Appendix
Mechanical Device Representations

This appendix is devoted to an in-depth presentation of the static, dynamic, and pragmatic representations for each of the devices that have been used in this research. Fig. A.1 illustrates the four mechanical device classes which are presented in this appendix: (1) simple devices, (2) simple compound devices, (3) multiple compound devices, and (4) complex devices

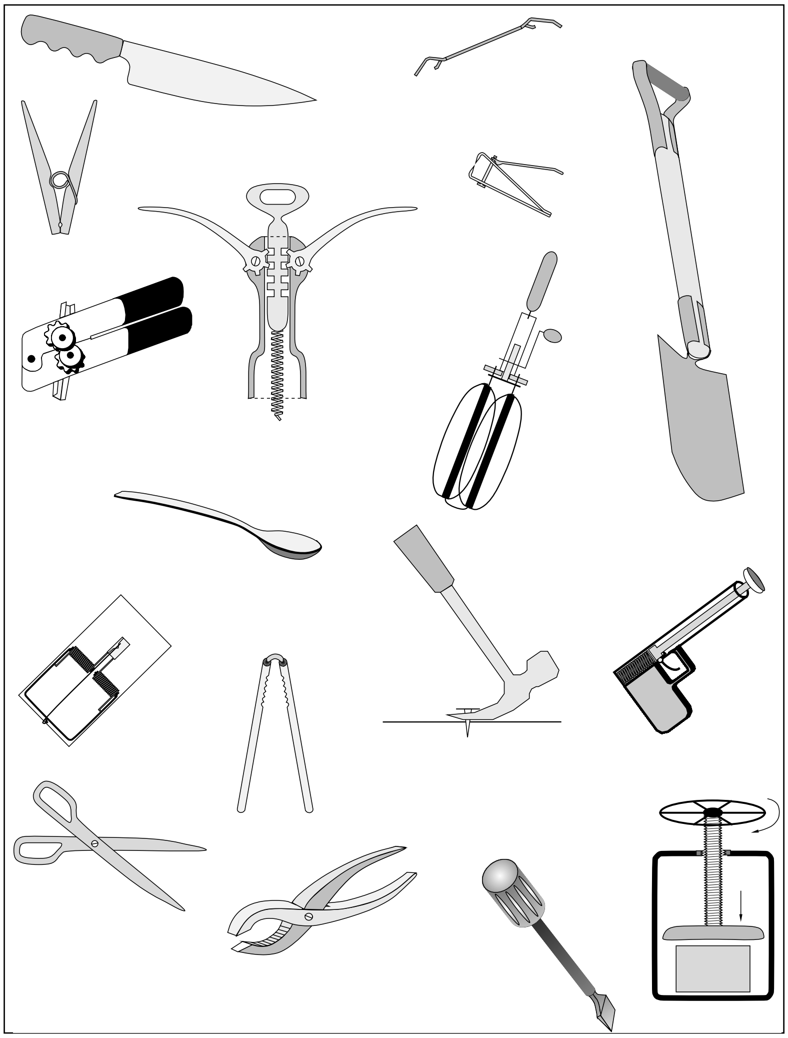


Figure A.1 FONM Devices.

A.1 Simple Devices

Simple device dynamics are representationally significant because even simple devices can be used to accomplish multiple tasks. The dynamic representation must make a device's different functions explicit with the same static representation. The distinction between device functions is often seen as a specialization on the type of input applied to a device, or on the physical properties of objects which comprise the device. Simple devices are also useful for showing that device descriptions which represent the same function, whether based on components or regions, are equivalent. The simple devices presented in this section all consist of components which are rigidly connected, or regions on a single component. As in Section 2.7.1, five devices will be addressed in this section:

(1) carving knife
(2) hammer
(3) shovel
(4) spoon

A.1.1 Carving Knife

A carving knife is designed for slicing and puncturing. One end is blunt and used as a handle, while the other is sharp and pointed.

Carving Knife Statics

A knife is comprised of a handle and a blade. The blade has an edge and a pointed tip, as illustrated in Fig. A.2 below.

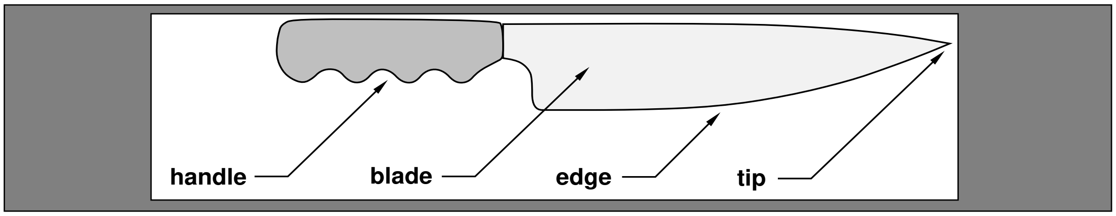


Figure A.2 Carving knife illustration.

An idealized carving knife and its static device diagram are illustrated in Fig. A.3.

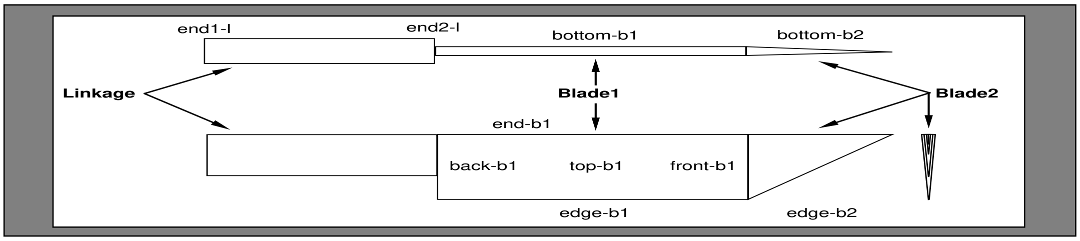


Figure A.3 Carving knife object primitives and static device diagram.

The linkage-object is rigidly connected to the wedge-shaped blade-object, which is itself rigidly connected to the tetrahedron-shaped blade-object. The SDD shown in Fig. A.4

illustrates these connections.

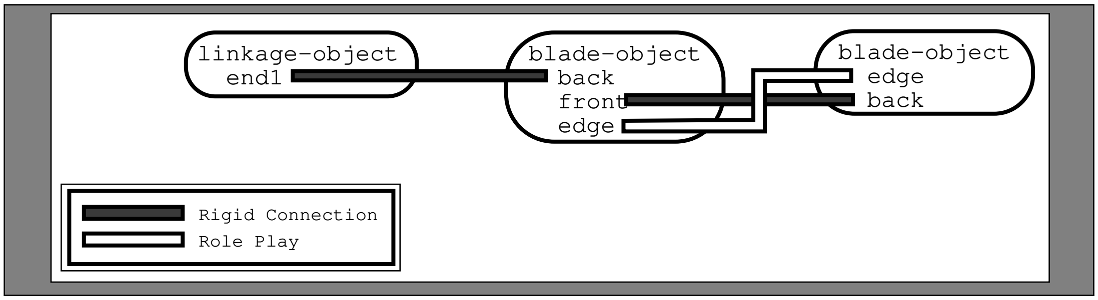


Figure A.4 Carving knife static representation diagram.

The two blade-object edges are actually the same edge, so there is a role-play link between the two objects. Role-play links identify which frame roles are related between representation schema, so that inferences about connectivity and force transmission are continuous. The carving knife objects share the colinear orientation of objects that the bottle opener has, so their representation will be skipped here.

Low-Level Carving Knife Dynamics

The behavioral sequences for knife components in the cutting function are shown in Figs. A.5, A.6. Only one of the blade sequences is shown. Note the similarities between the knife handle and the screwdriver handles, and between the can-opener blade and the knife blade..

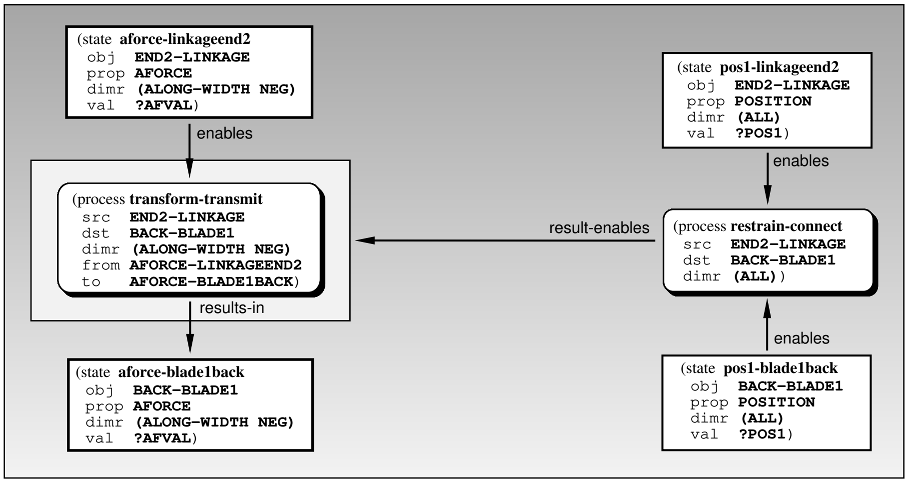


Figure A.5 Carving knife handle dynamics for cutting function.

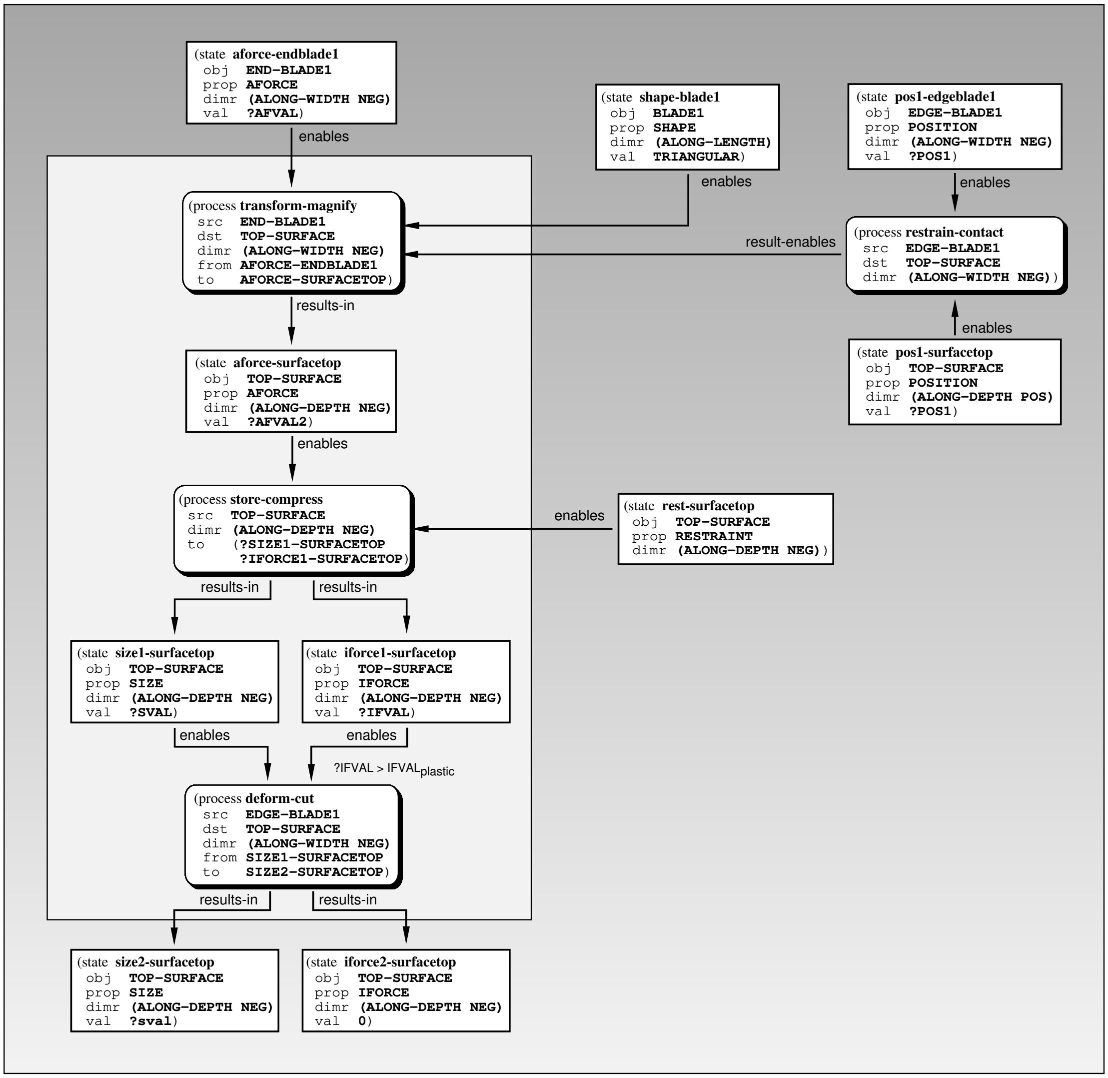


Figure A.6 Carving knife blade dynamics for cutting function.

High-Level Carving Knife Dynamics

Two functions which the carving knife statics instantiate are depicted in Fig. A.7. When the applied force is lateral (as shown at 1), the knife handle instantiates the function of force translation (MP-LINKAGE-TYPE2).

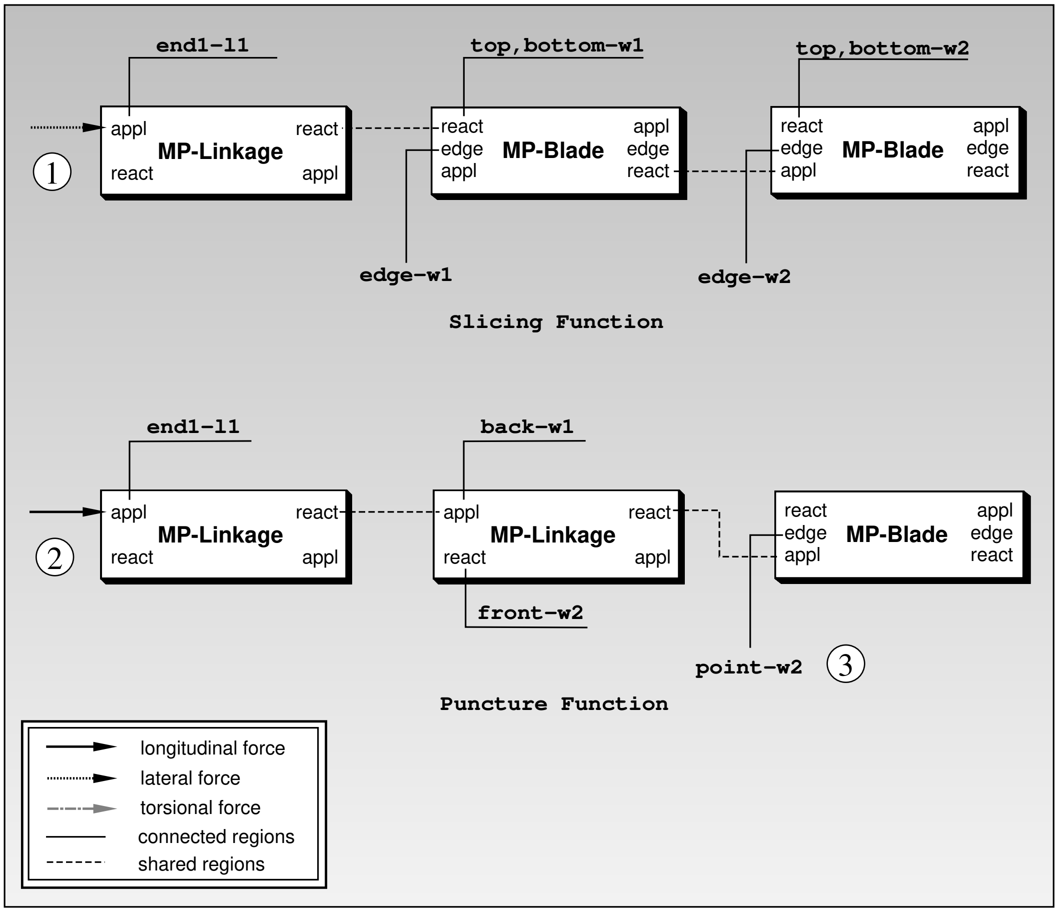
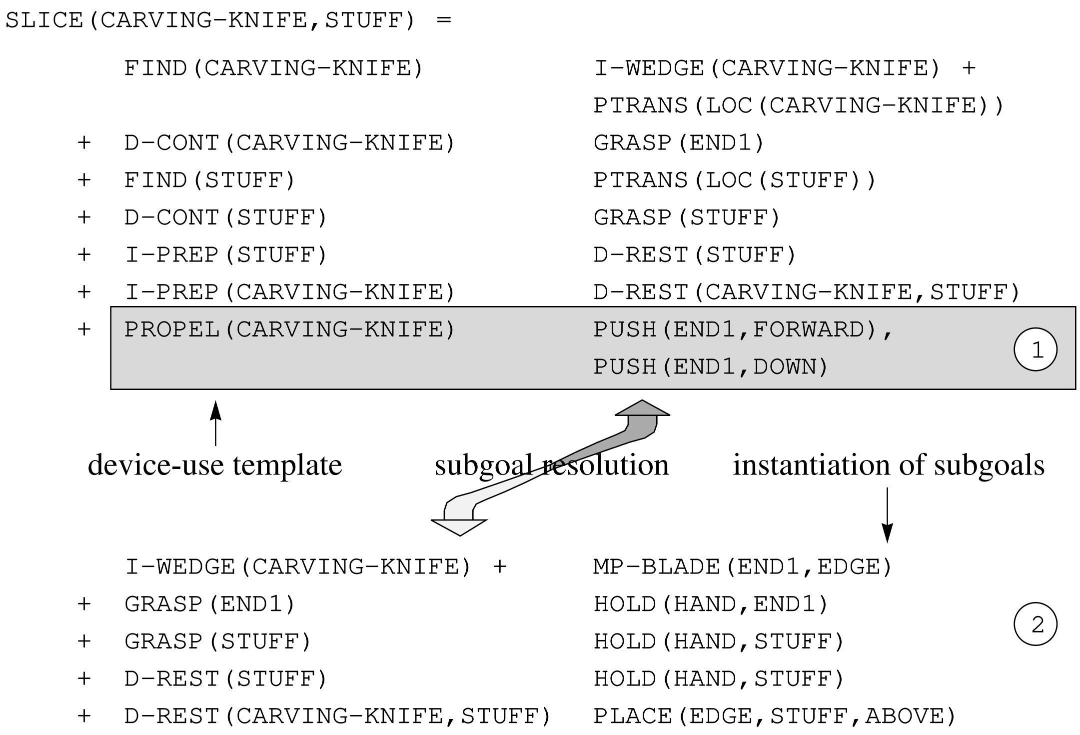


Figure A.7 MP-Diagrams for carving knife functions.

