  (params (<drill-bit>))
  (preconds
    (or
        (is-a <drill-bit> SPOT-DRILL)
        (is-a <drill-bit> CENTER-DRILL)
        (is-a <drill-bit> TWIST-DRILL)
        (is-a <drill-bit> STRAIGHT-FLUTED-DRILL)
        (is-a <drill-bit> HIGH-HELIX-DRILL)
        (is-a <drill-bit> OIL-HOLE-DRILL)
        (is-a <drill-bit> GUN-DRILL)
        (is-a <drill-bit> CORE-DRILL)
        (is-a <drill-bit> TAP)
        (is-a <drill-bit> COUNTERSINK)
        (is-a <drill-bit> COUNTERBORE)
        (is-a <drill-bit> REAMER)))
  (effects (
    (add (is-of-type <drill-bit> DRILL-BIT)))))

 (IS-LATHE-TOOLBIT
  (params (<toolbit>))
  (preconds
    (or
        (is-a <toolbit> ROUGH-TOOLBIT)
        (is-a <toolbit> FINISH-TOOLBIT)
        (is-a <toolbit> V-THREAD)
        (is-a <toolbit> KNURL)))
  (effects (
    (add (is-of-type <toolbit> LATHE-TOOLBIT)))))
```

```
 (IS-CUTTING-TOOL
  (params (<cutting-tool>))
  (preconds
    (or
        (is-a <cutting-tool> ROUGHING-CUTTING-TOOL)
        (is-a <cutting-tool> FINISHING-CUTTING-TOOL)))
  (effects (
    (add (is-of-type <cutting-tool> CUTTING-TOOL)))))

 (IS-CIRCULAR-SAW-ATTACHMENT
  (params (<attachment>))
  (preconds
    (or
        (is-a <attachment> COLD-SAW)
        (is-a <attachment> FRICTION-SAW)))
  (effects (
    (add (is-of-type <attachment> CIRCULAR-SAW-ATTACHMENT)))))


 (IS-BAND-SAW-ATTACHMENT
  (params (<attachment>))
  (preconds
    (or
        (is-a <attachment> SAW-BAND)
        (is-a <attachment> BAND-FILE)))
  (effects (
    (add (is-of-type <attachment> BAND-SAW-ATTACHMENT)))))

 (IS-MILLING-CUTTEP
  (params (<milling-cutter>))
  (preconds
    (or
        (is-a <milling-cutter> PLAIN-MILL)
        (is-a <milling-cutter> END-MILL)))
  (effects (
    (add (is-of-type <milling-cutter> MILLING-CUTTER)))))

 (IS-OPERATOR-TOOL
  (params (<tool>))
  (preconds
    (or
        (is-a <tool> LATHE-FILE)
        (is-a <tool> ABRASIVE-CLOTH)
        (is-a <tool> TORCH)
        (is-a <tool> WELDING-ROD)
        (is-a <tool> SPRAYING-METAL-WIRE)
        (is-a <tool> BRUSH)))
  (effects (
    (add (is-of-type <tool> OPERATOR-TOOL)))))

 (IS-CUTTING-FLUID
  (params (<cutting-fluid>))
  (preconds
    (or
        (is-a <cutting-fluid> SOLUBLE-OIL)
        (is-a <cutting-fluid> MINERAL-OIL)))
  (effects (
    (add (is-of-type <cutting-fluid> CUTTING-FLUID)))))

 (IS-HOLDING-DEVICE
  (params (<holding-device>))
  (preconds
    (or
        (is-a <holding-device> V-BLOCK)
        (is-a <holding-device> VISE)
        (is-a <holding-device> TOE-CLAMP)
        (is-a <holding-device> CENTERS)
        (is-a <holding-device> 4-JAW-CHUCK)
        (is-a <holding-device> COLLET-CHUCK)
        (is-a <holding-device> MAGNETIC-CHUCK)))
  (effects (
    (add (is-of-type <holding-device> HOLDING-DEVICE)))))


))

;;;************************************************************
;;;************************************************************
;;;************************************************************
; Functions


(defun same (x y)
   (cond ((is-variable x)
           (return-binding x y))
          ((is-variable y)
           (return-binding y x))
          (t
           (equal x y))))