The resulting force and shape of the knife blade enable MP-BLADE and the cutting function at the wedge edges. When the applied force is longitudinal (2), MP-LINKAGE-TYPE1 is enabled, and the force is propagated through the knife to its end (3). In this case, the end of the blade is wedge-shaped both longitudinally and laterally, so MP-BLADE also represents the function, however, the edge role is now instantiated by the knife point.

Carving Knife Pragmatics

The carving knife has a single design-intended use: (1) slicing, as shown in Fig. A.8. Slicing is an SOP type plan, in which some material STUFF is removed from a body of STUFF. The SOP plan requires a device which can instantiate MP-BLADE. The HOLD action is used in this plan to produce the necessary restraint on STUFF so that it will not move with respect to the carving-knife, as well as to control the carving-knife. When illustrating actions such as HOLD, the body part will be identified first, and the object second. HOLD(-HAND,STUFF) means that the hand is holding the stuff. It is assumed that a hand can only hold one object, so the two HOLD actions (2) must support the inference that different hands are being used to perform these tasks. The PUSH actions use a similar notation. The region where the application is made, as well as the direction, are identified in the figure. In this case, two actions are depicted to clarify the directions of application. No notion of cyclic motion is shown in this representation. For example, the PUSH action would enable MOTION-LINEAR, and the carving-knife could move forward and downward into STUFF. What is missing in this figure is the known MP-BLADE bounding state that, when the end of the blade is reached during slicing, motion must be reversed.



```
SLICE(CARVING-KNIFE,STUFF) =
        FIND(CARVING-KNIFE)             I-WEDGE(CARVING-KNIFE) +
                                        PTRANS(LOC(CARVING-KNIFE))
     +  D-CONT(CARVING-KNIFE)           GRASP(END1)
     +  FIND(STUFF)                     PTRANS(LOC(STUFF))
     +  D-CONT(STUFF)                   GRASP(STUFF)
     +  I-PREP(STUFF)                   D-REST(STUFF)
     +  I-PREP(CARVING-KNIFE)           D-REST(CARVING-KNIFE,STUFF)
     +  PROPEL(CARVING-KNIFE)           PUSH(END1,FORWARD),
                                        PUSH(END1,DOWN)              ①

        I-WEDGE(CARVING-KNIFE) +        MP-BLADE(END1,EDGE)
     +  GRASP(END1)                     HOLD(HAND,END1)
     +  GRASP(STUFF)                    HOLD(HAND,STUFF)             ②
     +  D-REST(STUFF)                   HOLD(HAND,STUFF)
     +  D-REST(CARVING-KNIFE,STUFF)     PLACE(EDGE,STUFF,ABOVE)
```

Figure A.8 Pragmatic carving knife representation.

A.1.2 Shovel

A shovel is depicted in Fig. A.9. This shovel is comprised of three components: a handle, a shaft, and a scoop.

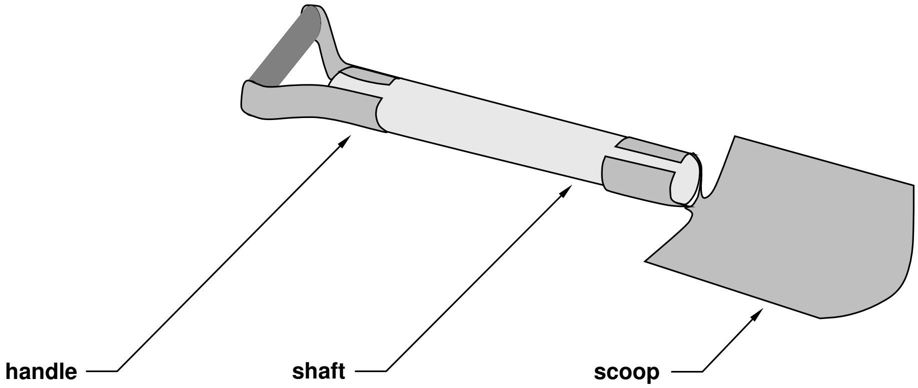


Figure A.9 Shovel illustration.

A shovel has a scoop or container at one end, a long shaft, and a handle at the other end.

Shovel Statics

The scoop is often sharp. An idealized shovel and its static device diagram are depicted in Fig. A.10.

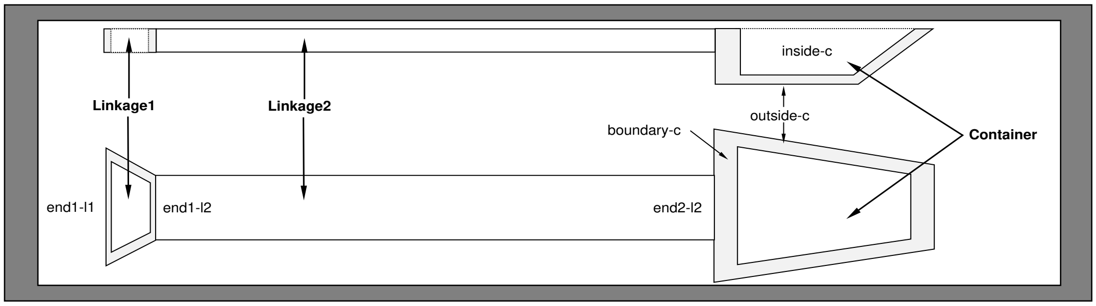


Figure A.10 Shovel object primitives and static device diagram.

The linkage-object which represents the handle has a hole in it, and the hole is approximately the size of the object. The hole is not represented in the SDD shown in Fig. A.11, since the handle can also be blunt..

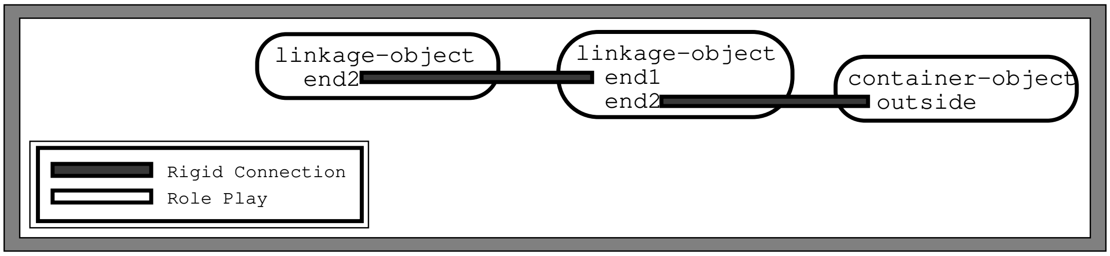
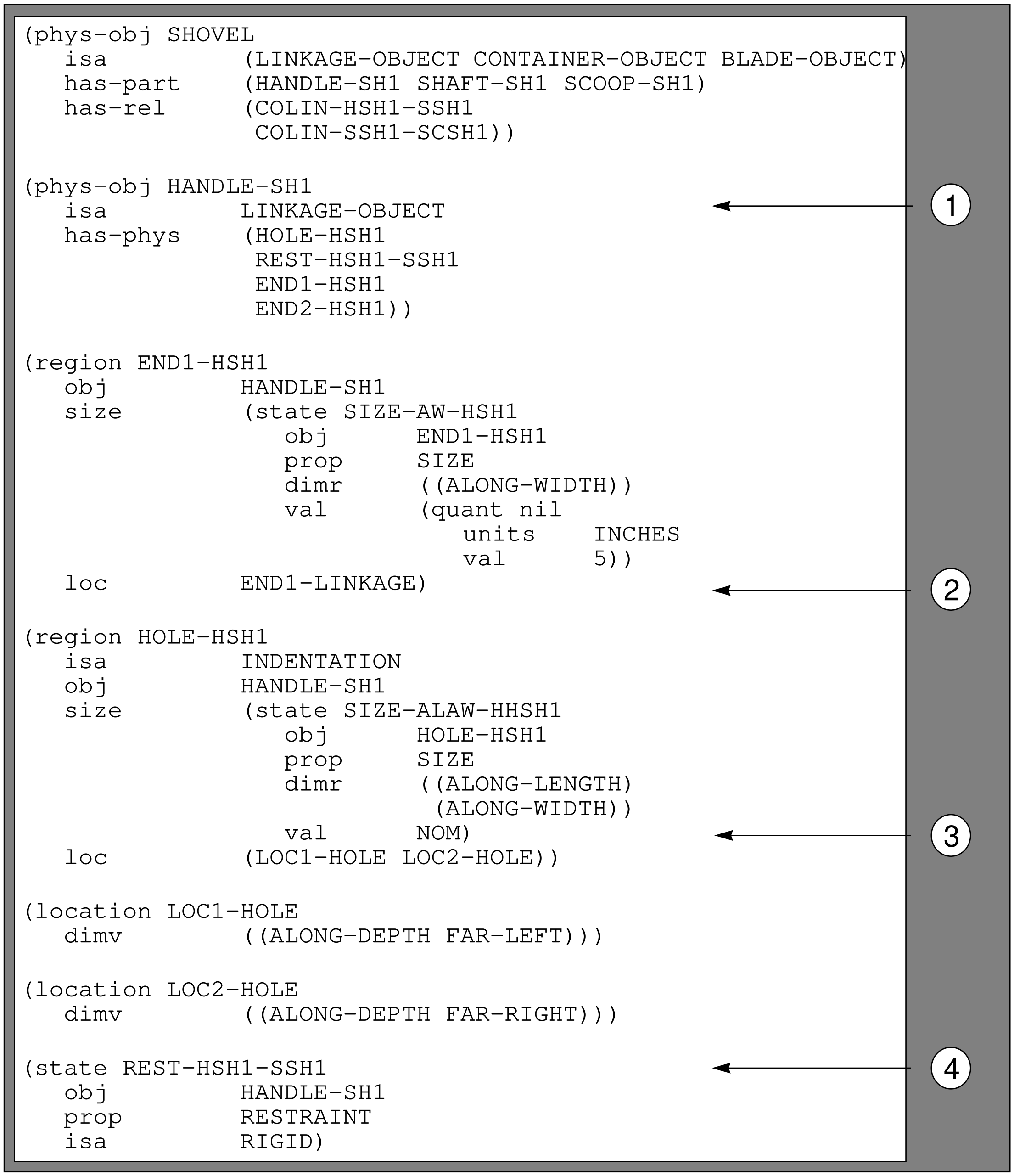


Figure A.11 Shovel static representation diagram.

The container-object is wedge shaped. These relationships are detailed in the accompanying representation (Fig. A.12). In this figure, the shovel is shown having three components and two spatial relations. The shovel handle is represented as a linkage-object (1) with a hole region and two end regions. It also has a local restraint state (4). The restraint is rigid, meaning that all local dimensions are fixed.



```
(phys-obj SHOVEL
   isa          (LINKAGE-OBJECT CONTAINER-OBJECT BLADE-OBJECT)
   has-part     (HANDLE-SH1 SHAFT-SH1 SCOOP-SH1)
   has-rel      (COLIN-HSH1-SSH1
                 COLIN-SSH1-SCSH1))

(phys-obj HANDLE-SH1
   isa         LINKAGE-OBJECT
   has-phys     (HOLE-HSH1
                 REST-HSH1-SSH1
                 END1-HSH1
                 END2-HSH1))

(region END1-HSH1
   obj         HANDLE-SH1
   size         (state SIZE-AW-HSH1
                   obj      END1-HSH1
                   prop     SIZE
                   dimr     ((ALONG-WIDTH))
                   val      (quant nil
                               units    INCHES
                               val      5))
   loc         END1-LINKAGE)

(region HOLE-HSH1
   isa         INDENTATION
   obj         HANDLE-SH1
   size         (state SIZE-ALAW-HHSH1
                   obj      HOLE-HSH1
                   prop     SIZE
                   dimr     ((ALONG-LENGTH)
                             (ALONG-WIDTH))
                   val      NOM)
   loc          (LOC1-HOLE LOC2-HOLE))

(location LOC1-HOLE
   dimv         ((ALONG-DEPTH FAR-LEFT)))

(location LOC2-HOLE
   dimv         ((ALONG-DEPTH FAR-RIGHT)))

(state REST-HSH1-SSH1
   obj         HANDLE-SH1
   prop        RESTRAINT
   isa         RIGID)
```

Figure A.12 Physical and relational representation for the shovel handle and its rigi connection to the shovel shaft.

The handle's left end is located at the same place as the linkage-objects left end (2), as is the handle's right end with the linkage-object's right end. The hole is represented as an indentation which is located at both extremes of the handle, and which has approximately the same area (NOM, at 3, in length and width) as the handle.

Low-Level Shovel Dynamics

The behavioral sequences for the shovel lifting function are illustrated in Figs. A.13 - A.15.

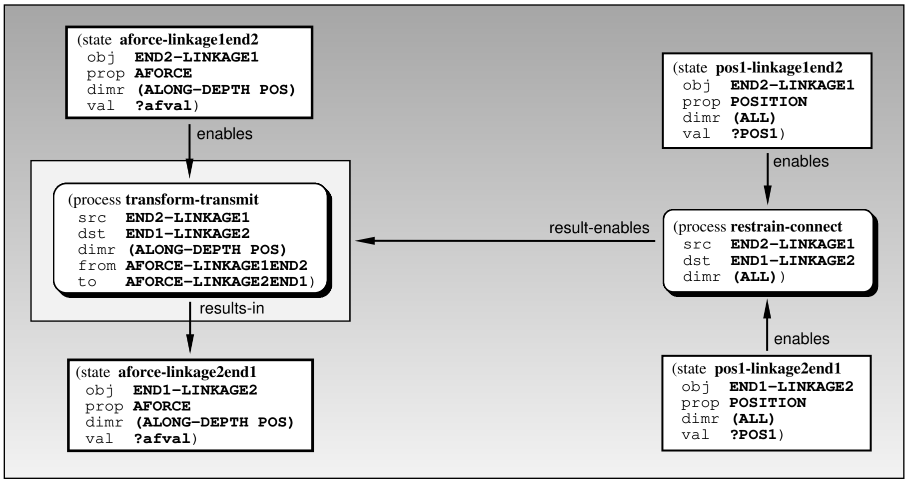


Figure A.13 Low-level shovel handle dynamics for lifting function

These two diagrams, except for their instantiations, are identical to the screwdriver handle and shaft low-level dynamics..

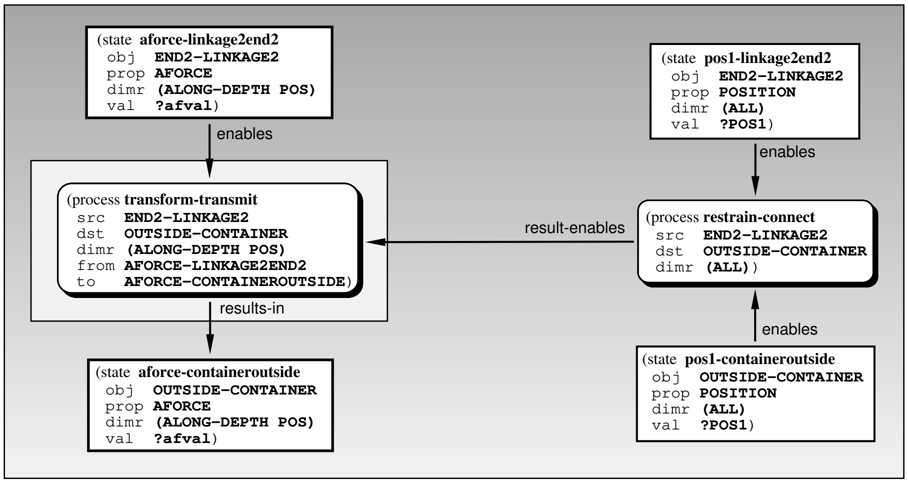


Figure A.14 Low-level shovel shaft dynamics for lifting function

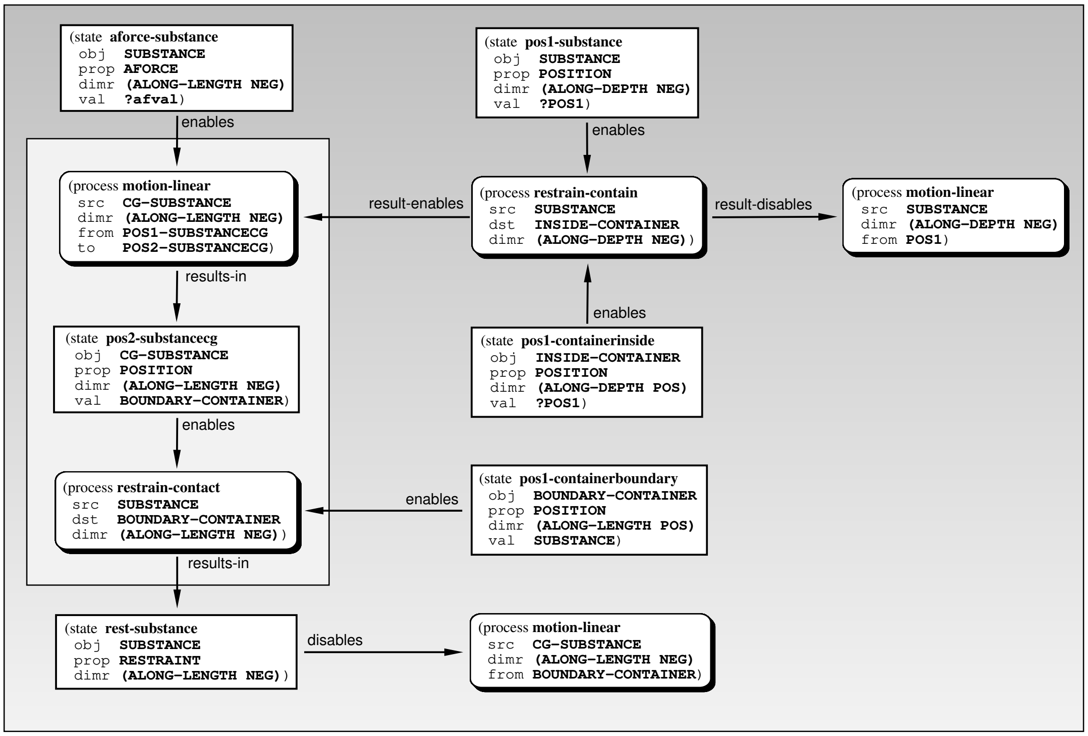


Figure A.15 Low-level shovel scoop dynamics for lifting function

High-Level Shovel Dynamics

A shovel can be used in many ways: for cutting, leveraging material, carrying, and throwing. The general use of a shovel involves four functions: (1) cutting the soil with the blade, (2) leveraging the soil from the new hole, (3) lifting and transporting the soil, and (4) flinging the soil. Each function is instantiated by applying force to different locations on the shovel while constraining its motion at others. These functions are shown as MP-Diagrams in Fig. A.16. The shovel functions add the complexity of multiple inputs to the representation of device function. When the shovel is used for digging, the handle is held (at 1) and and a longitudinal force is applied to the outside of the scoop (2). Since the scoop illustrated is wedge shaped, MP-BLADE is used to represent the cutting function. In the second function illustrated, the shovel handle is held and pushed (3) and the body is used as a fulcrum on the edge of the hole (4). In the diagram, the shovel body is shown instantiating the MP-LEVER pivot role (5), however, the instantiation is really the contact between the body and the hole.