(defun half-of (x y)
   (cond ((is-variable x)
          'no-match-attempted)
        ((is-variable y)
         (return-binding y (/ x 2)))
        ((= (/ x 2) y) t )))

(defun smaller (x y)
   (cond ((is-variable x)
          (if (> (- y .5) 0)
              (return-binding x (- y .5))))
         ((is-variable y)
          (return-binding y (+ x .5)))
         ((< x y) t )))

(defun smaller-than-2in (x y)
   (cond ((is-variable x)
          'no-match-attempted)
         ((is-variable y)
          'no-match-attempted)
         (t
          (<= (- x y) 2))))


; Function used for finish operations.
; They make the size of the part be .002 or .003 smaller.
;
(defun finishing-size (x y)
   (cond ((and (is-variable x)
               (is-variable y))
          'no-match-attempted)
         ((is-variable x)
          (return-binding x (+ y 0.002)))
         ((is-variable y)
          (if (> (- x 0.002) 0)
              (return-binding y (- x 0.002))))
         (t
          (<= (abs (- x y)) 0.003))))

; Functions for generating new values when two parts are
;  welded together.

(defun new-size (d1 d2 d)
   (cond ((is-variable d1)
          'no-match-attempted)
         ((is-variable d2)
          'no-match-attempted)
         ((is-variable d)
          (return-binding d (+ d1 d2)))
         (t
          (= d (+ d1 d2)))))

(defun new-part (part part1 part2)
   (cond ((is-variable part)
          (return-binding part (new-name part1 part2)))
         (t)))

(defun new-material (material material1 material2)
 (if (is-variable material)
   (cond ((same material1 material2)
          (return-binding material material1))
         (t
          (return-binding material
  (new-name material1 material2))))
   t))

(defun new-name (name1 name2)
  (intern (concatenate 'string
                       (symbol-name name1)
                       "-"
                       (symbol-name name2))))


;;;************************************************************

(defun return-binding (var val)
  (list (list (list var val))))


;;;************************************************************
;;;************************************************************
;;;************************************************************
;; Control rules


;;;************************************************************

(setq *SCR-NODE-SELECT-RULES* nil)

(setq *SCR-GOAL-SELECT-RULES*
      '((SELECT-FIRST-GOAL
           (lhs (and (current-node <node>)
                     (not-top-level-node <node>)
                     (primary-candidate-goal <node> <goal>)))
           (rhs (select goal <goal>)))
```

```
      ))

(setq *SCR-OP-SELECT-RULES*  nil)

(setq *SCR-BINDINGS-SELECT-RULES* nil)


;;;************************************************************

(setq *SCR-NODE-REJECT-RULES*
      '(
        ; the following control rule avoids using welders to change the
        ; material of a part.

        (NEVER-CHANGE-MATERIAL
           (lhs (and (candidate-node <node>)
                     (candidate-goal <node> (material-of <part> <material>))
                     (known <node> (material-of <part> <material>))))
           (rhs (reject node <node>)))

       ; the following rules may be too restrictive.
       ; if one wants to give the top goal of making a cylinder,
       ; then the goal has to be given as (size-of <part> DIAMETER <x>).

        (NEVER-MAKE-CYLINDERS
           (lhs (and (candidate-node <node>)
                     (candidate-goal <node> (shape-of <part> CYLINDRICAL))
                     (known <node> (size-of <part> WIDTH <w>))))
           (rhs (reject node <node>)))

        (NEVER-MAKE-RECTANGLES
           (lhs (and (candidate-node <node>)
                     (candidate-goal <node> (shape-of <part> RECTANGULAR))
                     (known <node> (size-of <part> DIAMETER <d>))))
           (rhs (reject node <node>)))
        ))

 (setq *SCR-GOAL-REJECT-RULES* nil)

 (setq *SCR-OP-REJECT-RULES*
       '(
            (DONT-MAKE-OBJ-WITH-METAL-ARC-WELDER
             (lhs (and (current-node <node>)
                       (current-goal <node> (is-a <x> PART))
                       (candidate-op <node> WELD-CYLINDERS-METAL-ARC)))
             (rhs (reject operator WELD-CYLINDERS-METAL-ARC)))

            (DONT-MAKE-OBJ-WITH-GAS-WELDER
             (lhs (and (current-node <node>)
                       (current-goal <node> (is-a <x> PART))
                       (candidate-op <node> WELD-CYLINDERS-GAS)))
             (rhs (reject operator WELD-CYLINDERS-GAS)))

            (DONT-HOLD-WITH-METAL-ARC-WELDER
             (lhs (and (current-node <node>)
                       (current-goal <node>
                          (holding <machine> <holding-device> <part> <side>))
                       (candidate-op <node> WELD-CYLINDERS-METAL-ARC)))
             (rhs (reject operator WELD-CYLINDERS-METAL-ARC)))

            (DONT-HOLD-WITH-GAS-WELDER
             (lhs (and (current-node <node>)
                       (current-goal <node>
                          (holding <machine> <holding-device> <part> <side>))
                       (candidate-op <node> WELD-CYLINDERS-GAS)))
             (rhs (reject operator WELD-CYLINDERS-GAS)))
       ))


 (setq *SCR-BINDINGS-REJECT-RULES* nil)


 ;;;************************************************************

 (setq *SCR-NODE-PREFERENCE-RULES* nil)

 (setq *SCR-GOAL-PREFERENCE-RULES*  nil)

 (setq *SCR-OP-PREFERENCE-RULES* nil)

 (setq *SCR-BINDINGS-PREFERENCE-RULES* nil)
```

2. A Sample Machine Shop

The following is the representation of a job shop that contains a wide variety of machines, tools, holding devices, and parts as they are described in this document. See for example [CLEVELAND, 1984] for a wider selection of tools.

```
;____________________________________________________________

(is-a milling-machine1 MILLING-MACHINE)
(is-a drill1 DRILL)
(is-a lathe1 LATHE)
(is-a shaper1 SHAPER)
(size-of-machine shaper1 20)
(is-a planer1 PLANER)
(is-a grinder1 GRINDER)
(is-a circular-saw1 CIRCULAR-SAW)
(is-a band-saw1 BAND-SAW)
(is-a welder1 METAL-ARC-WELDER)
(is-a oxygentank-regulator-torch1 GAS-WELDER)
(is-a electric-spray-gun1 ELECTRIC-ARC-SPRAY-GUN)

(is-a milling-machine2 MILLING-MACHINE)
(is-a drill2 DRILL)
(is-a lathe2 LATHE)
(is-a shaper2 SHAPER)
(size-of-machine shaper2 20)
(is-a planer2 PLANER)
(is-a grinder2 GRINDER)
(is-a circular-saw2 CIRCULAR-SAW)
(is-a band-saw2 BAND-SAW)

;____________________________________________________________

(is-a spot-drill1 SPOT-DRILL)

(is-a center-drill1 CENTER-DRILL)
(diameter-of-drill-bit center-drill1 1/16)

(is-a twist-drill1 TWIST-DRILL)
(diameter-of-drill-bit twist-drill1 1/16)

(is-a twist-drill2 TWIST-DRILL)
(diameter-of-drill-bit twist-drill2 5/64)

(is-a twist-drill3 TWIST-DRILL)
(diameter-of-drill-bit twist-drill3 3/32)

(is-a twist-drill4 TWIST-DRILL)
(diameter-of-drill-bit twist-drill4 7/64)

(is-a twist-drill5 TWIST-DRILL)
(diameter-of-drill-bit twist-drill5 1/8)

(is-a twist-drill6 TWIST-DRILL)
(diameter-of-drill-bit twist-drill6 9/64)

(is-a twist-drill7 TWIST-DRILL)
(diameter-of-drill-bit twist-drill7 5/32)

(is-a twist-drill8 TWIST-DRILL)
(diameter-of-drill-bit twist-drill8 11/64)

(is-a twist-drill9 TWIST-DRILL)
(diameter-of-drill-bit twist-drill9 3/16)

(is-a twist-drill10 TWIST-DRILL)
(diameter-of-drill-bit twist-drill10 13/64)

(is-a twist-drill11 TWIST-DRILL)
(diameter-of-drill-bit twist-drill11 7/32)

(is-a twist-drill12 TWIST-DRILL)
(diameter-of-drill-bit twist-drill12 15/64)

(is-a twist-drill13 TWIST-DRILL)
(diameter-of-drill-bit twist-drill13 1/4)