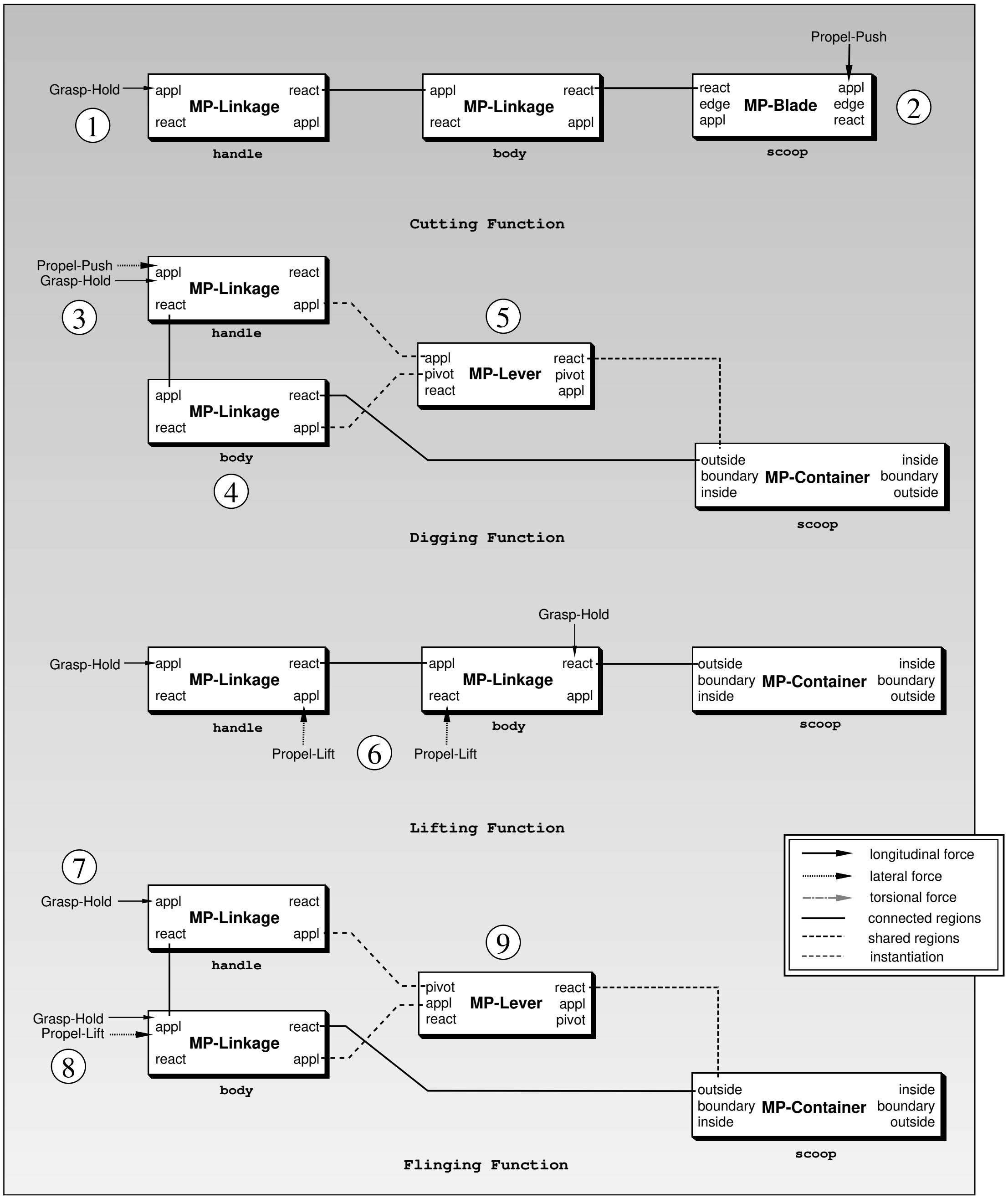


Figure A.16 MP-Diagrams for shovel functions.

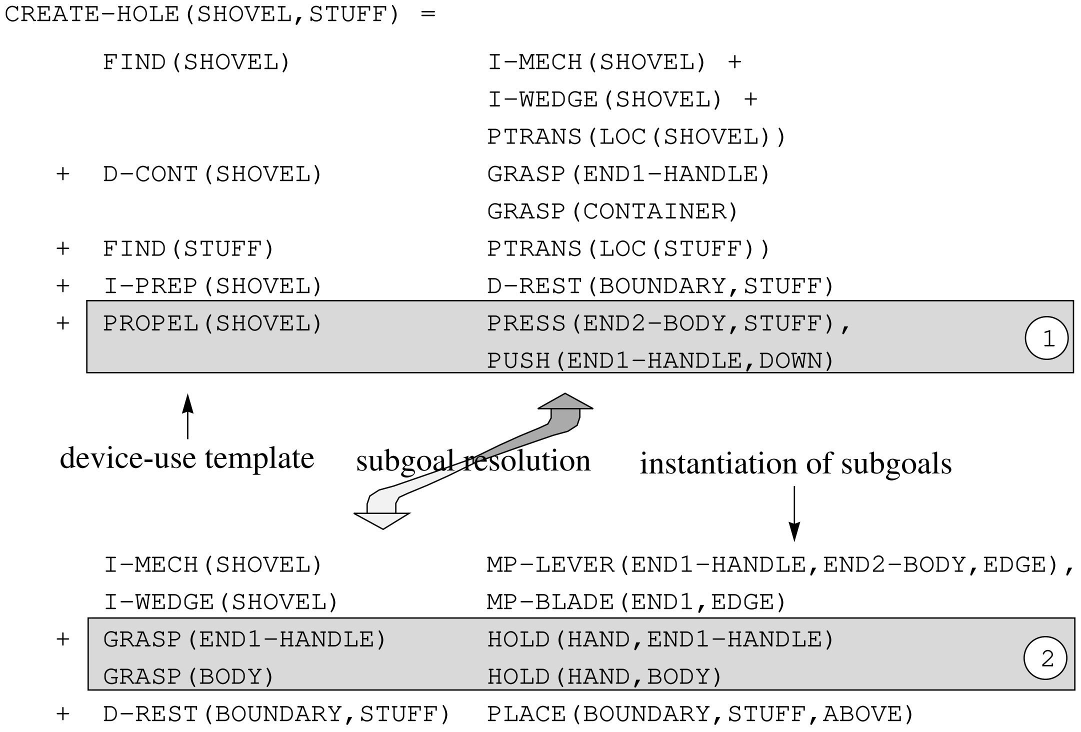
The light dotted lines depict the instantiation of MP-LEVER roles. The third function shows two lateral forces being applied to different regions (6), multiply enabling MP-LINKAGE-TYPE2. The last function illustrated shows the handle being held (7) and the body being lifted (8), which instantiates MP-LEVER-TYPE3 with the handle instantiating the pivot role. The lever is instantiated in two of these functions, but the difference in how and where the force is applied changes how it is instantiated and, thus, what subclass is used to represent the function. Each of the last three functions is represented using MP-

CONTAINER, since the soil is in contact with the container’s inside surface.

Shovel Pragmatics

The shovel has two design-intended uses: (1) create holes, and (2) transport material. Hole creation is an SOP type plan, and makes use of the cutting edge on the shovel container boundary, while transport is a CROM type plan and makes use of the container itself.

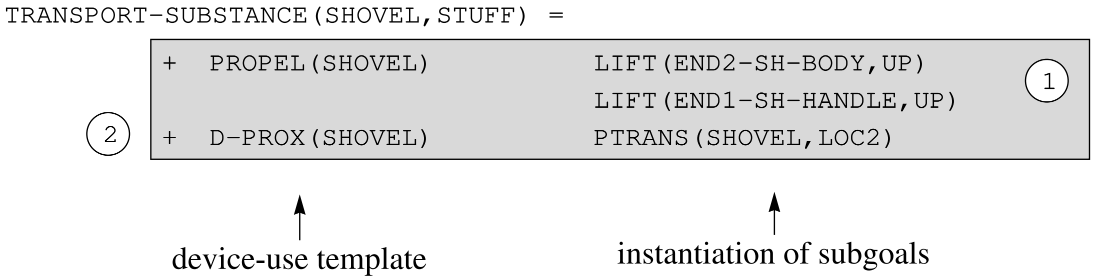
The hole-digging use is shown in Fig. A.17. Being a large device, the shovel requires control in more than one location by more than one appendage. In this application, the hand(s) are placed on the handle and a foot is placed on the edge of the scoop (at 2 in the figure). A hole can also be created by placing a second hand on the shovel body, depending on the nature of the shovel and stuff. The grasping actions are not noted as being either sequential or simultaneous, since the shovel can be used either way. Another new item is that the applied force is also located at two regions. The foot presses the container edge into the stuff while the hands push against the handle (1). Depending on the shovel and stuff, leverage is required to create a hole, in which case the device chosen must be used in such a way that it instantiates MP-LEVER-TYPE1. All digging plans require the application of an object which instantiates MP-BLADE (2).



```
CREATE-HOLE(SHOVEL,STUFF) =

      FIND(SHOVEL)             I-MECH(SHOVEL) +
                               I-WEDGE(SHOVEL) +
                               PTRANS(LOC(SHOVEL))
   +  D-CONT(SHOVEL)           GRASP(END1-HANDLE)
                               GRASP(CONTAINER)
   +  FIND(STUFF)              PTRANS(LOC(STUFF))
   +  I-PREP(SHOVEL)           D-REST(BOUNDARY,STUFF)
   +  PROPEL(SHOVEL)           PRESS(END2-BODY,STUFF),        ①
                               PUSH(END1-HANDLE,DOWN)

      I-MECH(SHOVEL)           MP-LEVER(END1-HANDLE,END2-BODY,EDGE),
      I-WEDGE(SHOVEL)          MP-BLADE(END1,EDGE)
   +  GRASP(END1-HANDLE)       HOLD(HAND,END1-HANDLE)         ②
      GRASP(BODY)              HOLD(HAND,BODY)
   +  D-REST(BOUNDARY,STUFF)   PLACE(BOUNDARY,STUFF,ABOVE)
```

Figure A.17 Pragmatic representation of shovel hole creation.

The second application, that of transporting stuff with the shovel, is shown in Fig. A.18. Transport is a CROM type plan, since the control over the substance is the primary concern. Most of the subgoals in the device-use plan are redundant from the previous plan, so they have been left out of the illustration. There is no additional preparation of the stuff required. In this example, force is applied at both locations where the shovel is being controlled, and the DO action of the plan is a D-PROX rather than a PROPEL.



```
TRANSPORT-SUBSTANCE(SHOVEL,STUFF) =
      +  PROPEL(SHOVEL)        LIFT(END2-SH-BODY,UP)
                               LIFT(END1-SH-HANDLE,UP)
      +  D-PROX(SHOVEL)        PTRANS(SHOVEL,LOC2)
```

Figure A.18 Pragmatic representation of shovel substance transport.

A.1.3 Spoon

The spoon and shovel are very similar objects. The shovel depicted in Fig. A.9 is comprised of three components, whereas the spoon depicted in Fig. A.19 is a single component with three regions of the same names as those in the shovel.

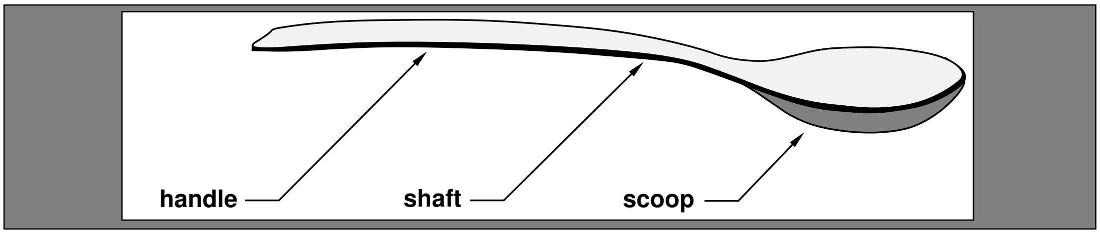


Figure A.19 Spoon illustration.

The spoon regions are associated with the same kinds of object primitives as they are for the shovel. The spoon is shorter than a shovel, but has a container or scoop at one end and a handle at the other. All components are scaled down in size.

Spoon Statics

The spoon and shovel have similar static representations, except that the spoon handle and body both comprise regions of a single component, while in the shovel they are represented with two components. An idealized spoon and its static device diagram are illustrated in Fig. A.20 and Fig. A.21.

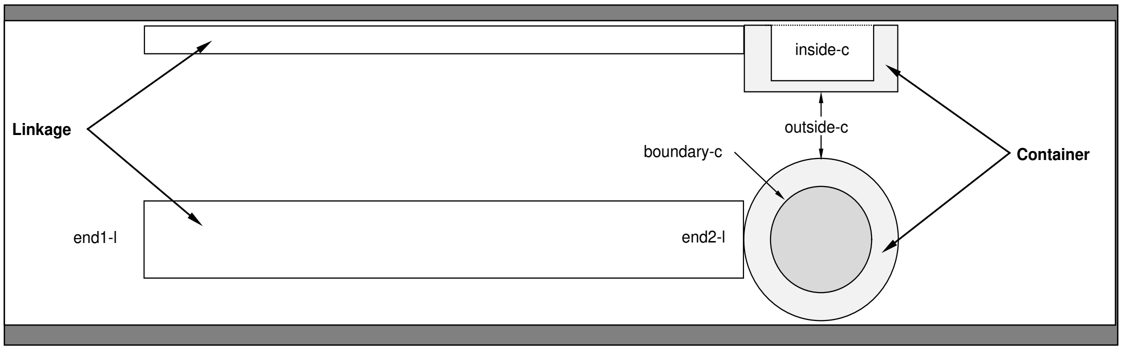


Figure A.20 Spoon object primitives and static device diagram.

The similarities between this figure and Fig. A.10, and between Fig. A.20 and Fig. A.11, are obvious. The two linkage-objects used to represent the shovel can be considered a single linkage-object in comparison to that of the spoon. In other respects, except size, the two objects have identical represention.

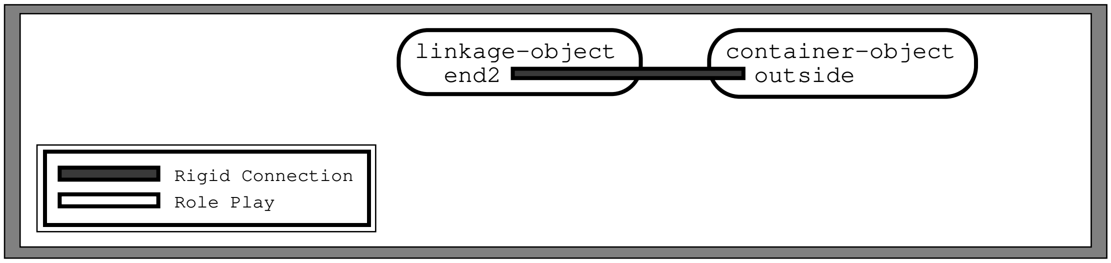


Figure A.21 Spoon static representation diagram.

Low-Level Spoon Dynamics

The behavioral sequences for the spoon are included here for continuity and comparison to the shovel diagrams. The only difference between the two devices at the behavioral level is magnitude: size of object, volume of containment, magnitude of applied and reacted forces..

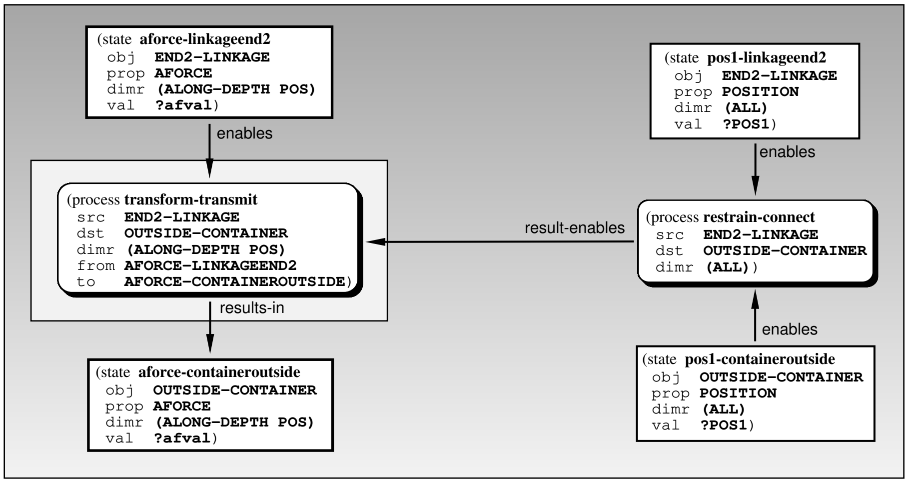


Figure A.22 Low-level spoon handle dynamics for lifting function

A spoon has the same kind of function as a shovel, and the representation of its behavior and function should capture their similarity.

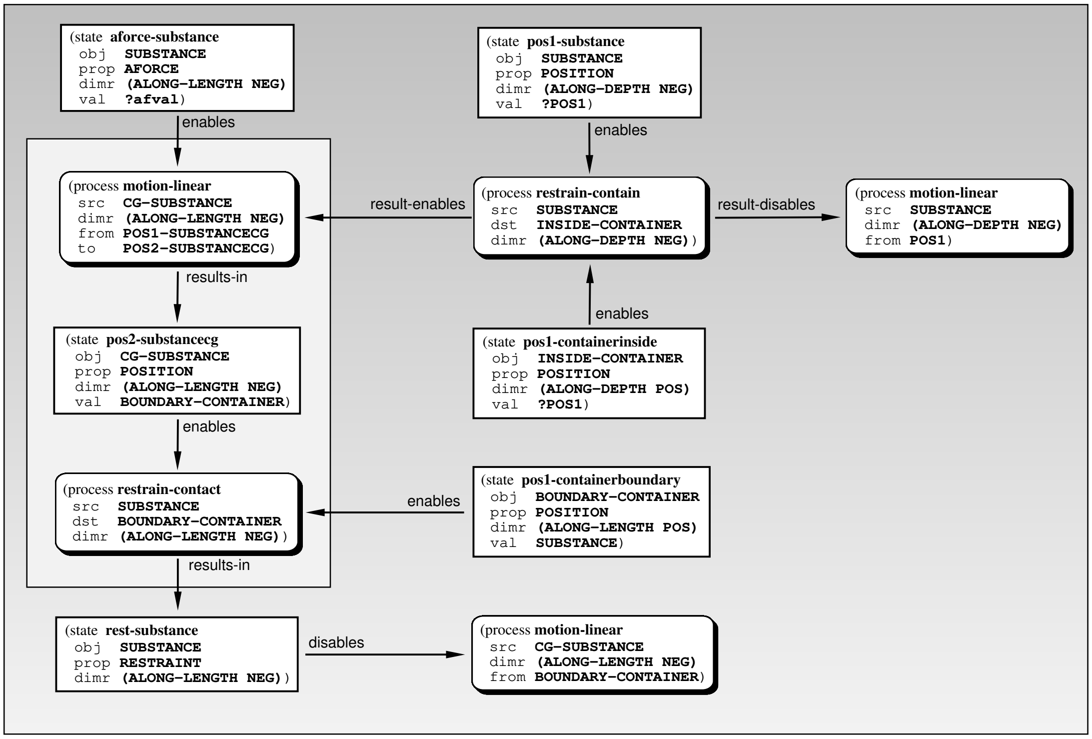


Figure A.23 Low-level spoon scoop dynamics for lifting function

High-Level Spoon Dynamics

Two spoon functions, dipping and lifting, are shown in Fig. A.24. The dipping function (1) has an identical MP-Diagram to that of the shovel cutting function, and performs an identical task.