(is-a twist-drill14 TWIST-DRILL)
(diameter-of-drill-bit twist-drill14 5/16)

(is-a twist-drill15 TWIST-DRILL)
(diameter-of-drill-bit twist-drill15 3/8)

(is-a twist-drill16 TWIST-DRILL)
(diameter-of-drill-bit twist-drill16 7/16)
```

```
(is-a twist-drill17 TWIST-DRILL)
(diameter-of-drill-bit twist-drill17 1/2)

(is-a twist-drill18 TWIST-DRILL)
(diameter-of-drill-bit twist-drill18 9/16)

(is-a twist-drill19 TWIST-DRILL)
(diameter-of-drill-bit twist-drill19 5/8)

(is-a twist-drill20 TWIST-DRILL)
(diameter-of-drill-bit twist-drill20 11/16)

(is-a twist-drill21 TWIST-DRILL)
(diameter-of-drill-bit twist-drill21 3/4)

(is-a twist-drill22 TWIST-DRILL)
(diameter-of-drill-bit twist-drill22 13/16)

(is-a twist-drill23 TWIST-DRILL)
(diameter-of-drill-bit twist-drill23 7/8)

(is-a twist-drill24 TWIST-DRILL)
(diameter-of-drill-bit twist-drill24 15/16)

(is-a twist-drill25 TWIST-DRILL)
(diameter-of-drill-bit twist-drill25 63/64)

(is-a twist-drill26 TWIST-DRILL)
(diameter-of-drill-bit twist-drill26 1)

(is-a straight-fluted-drill1 STRAIGHT-FLUTED-DRILL)
(diameter-of-drill-bit straight-fluted-drill1 1/32)

(is-a straight-fluted-drill2 STRAIGHT-FLUTED-DRILL)
(diameter-of-drill-bit straight-fluted-drill2 1/16)

(is-a straight-fluted-drill3 STRAIGHT-FLUTED-DRILL)
(diameter-of-drill-bit straight-fluted-drill3 3/32)

(is-a high-helix-drill1 HIGH-HELIX-DRILL)
(diameter-of-drill-bit high-helix-drill1 1/32)

(is-a high-helix-drill2 HIGH-HELIX-DRILL)
(diameter-of-drill-bit high-helix-drill2 1/16)

(is-a high-helix-drill3 HIGH-HELIX-DRILL)
(diameter-of-drill-bit high-helix-drill3 3/32)

(is-a oil-hole-drill1 OIL-HOLE-DRILL)
(diameter-of-drill-bit oil-hole-drill1 1/32)

(is-a oil-hole-drill2 OIL-HOLE-DRILL)
(diameter-of-drill-bit oil-hole-drill2 1/16)

(is-a oil-hole-drill3 OIL-HOLE-DRILL)
(diameter-of-drill-bit oil-hole-drill3 3/32)

(is-a gun-drill1 GUN-DRILL)
(diameter-of-drill-bit gun-drill1 1/32)

(is-a gun-drill2 GUN-DRILL)
(diameter-of-drill-bit gun-drill2 1/16)

(is-a gun-drill3 GUN-DRILL)
(diameter-of-drill-bit gun-drill3 3/32)

(is-a core-drill1 CORE-DRILL)
(diameter-of-drill-bit core-drill1 1/32)

(is-a core-drill2 CORE-DRILL)
(diameter-of-drill-bit core-drill2 1/16)

(is-a core-drill3 CORE-DRILL)
(diameter-of-drill-bit core-drill3 3/32)

(is-a tap1 TAP)
(diameter-of-drill-bit tap1 1/32)

(is-a tap2 TAP)
(diameter-of-drill-bit tap2 1/16)
```

```
(is-a tap3 TAP)
(diameter-of-drill-bit tap3 5/64)

(is-a tap4 TAP)
(diameter-of-drill-bit tap4 3/32)

(is-a tap5 TAP)
(diameter-of-drill-bit tap5 7/64)

(is-a tap6 TAP)
(diameter-of-drill-bit tap6 1/8)

(is-a tap7 TAP)
(diameter-of-drill-bit tap7 9/64)

(is-a tap8 TAP)
(diameter-of-drill-bit tap8 5/32)

(is-a tap9 TAP)
(diameter-of-drill-bit tap9 11/64)

(is-a tap10 TAP)
(diameter-of-drill-bit tap10 3/16)

(is-a tap11 TAP)
(diameter-of-drill-bit tap11 13/64)

(is-a tap12 TAP)
(diameter-of-drill-bit tap12 7/32)

(is-a tap13 TAP)
(diameter-of-drill-bit tap13 15/64)

(is-a tap14 TAP)
(diameter-of-drill-bit tap14 1/4)

(is-a tap15 TAP)
(diameter-of-drill-bit tap15 5/16)

(is-a tap16 TAP)
(diameter-of-drill-bit tap16 3/8)

(is-a tap17 TAP)
(diameter-of-drill-bit tap17 7/16)

(is-a tap18 TAP)
(diameter-of-drill-bit tap18 1/2)

(is-a tap19 TAP)
(diameter-of-drill-bit tap19 9/16)

(is-a tap20 TAP)
(diameter-of-drill-bit tap20 5/8)

(is-a tap21 TAP)
(diameter-of-drill-bit tap21 11/16)

(is-a tap22 TAP)
(diameter-of-drill-bit tap22 3/4)

(is-a tap23 TAP)
(diameter-of-drill-bit tap23 13/16)

(is-a tap24 TAP)
(diameter-of-drill-bit tap24 7/8)

(is-a tap25 TAP)
(diameter-of-drill-bit tap25 15/16)

(is-a tap26 TAP)
(diameter-of-drill-bit tap26 63/64)

(is-a tap27 TAP)
(diameter-of-drill-bit tap27 1)

(is-a countersink1 COUNTERSINK)
(angle-of-drill-bit countersink1 60)

(is-a countersink2 COUNTERSINK)
(angle-of-drill-bit countersink2 82)

(is-a countersink3 COUNTERSINK)
(angle-of-drill-bit countersink3 90)

(is-a countersink4 COUNTERSINK)
(angle-of-drill-bit countersink4 100)

(is-a counterbore1 COUNTERBORE)
(size-of-drill-bit counterbore1 1/4)

(is-a counterbore2 COUNTERBORE)
(size-of-drill-bit counterbore2 1/2)

(is-a counterbore3 COUNTERBORE)
(size-of-drill-bit counterbore3 3/4)

(is-a counterbore4 COUNTERBORE)
(size-of-drill-bit counterbore4 3/8)

(is-a reamer1 REAMER)
(diameter-of-drill-bit reamer1 1/32)

(is-a reamer2 REAMER)
(diameter-of-drill-bit reamer2 1/16)

(is-a reamer3 REAMER)
(diameter-of-drill-bit reamer3 3/32)

;__________________________________________________________________

(is-a milling-cutter1 PLAIN-MILL)
(is-a milling-cutter2 END-MILL)

(is-a rough-toolbit1 ROUGH-TOOLBIT)
(is-a finish-toolbit1 FINISH-TOOLBIT)
(is-a v-thread1 V-THREAD)
(is-a knurl1 KNURL)

(is-a cutting-tool1 ROUGHING-CUTTING-TOOL)
(is-a cutting-tool2 FINISHING-CUTTING-TOOL)

(is-a grinding-wheel1 GRINDING-WHEEL)
(hardness-of-wheel grinding-wheel1 HARD)
(grit-of-wheel grinding-wheel1 COARSE-GRIT)

(is-a grinding-wheel2 GRINDING-WHEEL)
(hardness-of-wheel grinding-wheel2 SOFT)
(grit-of-wheel grinding-wheel2 COARSE-GRIT)

(is-a grinding-wheel3 GRINDING-WHEEL)
(hardness-of-wheel grinding-wheel3 HARD)
(grit-of-wheel grinding-wheel3 FINE-GRIT)

(is-a grinding-wheel4 GRINDING-WHEEL)
(hardness-of-wheel grinding-wheel4 SOFT)
(grit-of-wheel grinding-wheel4 FINE-GRIT)

(is-a saw-attachment1 SAW-BAND)
(is-a saw-attachment2 BAND-FILE)

(is-a saw-attachment3 FRICTION-SAW)
(is-a saw-attachment4 COLD-SAW)