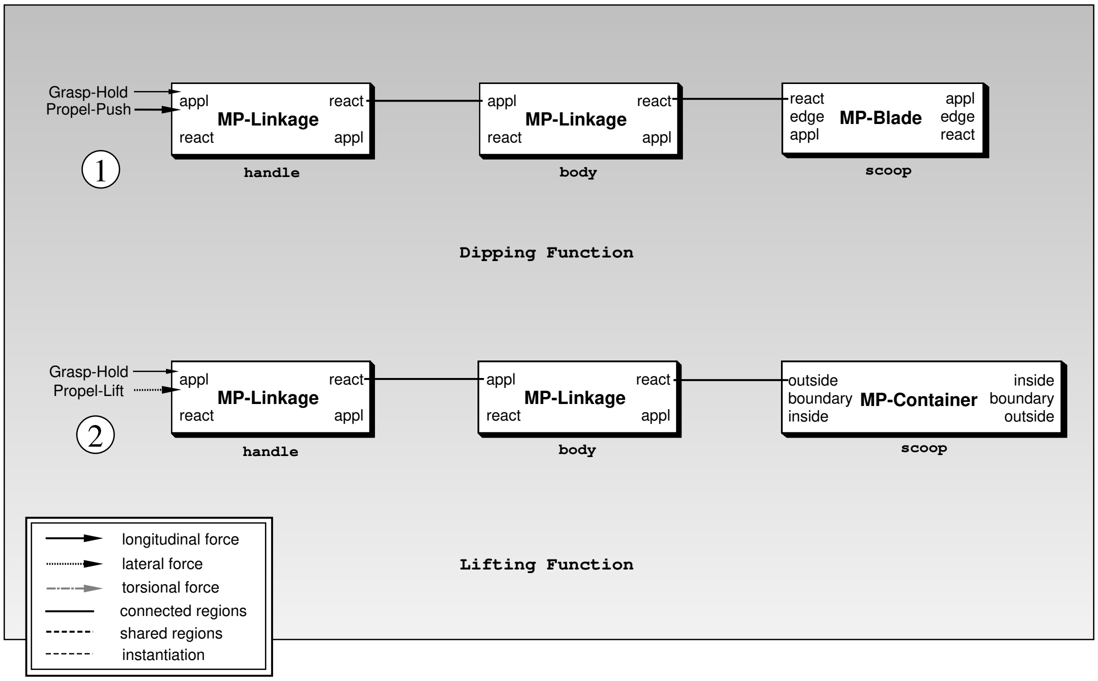


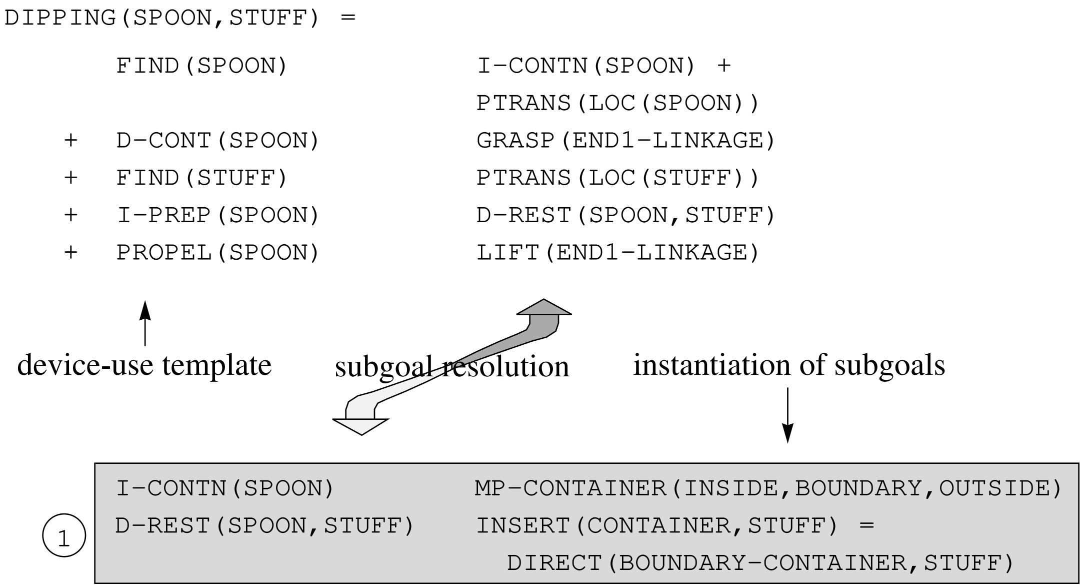
Figure A.24 MP-Diagrams for spoon functions.

Likewise, the lifting function (2) is represented using the same primitives as the shovel lifting function, the difference being the absence of the second applied force. Physically, the spoon and shovel differ in size more than composition, and these function representations illustrate their dynamic similarities. The digging and flinging shovel functions also have spoon counterparts, not illustrated in Fig. A.24.

Spoon Pragmatics

The spoon has two design-intended uses: (1) dipping, and (2) transport. The dipping application is illustrated in Fig. A.25. Dipping is an SOP plan like digging a hole with a shovel. Two new plans are involved in dipping. The spoon edge must be inserted into the stuff so that the inside of the spoon's container is in the stuff. This requires a second new plan, DI-

RECT, which points the edge of the spoon toward the stuff, before it is actually relocated.



```
DIPPING(SPOON,STUFF) =
      FIND(SPOON)          I-CONTN(SPOON) +
                           PTRANS(LOC(SPOON))
   +  D-CONT(SPOON)        GRASP(END1-LINKAGE)
   +  FIND(STUFF)          PTRANS(LOC(STUFF))
   +  I-PREP(SPOON)        D-REST(SPOON,STUFF)
   +  PROPEL(SPOON)        LIFT(END1-LINKAGE)
```

```
  I-CONTN(SPOON)         MP-CONTAINER(INSIDE,BOUNDARY,OUTSIDE)
① D-REST(SPOON,STUFF)    INSERT(CONTAINER,STUFF) =
                           DIRECT(BOUNDARY-CONTAINER,STUFF)
```

Figure A.25 Pragmatic representation of spoon dipping.

In both plans, the second argument in the plan represents an implicit direction (toward). For example, DIRECT(BOUNDARY-SPC,STUFF) means that the edge of the spoon container is aimed at the stuff. Since the edge is the entire container boundary, nothing is nor need be said about which portion of the spoon container is dipped into the stuff. The transport use of the spoon is identical to that of a shovel (i.e., a CROM type plan), and is not illustrated.

A.1.4 Claw Hammer

A claw hammer is illustrated in Fig. A.26. The claw hammer has a handle, a shaft, a head, and a claw. The claw is curved toward the handle and has a wedge-shaped notch in it

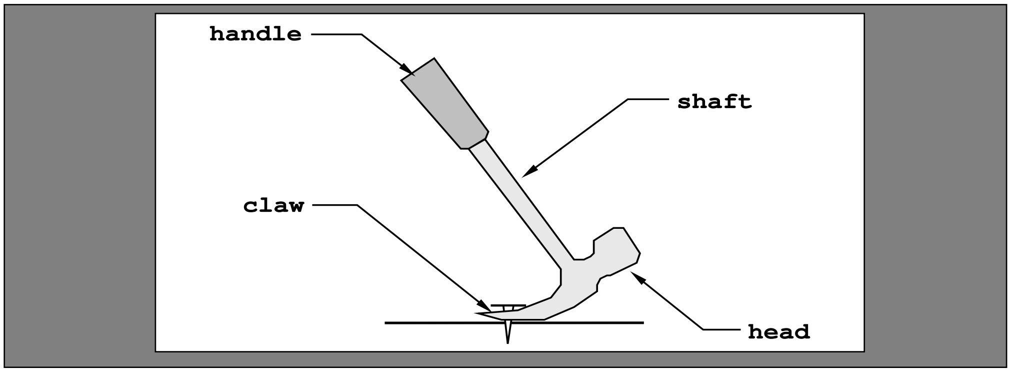


Figure A.26 Hammer illustration.

A claw hammer is used for pounding and prying.

Claw Hammer Statics

It has a long handle, and a head component which is heavy. One end of the head is flat, and the other is wedge-shaped and slotted. An idealized claw hammer and its static device diagram are depicted in Fig. A.27.

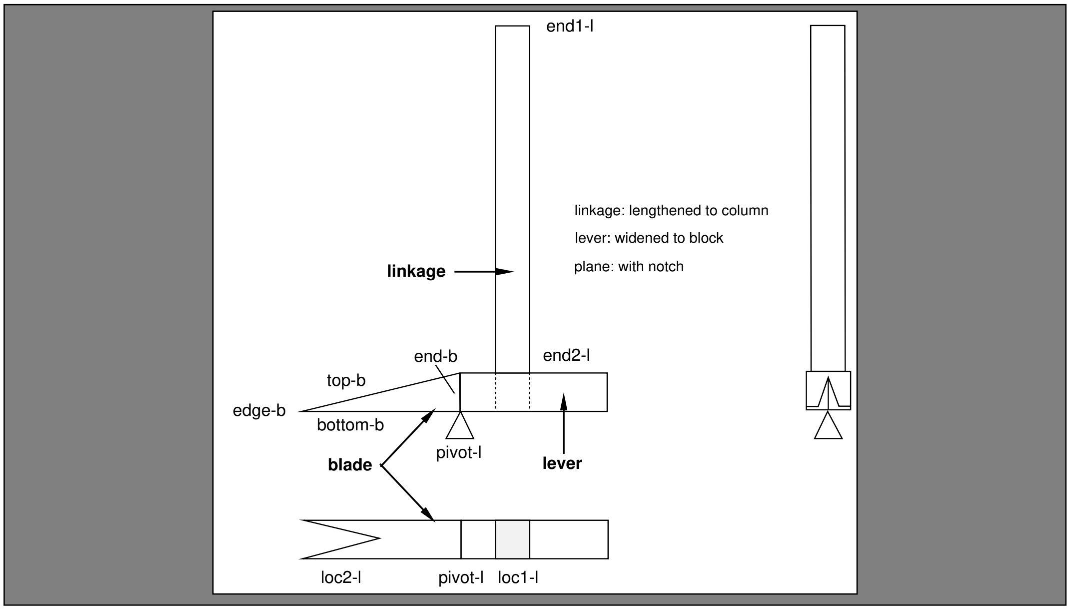


Figure A.27 Hammer object primitives and static device diagram.

The object is represented with a linkage-object, a lever-object, and a blade-object with an indentation for representing the slot. The lever-object `loc2` region is represented with a blade-object, and the lever-object pivot region represents the claw curvature. The accompanying SDD is shown in Fig. A.28.

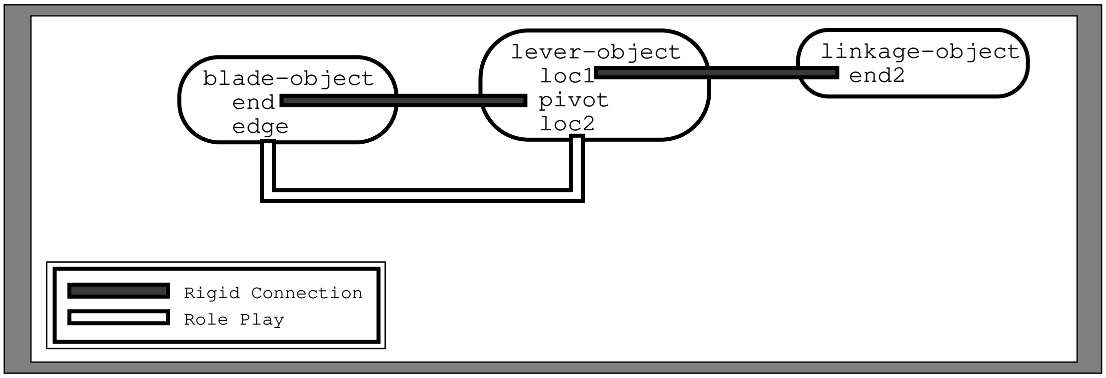


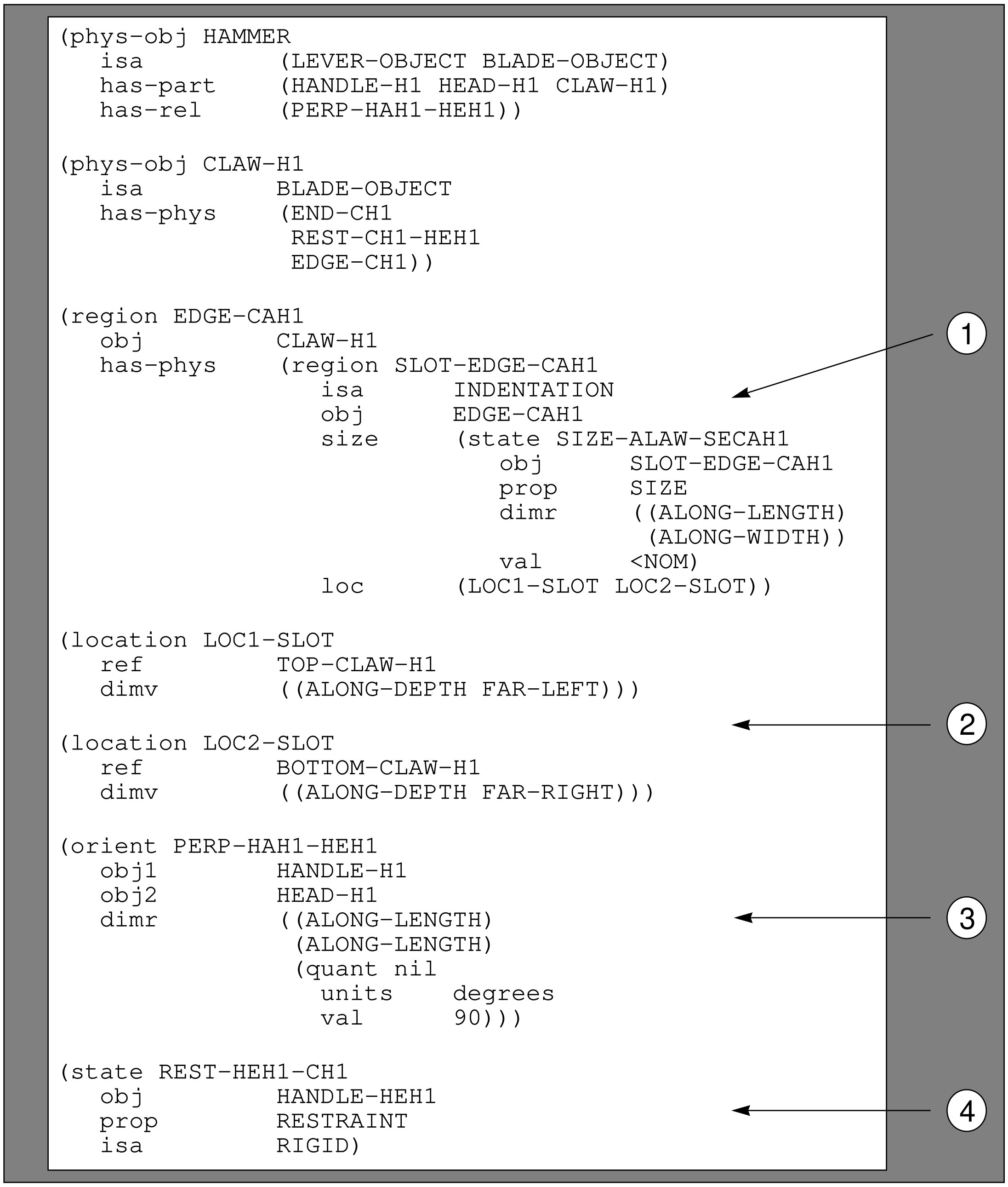
Figure A.28 Hammer static representation diagram.

In this figure, the `end` region of the plane-object is fixed to the `pivot` region of the lever-object. The `end2` region of the linkage-object is also fixed to the `loc1` region of the lever-object. A role-play link associates the `loc2` lever-object region with the `edge` blade-object region. The representation for the slot indentation on the claw, the perpendicular orientation of the handle and head, and the rigid connection between the head and the claw are represented in Fig. A.29. The blade-object edge region (`edge-cach1`, shown at item

1) has an indentation located at along its width and length. The slot is sized less than either dimension. The locations `loc1-slot` and `loc2-slot` place the slot ALONG-DEPTH at extremes (2), so that the slot goes all the way through the claw in this dimension.

The orientation between the hammer handle and head is represented with an orient knowledge structure (3). The two components, HANDLE-H1 and HEAD-H1, are oriented with respect to their longitudinal axes, which are 90 degrees offset. This relationship is represented with a single element size, where the dimensions are both ALONG-LENGTH.

Finally, the connection between HEAD-H1 and CLAW-H1 is represented with a fixed restraint state (4).



```
(phys-obj HAMMER
   isa            (LEVER-OBJECT BLADE-OBJECT)
   has-part       (HANDLE-H1 HEAD-H1 CLAW-H1)
   has-rel        (PERP-HAH1-HEH1))

(phys-obj CLAW-H1
   isa            BLADE-OBJECT
   has-phys       (END-CH1
                   REST-CH1-HEH1
                   EDGE-CH1))

(region EDGE-CAH1
   obj            CLAW-H1
   has-phys       (region SLOT-EDGE-CAH1
                     isa      INDENTATION
                     obj      EDGE-CAH1
                     size     (state SIZE-ALAW-SECAH1
                                 obj      SLOT-EDGE-CAH1
                                 prop     SIZE
                                 dimr     ((ALONG-LENGTH)
                                            (ALONG-WIDTH))
                                 val      <NOM)
                     loc      (LOC1-SLOT LOC2-SLOT))

(location LOC1-SLOT
   ref            TOP-CLAW-H1
   dimv           ((ALONG-DEPTH FAR-LEFT)))

(location LOC2-SLOT
   ref            BOTTOM-CLAW-H1
   dimv           ((ALONG-DEPTH FAR-RIGHT)))

(orient PERP-HAH1-HEH1
   obj1           HANDLE-H1
   obj2           HEAD-H1
   dimr           ((ALONG-LENGTH)
                   (ALONG-LENGTH)
                   (quant nil
                     units    degrees
                     val      90)))

(state REST-HEH1-CH1
   obj            HANDLE-HEH1
   prop           RESTRAINT
   isa            RIGID)
```

Figure A.29 Representation of claw hammer orientation and claw slot.