(is-a milling-cutter11 PLAIN-MILL)
(is-a milling-cutter12 END-MILL)

(is-a rough-toolbit11 ROUGH-TOOLBIT)
(is-a finish-toolbit11 FINISH-TOOLBIT)
(is-a v-thread11 V-THREAD)
(is-a knurl11 KNURL)

(is-a cutting-tool11 ROUGHING-CUTTING-TOOL)
(is-a cutting-tool12 FINISHING-CUTTING-TOOL)

(is-a grinding-wheel11 GRINDING-WHEEL)
(hardness-of-wheel grinding-wheel11 HARD)
(grit-of-wheel grinding-wheel11 COARSE-GRIT)

(is-a grinding-wheel12 GRINDING-WHEEL)
(hardness-of-wheel grinding-wheel12 SOFT)
(grit-of-wheel grinding-wheel12 COARSE-GRIT)

(is-a grinding-wheel13 GRINDING-WHEEL)
(hardness-of-wheel grinding-wheel13 HARD)
(grit-of-wheel grinding-wheel13 FINE-GRIT)

(is-a grinding-wheel14 GRINDING-WHEEL)
(hardness-of-wheel grinding-wheel14 SOFT)
(grit-of-wheel grinding-wheel14 FINE-GRIT)

(is-a electrode1 ELECTRODE)
(is-a rod1 WELDING-ROD)

(is-a wire1 SPRAYING-METAL-WIRE)
(material-of wire1 STAINLESS-STEEL)

(is-a wire2 SPRAYING-METAL-WIRE)
(material-of wire2 ZIRCONIUM-OXIDE)

(is-a wire3 SPRAYING-METAL-WIRE)
(material-of wire3 ALUMINUM-OXIDE)

(is-a wire10 SPRAYING-METAL-WIRE)
(material-of wire10 TUNGSTEN)

(is-a wire11 SPRAYING-METAL-WIRE)
```

```
(material-of wire11 MOLYBDENUM)

;____________________________________________________________

(is-a v-block1 V-BLOCK)
(is-a vise1 VISE)
(is-a toe-clamp1 TOE-CLAMP)
(is-a centers1 CENTERS)
(is-a 4-jaw-chuck1 4-JAW-CHUCK)
(is-a collet-chuck1 COLLET-CHUCK)
(is-a magnetic-chuck1 MAGNETIC-CHUCK)

(is-a v-block2 V-BLOCK)
(is-a vise2 VISE)
(is-a toe-clamp2 TOE-CLAMP)
(is-a centers2 CENTERS)
(is-a 4-jaw-chuck2 4-JAW-CHUCK)
(is-a collet-chuck2 COLLET-CHUCK)
(is-a magnetic-chuck2 MAGNETIC-CHUCK)

(is-a v-block3 V-BLOCK)
(is-a vise3 VISE)
(is-a toe-clamp3 TOE-CLAMP)
(is-a centers3 CENTERS)
(is-a 4-jaw-chuck3 4-JAW-CHUCK)
(is-a collet-chuck3 COLLET-CHUCK)
(is-a magnetic-chuck3 MAGNETIC-CHUCK)

(is-a lathe-file1 LATHE-FILE)
(is-a abrasive-cloth1 ABRASIVE-CLOTH)
(material-of-abrasive-cloth abrasive-cloth1 EMERY)

(is-a lathe-file2 LATHE-FILE)
(is-a abrasive-cloth2 ABRASIVE-CLOTH)
(material-of-abrasive-cloth abrasive-cloth2 EMERY)

(is-a lathe-file3 LATHE-FILE)
(is-a abrasive-cloth3 ABRASIVE-CLOTH)
(material-of-abrasive-cloth abrasive-cloth3 EMERY)

(is-a torch1 TORCH)
;____________________________________________________________

(is-a brush1 BRUSH)
(is-a brush2 BRUSH)

(is-a soluble-oil SOLUBLE-OIL)
(is-a mineral-oil MINERAL-OIL)

;____________________________________________________________

(is-a part1 PART)
(material-of part1 BRASS)
(size-of part1 LENGTH 5)
(size-of part1 WIDTH 3)
(size-of part1 HEIGHT 3)

(is-a part2 PART)
(material-of part2 COPPER)
(size-of part2 LENGTH 5)
(size-of part2 WIDTH 3)
(size-of part2 HEIGHT 3)