High-Level Claw Hammer Dynamics

Two hammer functions, those for prying and pounding, are depicted in Fig. A.30. The significance of this example is the representation of two different MP-LEVER classes with the same force type (lateral, at 1 and 3) and location (end1). In the prying function, the pivot is the head of the hammer (end2 of the handle), and MP-LEVER-TYPE1 represents the function (2). In the pounding function, the pivot is not a component of the device but, rather, the person wielding the device, and MP-LEVER-TYPE3 represents the function (4).

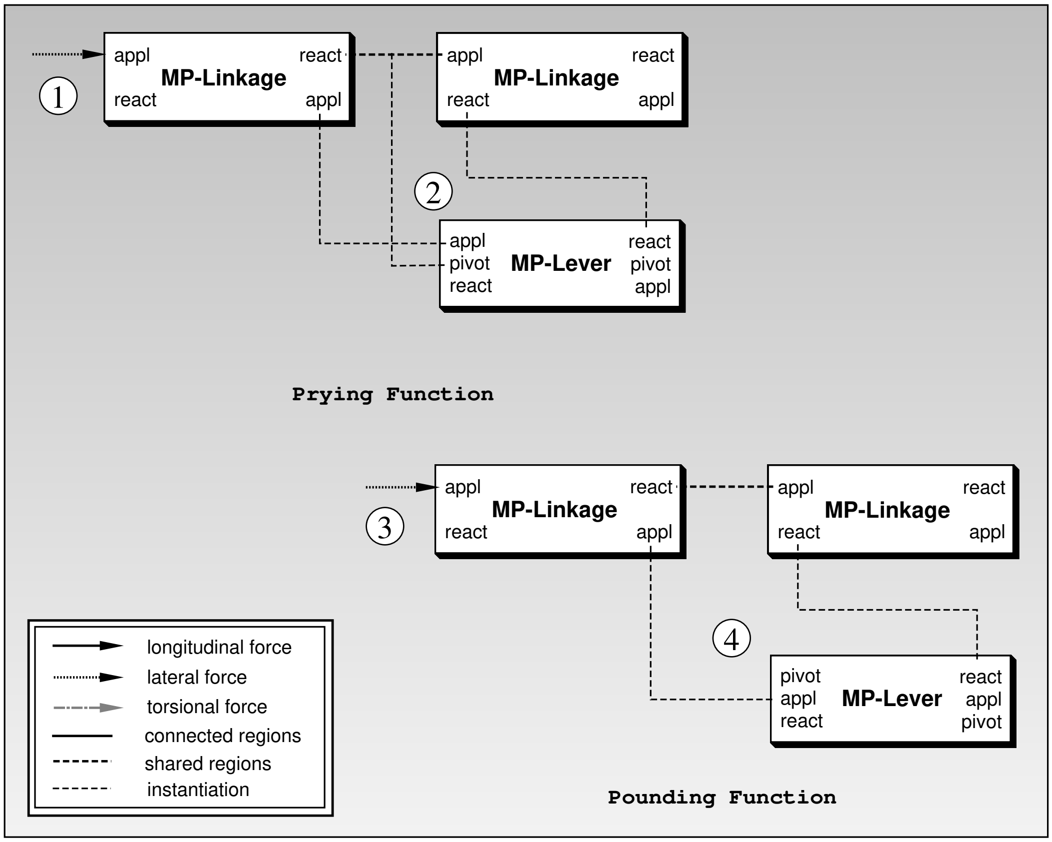


Figure A.30 MP-Diagrams for hammer functions.

Claw Hammer Pragmatics

The claw hammer has two design-intended uses: (1) nail pounding, and (2) nail removal. Nail pounding is a TFD plan, and nail removal is a CROM plan. Both plans support CROM plans for connecting and disconnecting objects. The pounding use is illustrated in Fig. A.31.

This use introduces the possibility of altering the device-use subgoals. No preparation of the hammer is required once control is effected, however, both an applied force, and movement of the hammer head into position, are required (i.e., PROPEL and D-PROX). The applied force is effected by swinging the hammer handle toward the object, where SWING is a plan which instantiates the MOTION-SWING process. The SWING plan also instantiates MP-LEVER-TYPE3 with the arm-hammer combination.

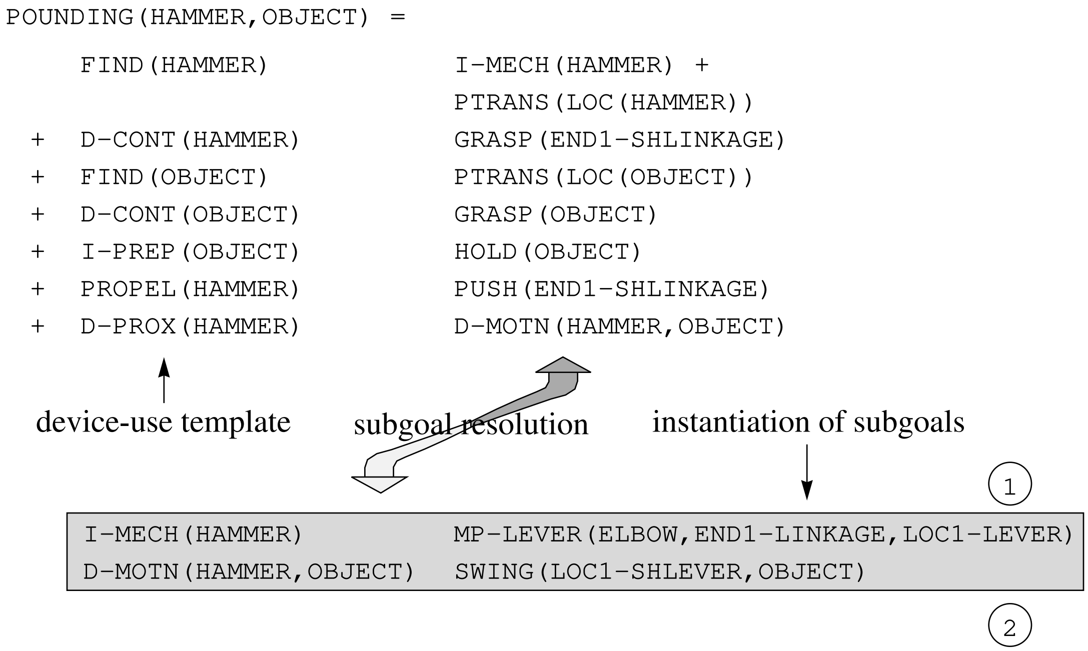


Figure A.31 Pragmatic representation of hammer pounding.

The nail removal plan is a confinement plan (CROM) which utilizes leverage, as illustrated in Fig. A.32.. In this figure, a new plan is introduced, called SLIDE Fig. A.32. In this figure, a new plan is introduced, called SLIDE To remove a nail with a claw hammer, the user (a) positions the hammer claw under the nail head, (b) slides the hammer toward the nail, (c) creates a fulcrum with the hammer head, and (d) pulls the hammer away from the nail. These tasks are represented with the behavioral delta goals D-REST and D-MOTN at (1) in Fig. A.32. SLIDE is used to describe the movement of the hammer after being positioned under the head of the nail, so as to wedge the nail in the claw slot (2). When the hammer is pulled, the direction is identified implicitly as LOC1-HLEVER (toward the head) to indicate away from the claw.

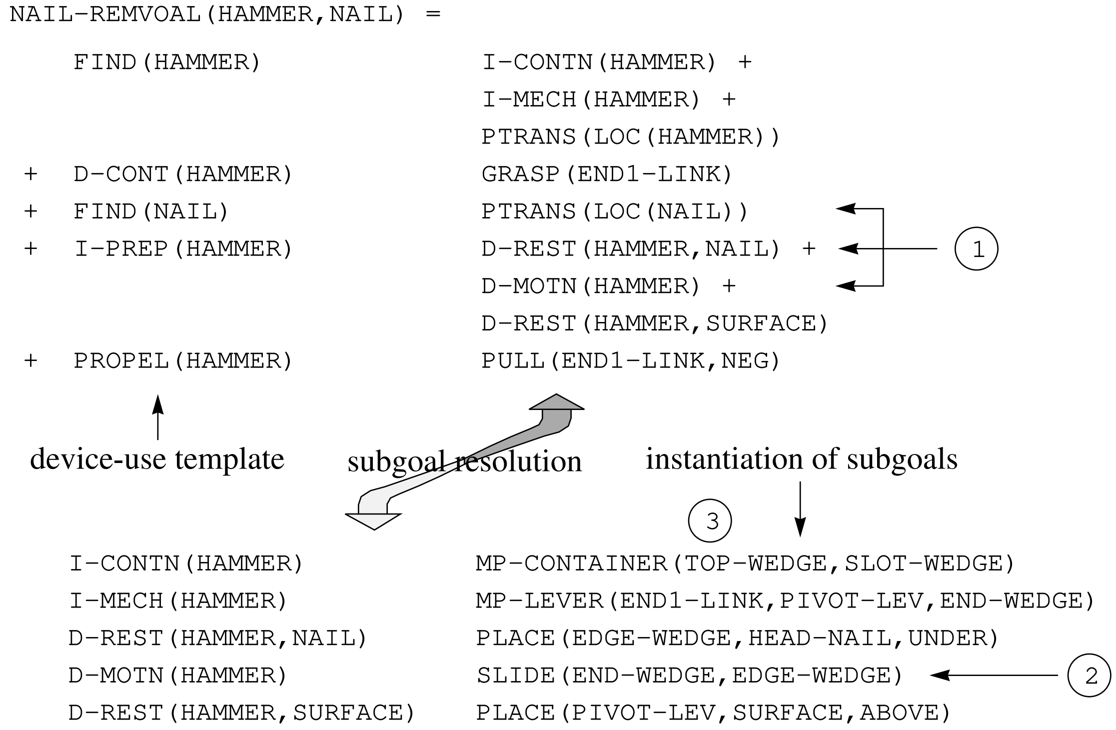


Figure A.32 Pragmatic representation of hammer nail removing.

A.2 Simple Compound Devices

Devices in which relative motion between components is possible are *compound* devices. Compound device dynamics are representationally significant because components can function independently. This has two affects on the representation and interpretation of compound devices. First, if a device component is not rigidly connected to other components, then the functions it can instantiate can be invoked without considering the other device components. Second, if a machine primitive participates in a device's function representation, then its function can be considered independently of the function in which it participates. In this section, three devices are presented which support these notions insofar as two components can be considered somewhat independently:

(1) scissors
(2) pliers
(3) door

A.2.1 Scissors

A pair of scissors is illustrated in Fig. A.33. A pair of scissors has two handles, a pivot, two blades and two pointed tips. The device is comprised of two components connected together at a common location. Like the shovel, the scissor components have a hole region called a handle.

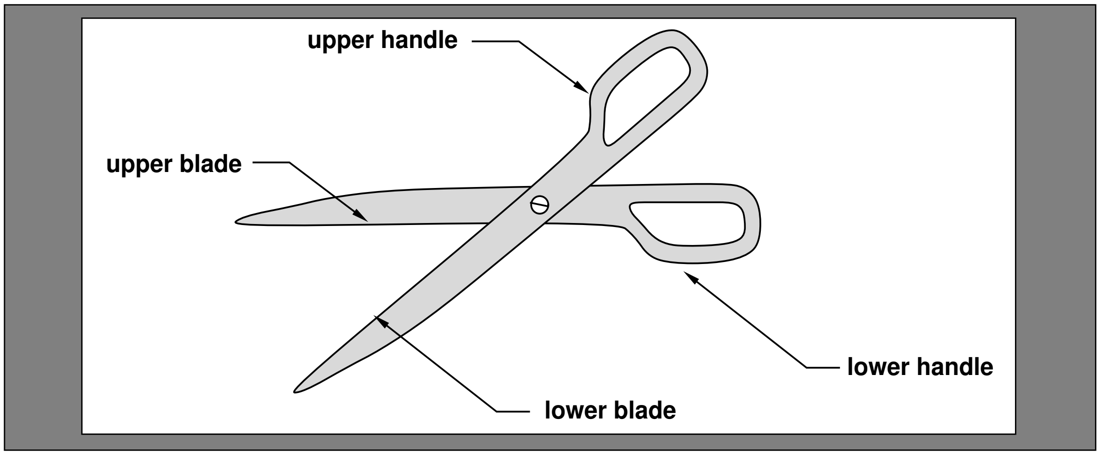


Figure A.33 Scissors illustration.

Scissors Statics

A pair of scissors is a simple compound device in which two components are pivoted to one another. The second component becomes the fulcrum for the first, so either can function independently of the other. Each component consists of a handle at one end and a pointed blade at the other. An idealized pair of scissors and its associated static device diagram are illustrated in Fig. A.34.

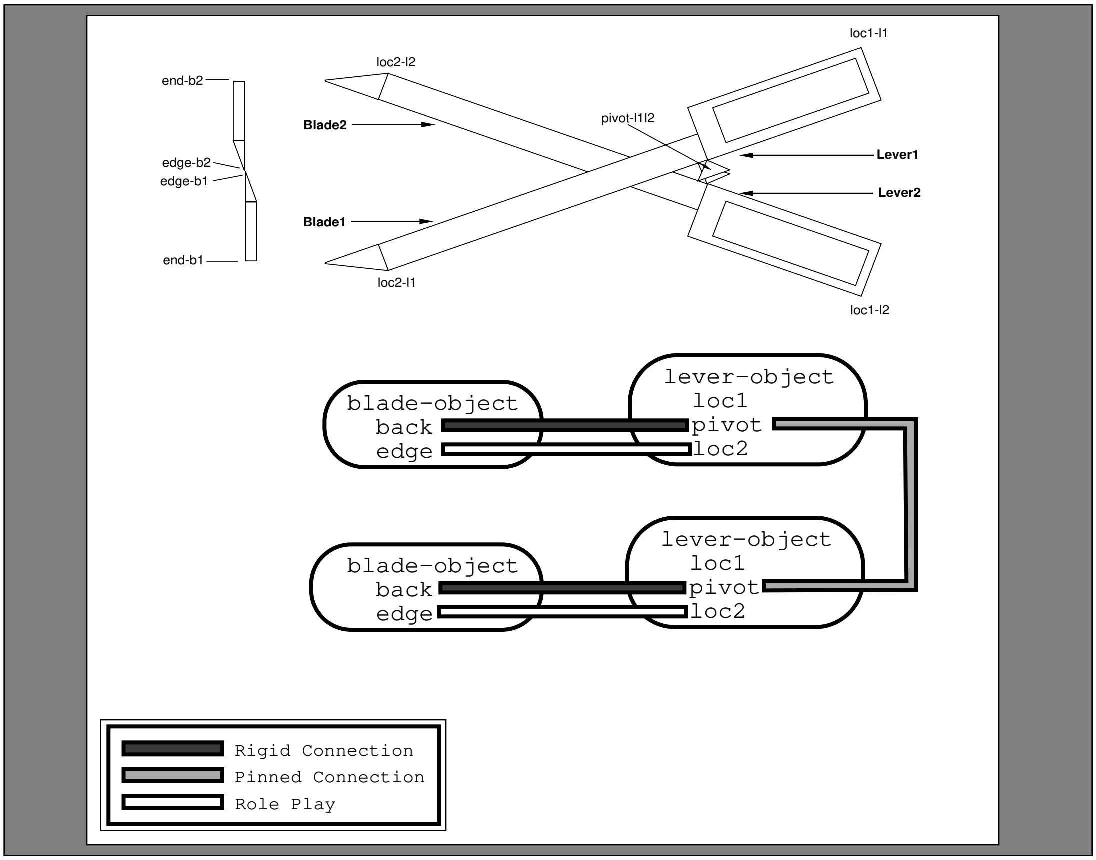


Figure A.34 Scissors object primitives and static device diagram.

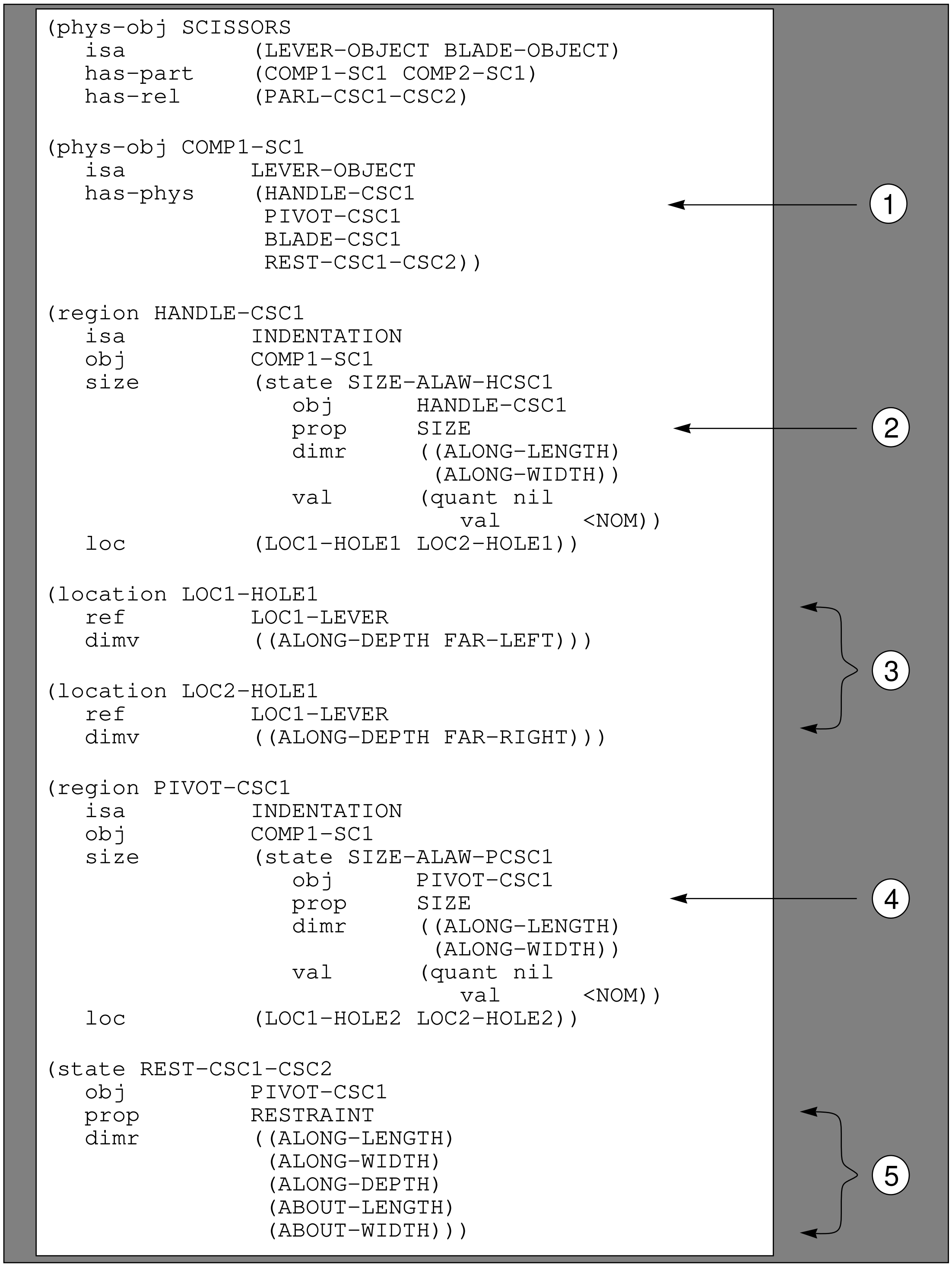
Because of the pivot in the static description, the components are represented with lever objects. Because of the shape of the components, they are represented as blade objects. Since each component can be a single device the lever and blade objects are depicted as rigidly connected with regions being shared. The shaded arc between the lever object pivot regions represents a pinned connection.

The components are each represented with a lever-object whose `loc2` region is instantiated with a blade-object. The `pivot` regions are shared between the two objects, as a pinned connection. The SDD for the device is shown in Fig. A.34.

In the figure, the `pivot` region for both lever-objects is represented as the other object. The handle regions, represented in Fig. A.35, are rigidly connected to the lever-object `loc1` regions. Similarly, the `back` region of either blade-object is rigidly connected to the lever-object `pivot` regions. The blade-object `edge` regions play the role of the lever-object `loc2` regions.

Fig. A.35 represents the scissor device (1) as a combination of two components in parallel (PARL-CSC1-CSC2). The handle of one of the scissor components (COMP1-SC1) is represented at (2). The handle is a hole region located at the lever-object `loc1` region (at the end). Like the shovel, the hole is approximately the size of the component, and goes through the component (3). The pivot region of COMP1-SC1 is represented at (4). It is located at the lever-object pivot region, and has similar characteristics to the handle hole. COMP1-SC1 is restrained (5) in all three translational dimensions and two of the three rotational dimensions. This is a pinned connection from Table 2.11.

The components of the scissor pair, unlike other devices represented so far, are connected in a nonrigid manner which enables independent motion of one component with respect to the other. This type of connection is characteristic of compound devices.



```
(phys-obj SCISSORS
   isa          (LEVER-OBJECT BLADE-OBJECT)
   has-part     (COMP1-SC1 COMP2-SC1)
   has-rel      (PARL-CSC1-CSC2)

(phys-obj COMP1-SC1
   isa          LEVER-OBJECT
   has-phys     (HANDLE-CSC1
                 PIVOT-CSC1
                 BLADE-CSC1
                 REST-CSC1-CSC2))

(region HANDLE-CSC1
   isa          INDENTATION
   obj          COMP1-SC1
   size         (state SIZE-ALAW-HCSC1
                   obj      HANDLE-CSC1
                   prop     SIZE
                   dimr     ((ALONG-LENGTH)
                             (ALONG-WIDTH))
                   val      (quant nil
                               val      <NOM))
   loc          (LOC1-HOLE1 LOC2-HOLE1))

(location LOC1-HOLE1
   ref          LOC1-LEVER
   dimv         ((ALONG-DEPTH FAR-LEFT)))

(location LOC2-HOLE1
   ref          LOC1-LEVER
   dimv         ((ALONG-DEPTH FAR-RIGHT)))

(region PIVOT-CSC1
   isa          INDENTATION
   obj          COMP1-SC1
   size         (state SIZE-ALAW-PCSC1
                   obj      PIVOT-CSC1
                   prop     SIZE
                   dimr     ((ALONG-LENGTH)
                             (ALONG-WIDTH))
                   val      (quant nil
                               val      <NOM))
   loc          (LOC1-HOLE2 LOC2-HOLE2))

(state REST-CSC1-CSC2
   obj          PIVOT-CSC1
   prop         RESTRAINT
   dimr         ((ALONG-LENGTH)
                 (ALONG-WIDTH)
                 (ALONG-DEPTH)
                 (ABOUT-LENGTH)
                 (ABOUT-WIDTH)))
```

Figure A.35 Partial static representation for scissors.

High-Level Scissors Dynamics

Two scissors functions are illustrated in Fig. A.36. The cutting function is initiated by applying lateral force to the handles.

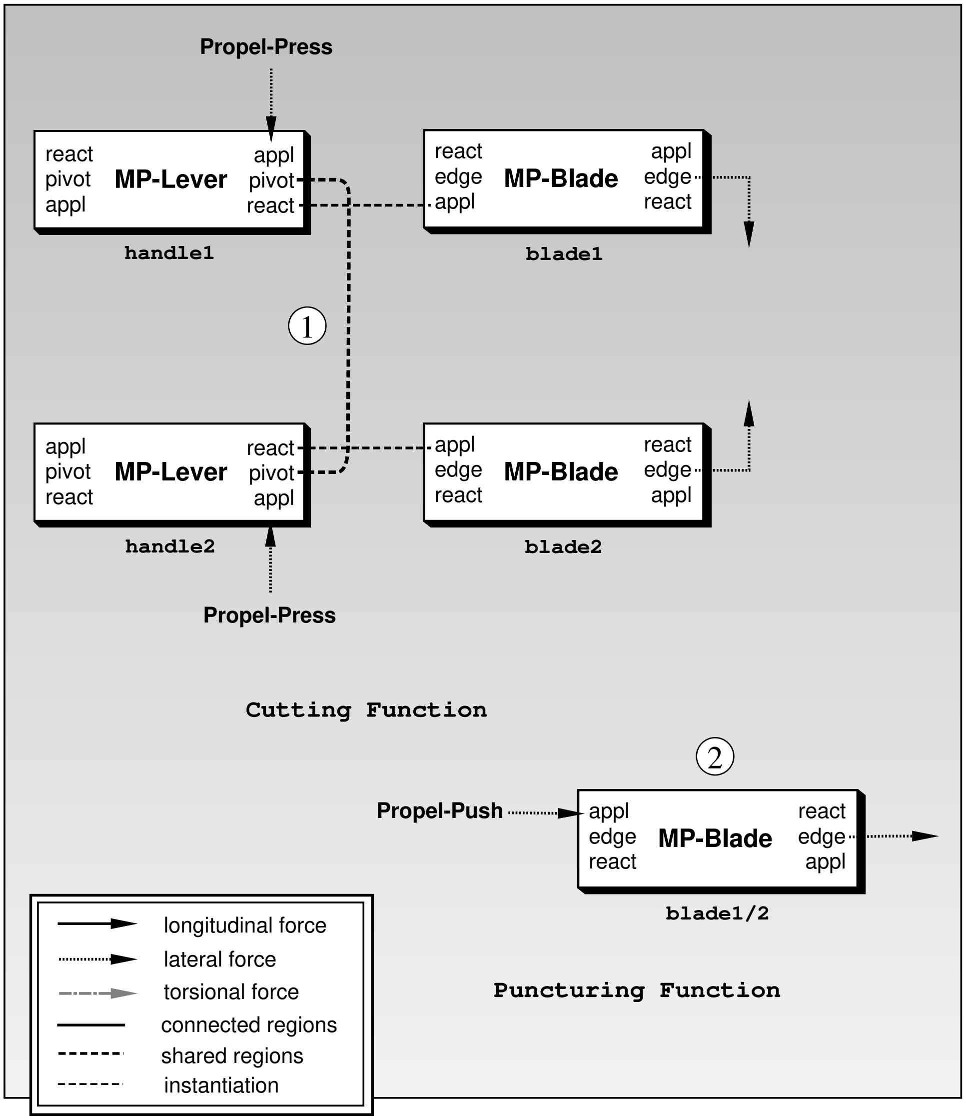
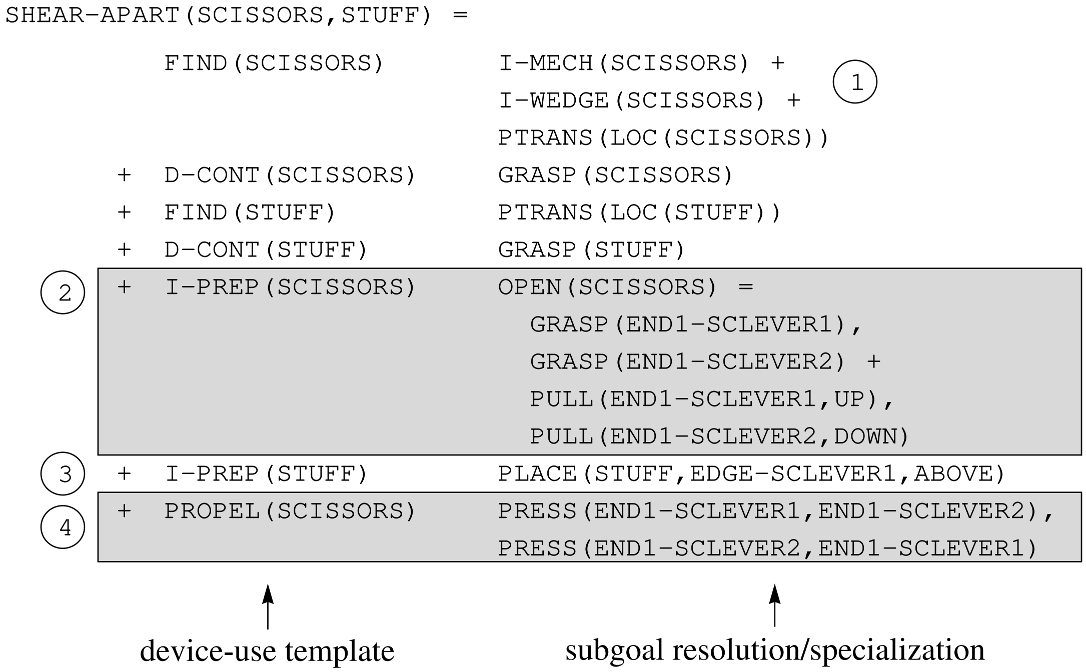


Figure A.36 MP-Diagrams for scissor functions.

The pivot role of MP-LEVER-TYPE1 is shared by the two lever objects (1), and is represented statically as a pinned connection. Thus either pair of MP-LEVER/MP-BLADE primitives can be applied for cutting if the other pair is unable to move. The second function (2) also shows that the entire device can be used as a generalized MP-BLADE, or that one component can instantiate MP-BLADE, for puncturing, if the applied force is longitudinal.

Scissors Pragmatics

The pair of scissors has two design-intended uses: (1) poking holes, and (2) shearing, both of which are SOP plans. As a simple compound device, using a pair of scissors requires the repetitive application of two subplans: opening, which enables the shear-apart function; and closing, which initiates the shear-apart function, as shown in Fig. A.37. The device is used by (a) opening the blades, (b) inserting the material between the blades, and (c) closing the blades on the material. As the blades impinge on the material, a cut is made which propagates along the length of the blade. The preparation (OPENing) of the scissors involves grasping the two handles and then pulling them apart, as shown in the shaded portion of the figure, at (2). These actions are represented individually, so that their directions can be shown. The UP and DOWN directions are not very effective in this example, since neither UP nor DOWN is at all necessary. The insertion of material is labeled (3). Unlike the directions shown for pulling the handles apart, when the handles are pressed together (at 4), the directions are well-defined, since they are implicitly defined by the other handle.



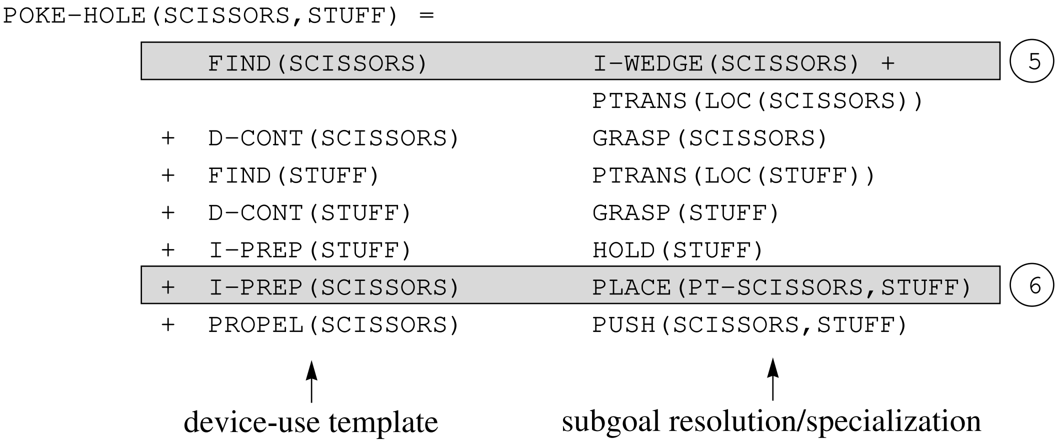
```
SHEAR-APART(SCISSORS,STUFF) =

         FIND(SCISSORS)       I-MECH(SCISSORS) +
                              I-WEDGE(SCISSORS) +     ①
                              PTRANS(LOC(SCISSORS))
       + D-CONT(SCISSORS)     GRASP(SCISSORS)
       + FIND(STUFF)          PTRANS(LOC(STUFF))
       + D-CONT(STUFF)        GRASP(STUFF)
 ②     + I-PREP(SCISSORS)     OPEN(SCISSORS) =
                                GRASP(END1-SCLEVER1),
                                GRASP(END1-SCLEVER2) +
                                PULL(END1-SCLEVER1,UP),
                                PULL(END1-SCLEVER2,DOWN)
 ③     + I-PREP(STUFF)        PLACE(STUFF,EDGE-SCLEVER1,ABOVE)
 ④     + PROPEL(SCISSORS)     PRESS(END1-SCLEVER1,END1-SCLEVER2),
                              PRESS(END1-SCLEVER2,END1-SCLEVER1)
```

Figure A.37 Pragmatic representation scissors shearing.

In this example, both the comma (,) and plus (+) notation are used to describe simultaneous and sequential events, respectively As mentioned above, the instrumental mechanical goals (1) assure retrieval of a device which can both apply the force and cut the material.

The second use of a pair of scissors, to poke a hold in some stuff, is illustrated in Fig. A.38. In this latter example, the application is very straight forward, because the pair of scissors is being used as a simple device. The poke-hole plan makes use of the scissors blades being pointed and not on the device's ability to produce mechanical advantage. The one distinction is that the device must be pointed in the correct direction, toward the stuff

(6), in order to puncture it.



```
POKE-HOLE(SCISSORS,STUFF) =
          FIND(SCISSORS)          I-WEDGE(SCISSORS) +        5
                                  PTRANS(LOC(SCISSORS))
      +   D-CONT(SCISSORS)        GRASP(SCISSORS)
      +   FIND(STUFF)             PTRANS(LOC(STUFF))
      +   D-CONT(STUFF)           GRASP(STUFF)
      +   I-PREP(STUFF)           HOLD(STUFF)
      +   I-PREP(SCISSORS)        PLACE(PT-SCISSORS,STUFF)   6
      +   PROPEL(SCISSORS)        PUSH(SCISSORS,STUFF)
```

Figure A.38 Pragmatic representation scissors poking.

A.2.2 Pliers

A pair of pliers is illustrated in Fig. A.39. Like the scissors, the pliers is comprised of two components pinned together. There are two primary static differences between the devices. First, instead of blades the plier components have jaws at one end.

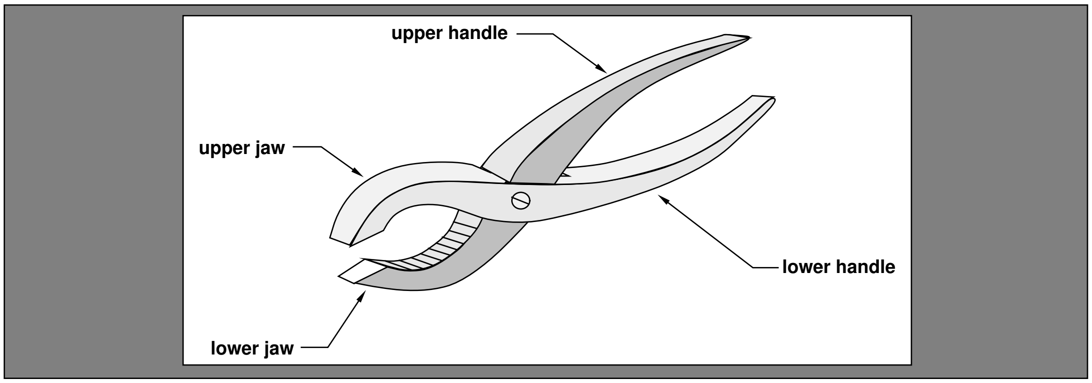


Figure A.39 Pliers illustration.

Second, the plier jaws are opposed rather than parallel.

Pliers Statics

The idealized version of the pliers is shown in Fig. A.40.