(is-a part3 PART)
(material-of part3 BRONZE)
(size-of part3 LENGTH 5)
(size-of part3 WIDTH 3)
(size-of part3 HEIGHT 3)

(is-a part5 PART)
(material-of part5 ALUMINUM)
(size-of part5 LENGTH 5)
(size-of part5 WIDTH 3)
(size-of part5 HEIGHT 3)

(is-a part6 PART)
(material-of part6 STEEL)
(size-of part6 LENGTH 5)
(size-of part6 WIDTH 3)
(size-of part6 HEIGHT 3)

(is-a part7 PART)
(material-of part7 IRON)
(size-of part7 LENGTH 5)
(size-of part7 WIDTH 3)
(size-of part7 HEIGHT 3)

(is-a part21 PART)
(material-of part21 BRASS)
(size-of part21 LENGTH 7)
(size-of part21 WIDTH 4)
(size-of part21 HEIGHT 2)
```

```
(is-a part22 PART)
(material-of part22 COPPER)
(size-of part22 LENGTH 7)
(size-of part22 WIDTH 4)
(size-of part22 HEIGHT 2)

(is-a part22 PART)
(material-of part22 BRONZE)
(size-of part22 LENGTH 7)
(size-of part22 WIDTH 4)
(size-of part22 HEIGHT 2)

(is-a part27 PART)
(material-of part27 ALUMINUM)
(size-of part27 LENGTH 7)
(size-of part27 WIDTH 4)
(size-of part27 HEIGHT 2)

(is-a part26 PART)
(material-of part26 STEEL)
(size-of part26 LENGTH 7)
(size-of part26 WIDTH 4)
(size-of part26 HEIGHT 2)

(is-a part27 PART)
(material-of part27 IRON)
(size-of part27 LENGTH 7)
(size-of part27 WIDTH 4)
(size-of part27 HEIGHT 2)

(is-a part51 PART)
(material-of part51 BRASS)
(size-of part51 LENGTH 5)
(size-of part51 WIDTH 2)
(size-of part51 HEIGHT 3)


(is-a part55 PART)
(material-of part55 ALUMINUM)
(size-of part55 LENGTH 5)
(size-of part55 WIDTH 4)
(size-of part55 HEIGHT 2)


;____________________________________________________________


(is-a part11 PART)
(material-of part11 BRASS)
(size-of part11 LENGTH 5)
(size-of part11 DIAMETER 3)

(is-a part12 PART)
(material-of part12 COPPER)
(size-of part12 LENGTH 5)
(size-of part12 DIAMETER 3)

(is-a part13 PART)
(material-of part13 BRONZE)
(size-of part13 LENGTH 5)
(size-of part13 DIAMETER 3)

(is-a part15 PART)
(material-of part15 ALUMINUM)
(size-of part15 LENGTH 5)
(size-of part15 DIAMETER 3)

(is-a part16 PART)
(material-of part16 STEEL)
(size-of part16 LENGTH 5)
(size-of part16 DIAMETER 3)

(is-a part17 PART)
(material-of part17 IRON)
(size-of part17 LENGTH 5)
(size-of part17 DIAMETER 3)


(is-a part31 PART)
(material-of part31 BRASS)
(size-of part31 LENGTH 10)
(size-of part31 DIAMETER 5)

(is-a part32 PART)
(material-of part32 COPPER)
(size-of part32 LENGTH 10)
(size-of part32 DIAMETER 5)

(is-a part33 PART)
(material-of part33 BRONZE)
(size-of part33 LENGTH 10)
(size-of part33 DIAMETER 5)
```

```
(is-a part35 PART)
(material-of part35 ALUMINUM)
(size-of part35 LENGTH 10)
(size-of part35 DIAMETER 5)

(is-a part36 PART)
(material-of part36 STEEL)
(size-of part36 LENGTH 10)
(size-of part36 DIAMETER 5)

(is-a part37 PART)
(material-of part37 IRON)
(size-of part37 LENGTH 10)
(size-of part37 DIAMETER 5)

(is-a part136 PART)
(material-of part136 STEEL)
(size-of part136 LENGTH 3)
(size-of part136 DIAMETER 5)
```