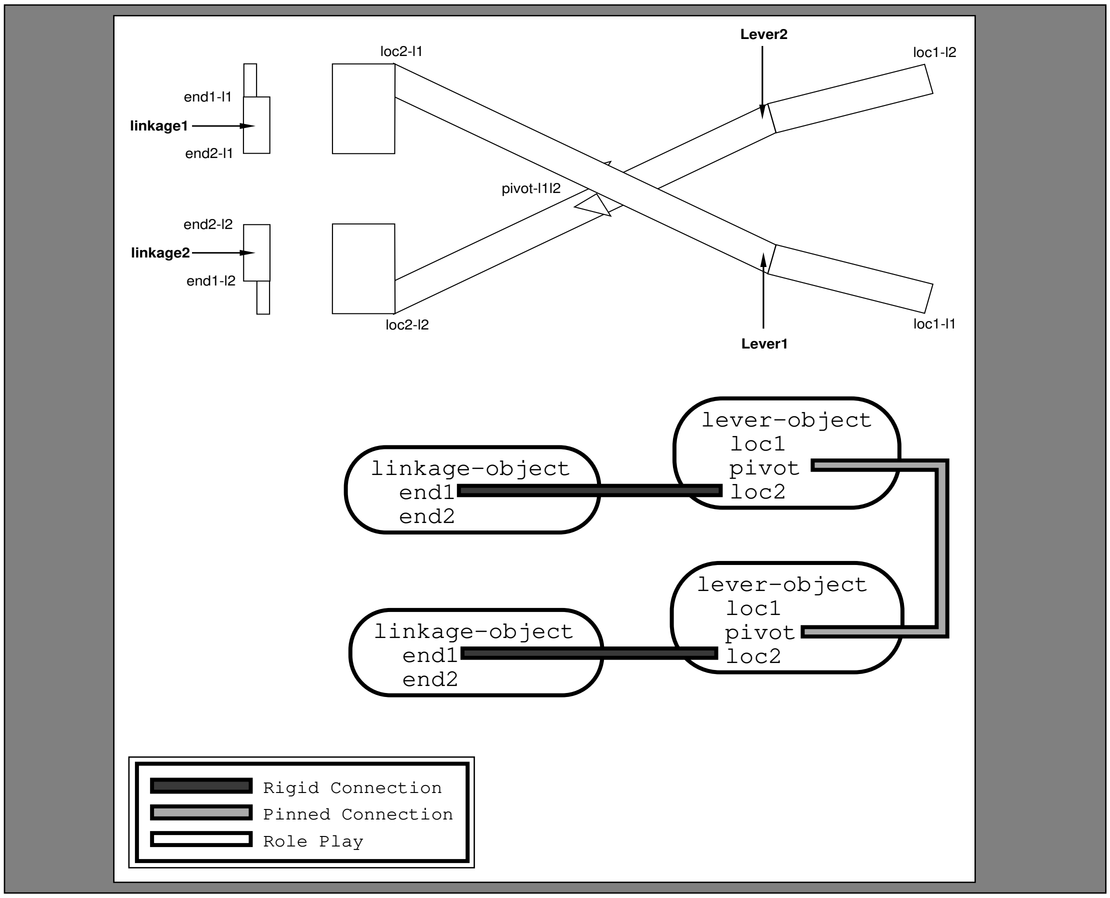
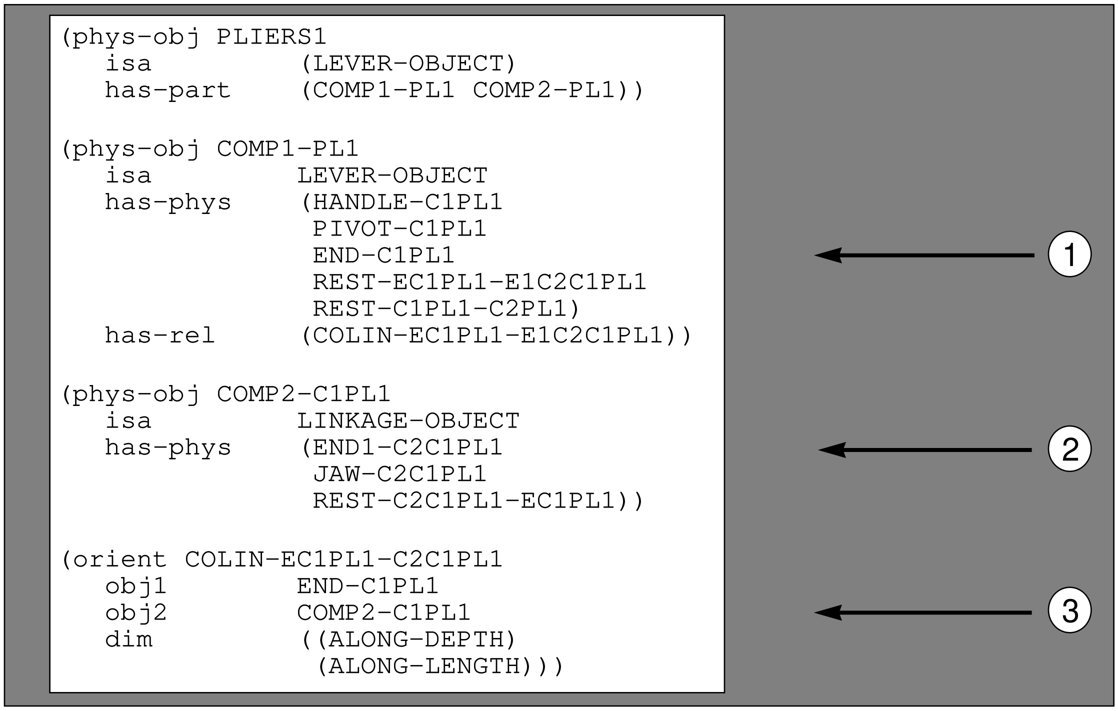


Figure A.40 pliers object primitives. Two levers and two linkages.

Each component is represented as a lever-object rigidly connected to a linkage-object, as shown in the accompanying SDD of Fig. A.40.

In Fig. A.40, the linkage-object is used to represent the plier component jaw region and is rigidly connected to the lever-object `loc2` region. The linkage-object and lever-object are colinear in orientation, but offset so that the jaws of the different components can make contact. This orientation difference, and the simplification of the handle representation (no holes for hands), are the primary static differences between the pliers and scissors. The orientations are represented in Fig. A.41 below.



```
(phys-obj PLIERS1
   isa          (LEVER-OBJECT)
   has-part     (COMP1-PL1 COMP2-PL1))

(phys-obj COMP1-PL1
   isa          LEVER-OBJECT
   has-phys     (HANDLE-C1PL1
                 PIVOT-C1PL1
                 END-C1PL1
                 REST-EC1PL1-E1C2C1PL1
                 REST-C1PL1-C2PL1)
   has-rel      (COLIN-EC1PL1-E1C2C1PL1))

(phys-obj COMP2-C1PL1
   isa          LINKAGE-OBJECT
   has-phys     (END1-C2C1PL1
                 JAW-C2C1PL1
                 REST-C2C1PL1-EC1PL1))

(orient COLIN-EC1PL1-C2C1PL1
   obj1         END-C1PL1
   obj2         COMP2-C1PL1
   dim          ((ALONG-DEPTH)
                 (ALONG-LENGTH)))
```

Figure A.41 Static representation for plier jaw orientation.

To represent the orientation between the lever-object and the linkage-object, the loc2 lever-object region and the linkage-object dimensional axes must be represented. The connection between the two objects provides a dimension of comparison. The two plier components each have a handle, a pivot, and a jaw region (1), and restraint states with their respective linkage-object and the other lever-object. The linkage-object (COMP2-C1PL1) is represented (2) with an end region and a jaw (end) region. The orientation is defined (3) between the lever-object ALONG-DEPTH dimension and the linkage-object ALONG-LENGTH dimension. The notion of colinearity arises when the two objects are connected in the proximity of the orientations definition. That is why both colinear and parallel orientations can be represented with similar instantiations.

A pair of pliers is used to hold or restrain objects and employ mechanical advantage to do so. Pliers are physically similar to a pair of scissors, the difference being that the plier components flat rather than sharp.

High-Level Pliers Dynamics

The clamping pliers function is depicted in Fig. A.42. The pliers and scissors are very similar physically, as were the spoon and shovel, and their MP-Diagrams demonstrate that their functions are also similar. The physical difference between the shape of the scissor blades and the plier hammers shows up in the whether MP-BLADE is used to represent the function or whether MP-LINKAGE is used to represent the function. In both devices/functions, the pivot region is shared by the two components.

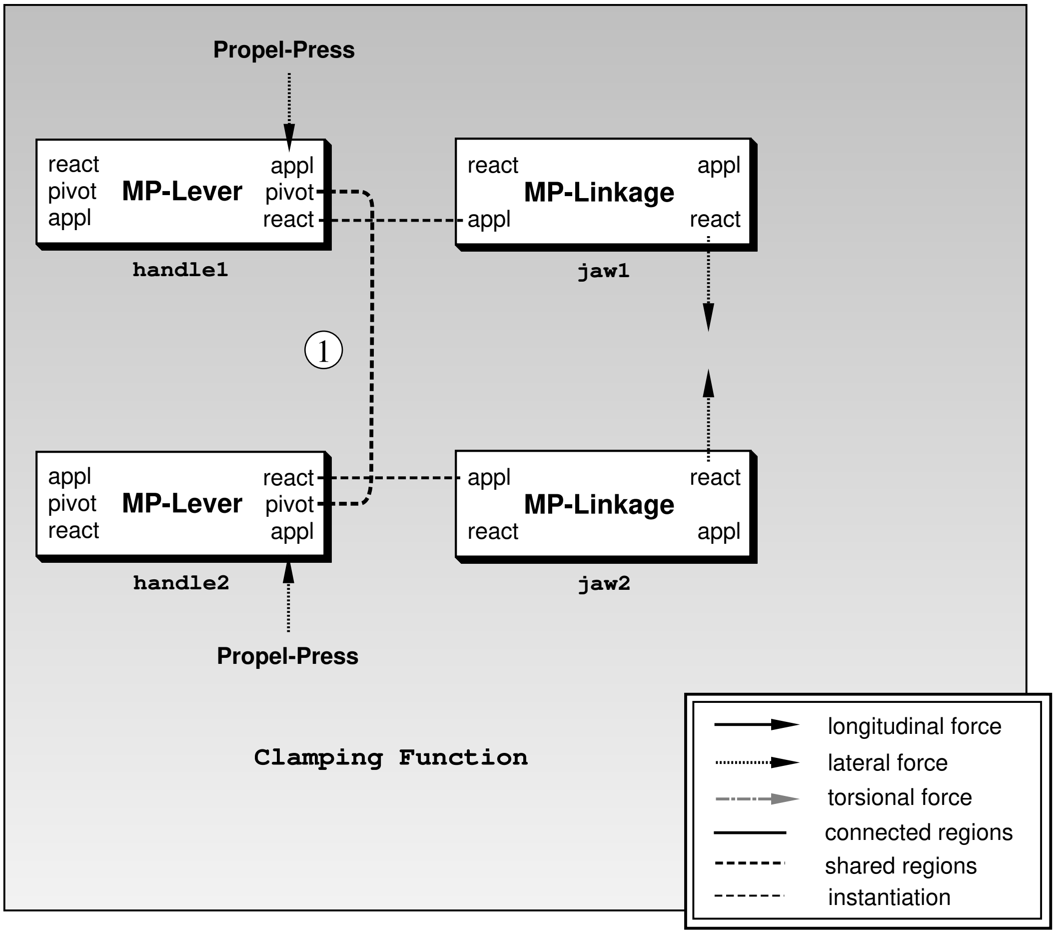


Figure A.42 MP-Diagrams for plier functions.

Pliers Pragmatics

The pair of pliers has a single design-intended use: (1) gripping objects together, as shown in Fig. A.43. The gripping plan is a CROM plan used to confine and control an object. Despite their static and dynamic similarities, a pair of pliers and a pair of scissors are used to achieve different ends. These differences show up in the types of plans they are intended to address, and in the types of machines they instantiate in enabling these plans.The scissors use is to separate objects whereas the pliers use is to control objects. In Fig. A.43, the difference is identified by the I-CONTN goal, as opposed to the I-WEDGE goal for the scissors. Both devices restrain their object (at 3), however, the application of force is used differently in each. In other respects the plans are identical.

```
GRIP-TOGETHER(PLIERS,STUFF) =
 ①      FIND(PLIERS)          I-MECH(PLIERS) +
                              I-CONTN(PLIERS) +
                              PTRANS(LOC(PLIERS))
     +  D-CONT(PLIERS)        GRASP(PLIERS)
     +  FIND(STUFF)           PTRANS(LOC(STUFF))
     +  D-CONT(STUFF)         GRASP(STUFF)
 ②   +  I-PREP(PLIERS)        OPEN(PLIERS) =
                                GRASP(END1-PLLEVER1),
                                GRASP(END1-PLLEVER2) +
                                PULL(END1-PLLEVER1),
                                PULL(END1-PLLEVER2)
 ③   +  I-PREP(STUFF)         PLACE(STUFF,END2-PLLINKAGE1,ABOVE)
 ④   +  PROPEL(PLIERS)        PRESS(END1-PLLEVER1),
                              PRESS(END1-PLLEVER2)
```

device-use template

instantiation of subgoals

Figure A.43 Pragmatic representation of pliers gripping.

A.2.3 Door

A door physically blocks the motion of objects into or out of a container.

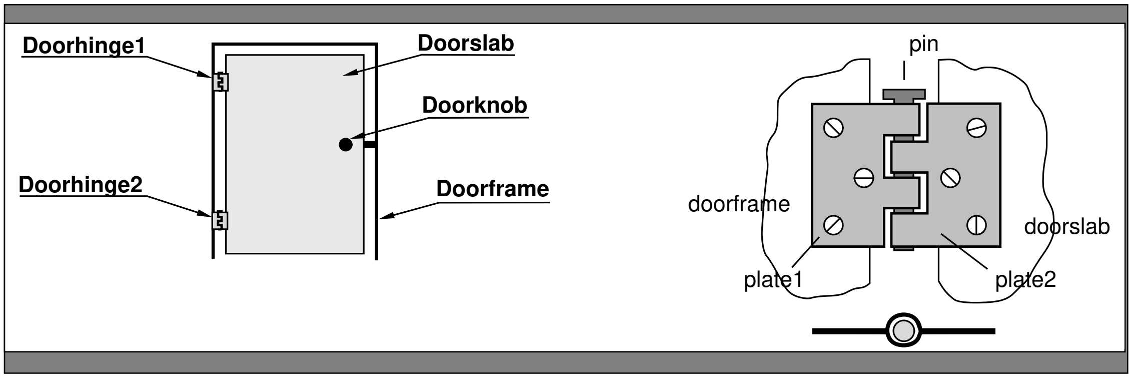


Figure A.44 Door amd doorhinge illustration.

Door Statics

A simple door is comprised of four components: a piece which moves, called a doorslab, a doorframe, and two hinges. When the door is open, motion past the door is enabled. When the door is closed it isn't. The doorslab is generally a solid object which functions to transmit force and motion, so the doorhinges are significant components in the device. An idealized doorhinge and its static device diagram is shown in Fig. A.45.

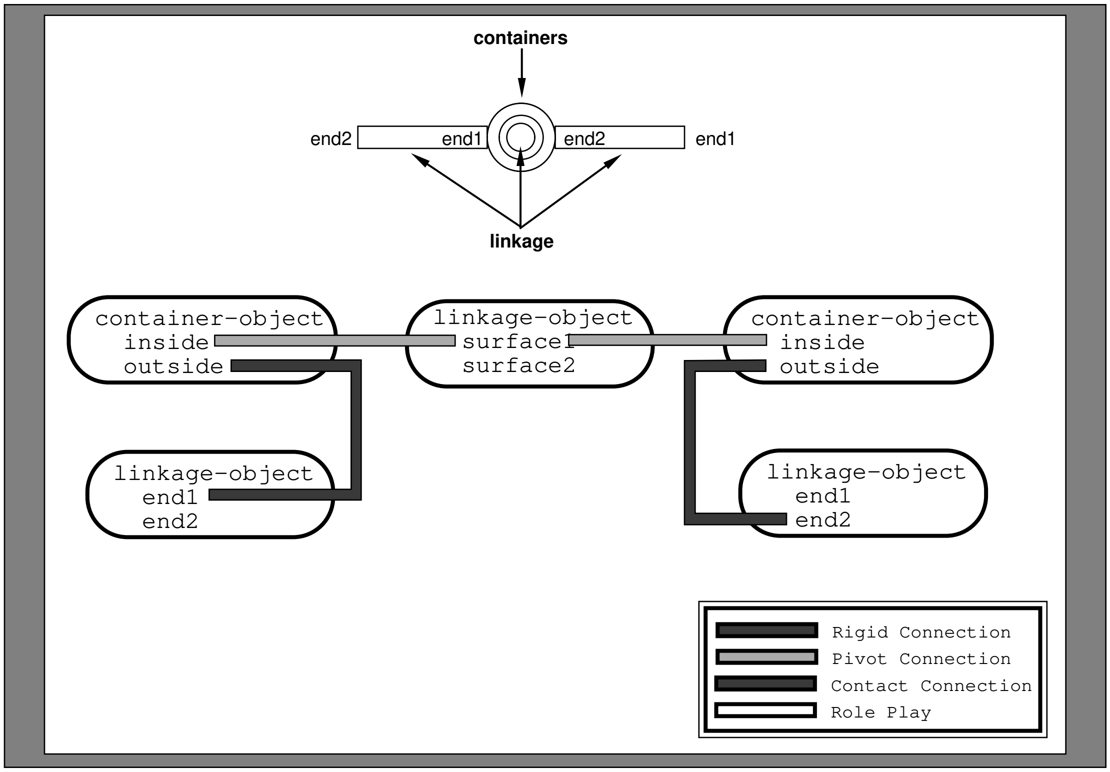


Figure A.45 Doorhinge object primitives and static device diagram.

The doorhinge is comprised of three components: two plates and a pin. The pin is contained by containers from each plate, which restricts translational motion of each plate, but allows rotational motion of each plate. Thus the connection is represented as a pinned connection.

High-Level Door Dynamics

The representation of doorhinge function is shown in Fig. A.46.

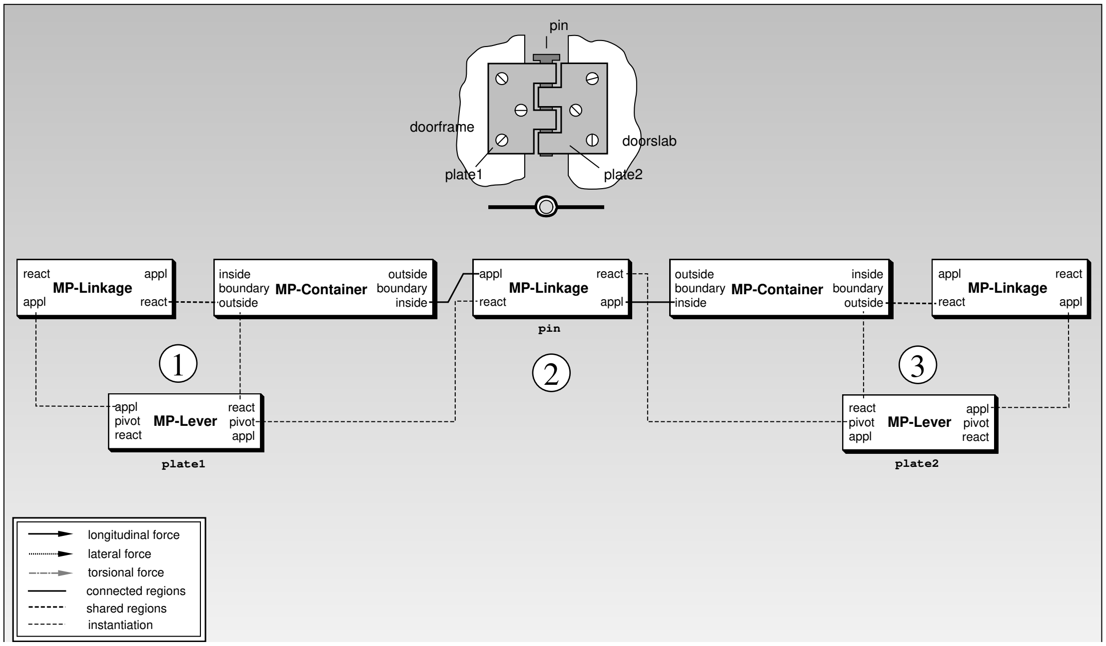


Figure A.46 MP-Diagram for doorhinge function.

Alone each doorhinge plate function is represented with MP-LINKAGE (at 1 and 3). However, by sharing the common hinge pin (PIN, at 2), the two plates (PLATE1 and PLATE2) instantiate mutually dependent MP-LEVER-TYPE2 devices. Without the other plate, neither hinge plate's motion is contrained by the hinge pin and cannot function as a MP-LEVER. However, with this caveat, i.e., as long as the plates remain pinned, both PLATE1 and PLATE2 can be considered as levers independently. This representation can be used in the overall door dynamic representation by replacing the linkage/container object with the lever object. An idealized version of the simple door and its static device diagram are illustrated in Fig. A.47. In this figure, the doorhinge is idealized to a single lever object, since the second plate is rigidly connected to the doorframe.

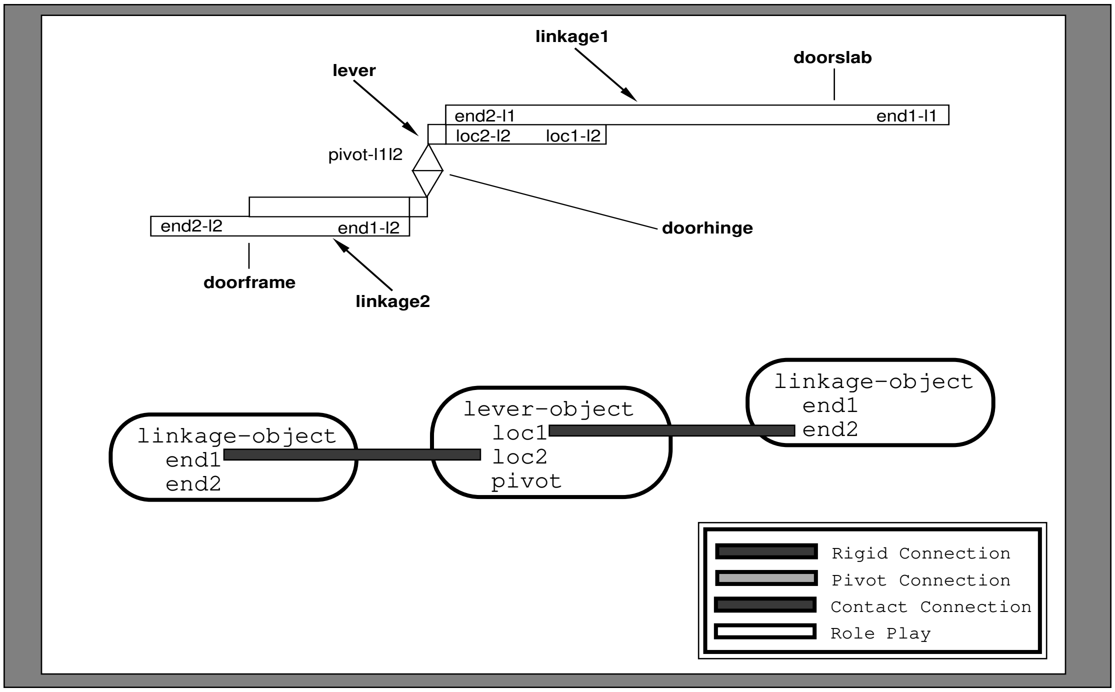


Figure A.47 Door object primitives and static device diagram.

The door opening function illustrated in Fig. A.48 presents both doorhinge plates, and also represents the doorknob function. In this figure, the doorknob function of releasing the latch is represented with MP-WHEEL-AXLE-TYPE2. This function is initiated by turning the doorknob. When the latch is released, the door opening function is initiated by pulling the doorknob, in which case the force is transmitted directly to the doorslab, which is represented with MP-LINKAGE (2). The doorslab is connected to the hinges, so the resulting force is transformed via MP-LEVER (3) to the doorframe.The function of the slot in the doorframe is to restrict the motion of the door, so it is represented with MP-CONTAINER

(4).

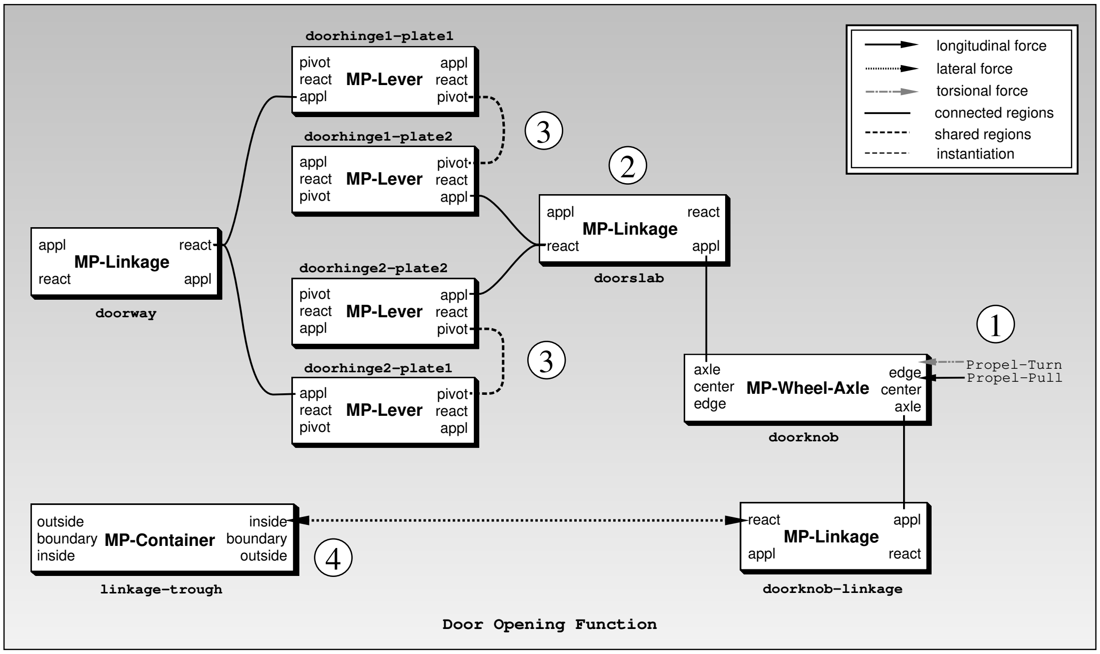


Figure A.48 MP-Diagram for door opening function.

Door Pragmatics

The door has two design-intended uses: (1) secure an environment, and (2) access an environment, both of which are containment (i.e., CROM) plans. Access makes use of the door opening function, while security and containment use the door closing function. The containment use plan is illustrated in Fig. A.50.

Access means that we activate movement of the door so as to create an opening through which objects may pass. This is accomplished by grabbing and turning the doorknob and then pushing the door. Because the door is part of the environment, the environment need not be found or controlled. The access-environment(door,environment) plan thus degenerates to a use(door) plan, but makes use of the DISENGAGE-LATCH plan, which is also a containment plan.

```
ACCESS-ENVIRONMENT(DOOR,ENVIRONMENT) =

              FIND(DOOR)              PTRANS(LOC(DOOR))
          +   D-CONT(DOOR)            GRASP(KNOB-DOORSLAB)
          +   I-PREP(DOOR)            DISENGAGE-LATCH(DOOR) =
                                        TURN(KNOB-DOORSLAB)
          +   PROPEL(DOOR)            PULL(KNOB-DOORSLAB)

                  ↑                               ↑
          device-use template     subgoal resolution/specialization
```

Figure A.49 Pragmatic representation of door environmental access.

Door use for environmental containment is illustrated in Fig. A.50. Environmental containment means that by closing the door, the means of passage is removed, disabling entry and exit. This plan also degenerates to a use(door) plan.

```
CONTAIN-ENVIRONMENT(DOOR,ENVIRONMENT) =
                FIND(DOOR)                  PTRANS(LOC(DOOR))
              + D-CONT(DOOR)                GRASP(END1-DOORSLAB)
              + PROPEL(DOOR)                PUSH(END1-DOORSLAB)
```

device-use template

subgoal resolution/specialization

Figure A.50 Pragmatic representation of door environmental containment.

A.3 Multiple Compound Devices

Whereas a pair of scissors or a clothes pin have two components which can move independently, a multiple-compound device has more than one independently moving component. Although the statics and intra-object dynamics are fairly straightforward, the inter-object dynamics begins to become complex when fully elaborated. One such device is the fingernail clipper, which is the first of three multiple compound devices represented in this section.

(1) press
(2) crank can opener

A.3.4 Press

A press modifies the shape and size of objects with a screw. The press functions by disabling motion in a container and then reducing the volume the object can occupy.

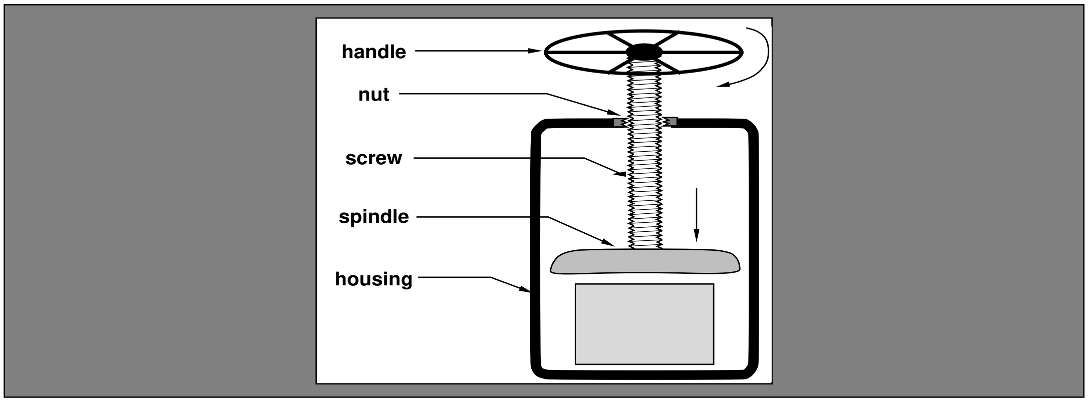


Figure A.51 Press illustration.

Press Statics

An idealized press and static device diagram are illustrated in Fig. A.52.

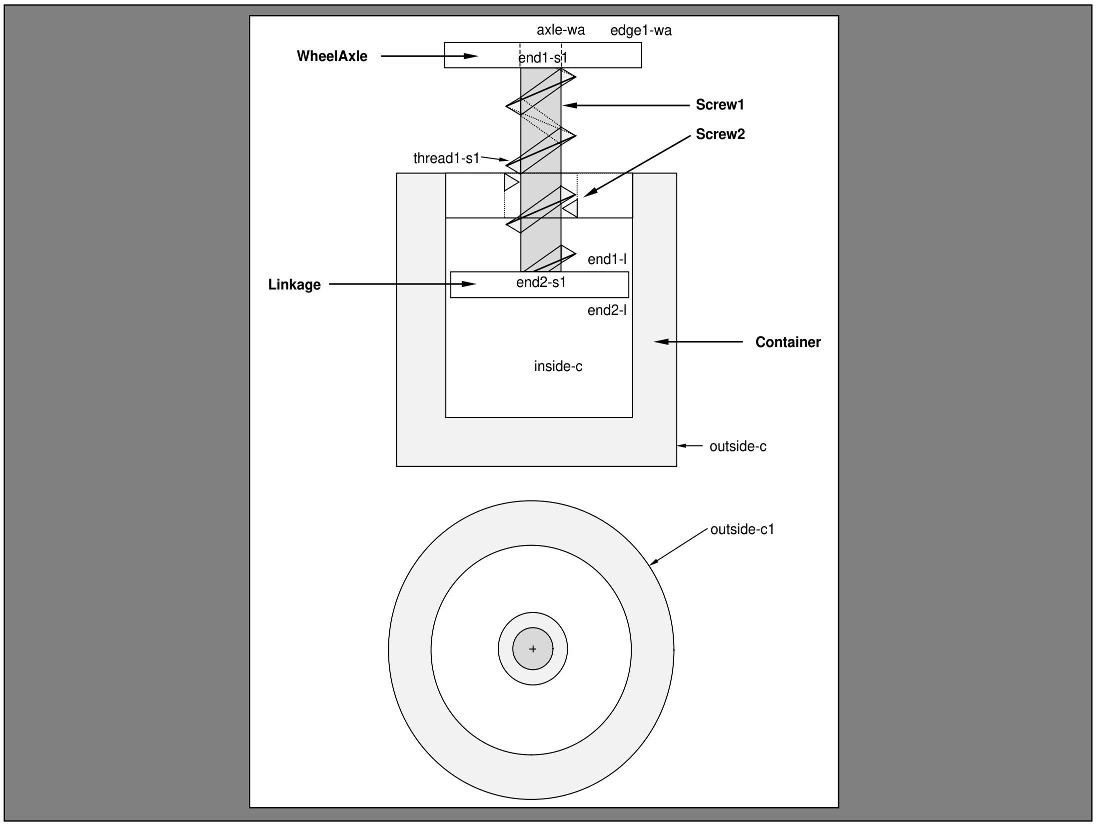


Figure A.52 Press object primitives.

.

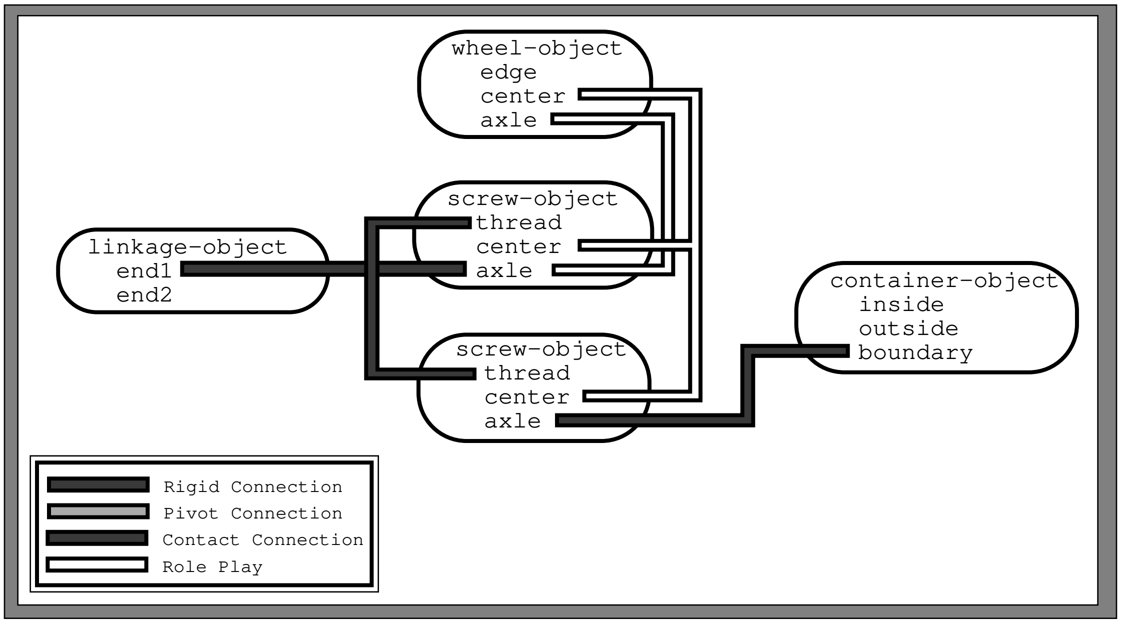


Figure A.53 Press static device diagram.

High-Level Press Dynamics

The pressing function is shown as an MP-Diagram in Fig. A.54. The device function is initiated by a torsional force on the handle rim (EDGE). This is represented with MP-WHEEL-AXLE. The wheel's axle is shared with that of a screw (2), which magnifies force and transforms it to longitudinal force using MP-SCREW.

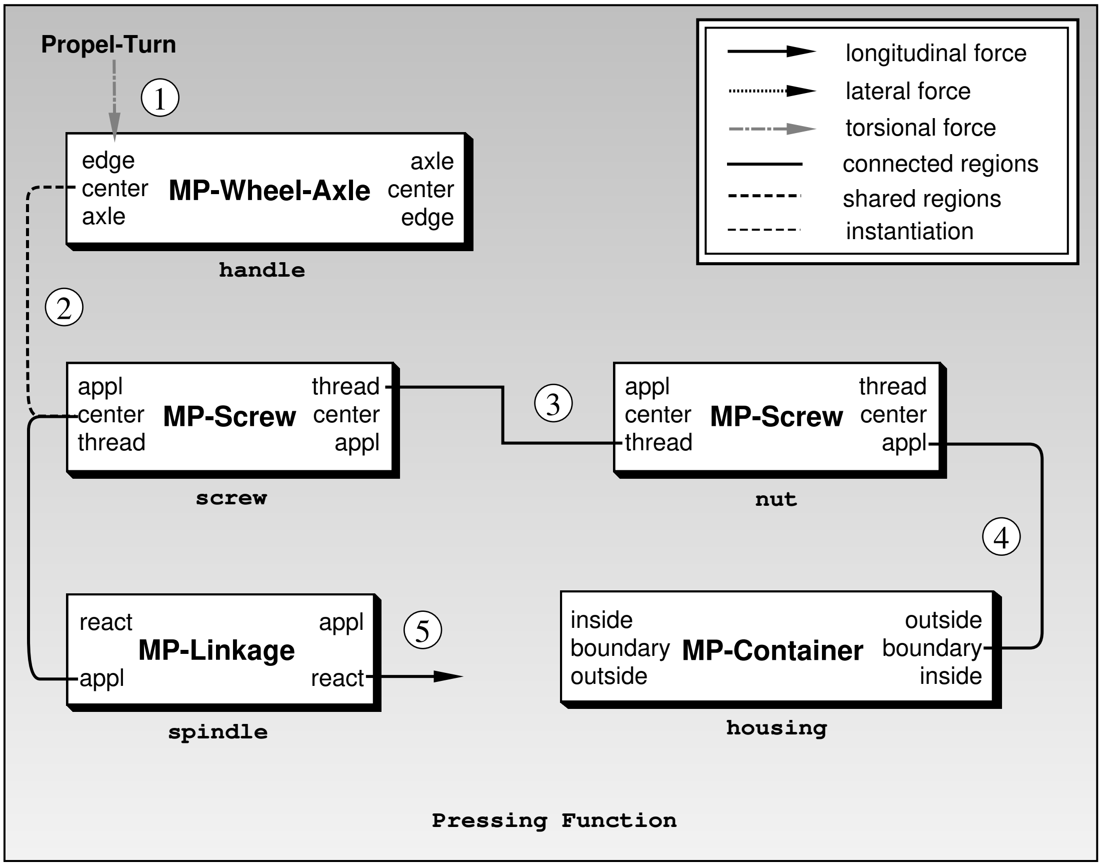


Figure A.54 MP-Diagram for press function.

The screw threads are in contact with the threads of the nut (3), which is represented with another MP-SCREW. The nut is connected to the housing (4), and the spindle which is connected to the screw is within the housing boundary, the it is represented with MP-CONTAINER. As the wheel is turned back and forth, the spindle moves up and down in the housing.

Press Pragmatics

A press has two design-intended uses, to: (1) compress objects together or apart (SOP plan), and (2) extrude material (SO plan). The press illustrated is a container object which has as a component a plane object (a screw). Each of the plans makes use of the fact that a press can apply great force on an object, because the object is confined, and because the screw can magnify force. The press-together plan is illustrated in Fig. A.55. In this plan, the press must be opened so that the objects can be placed inside. Once inside, the shape of spindle (linkage in Figure B.69), container, and objects determines whether the objects are forced together, apart, or extruded. For example, if the objects are a ring and a bearing, and the bearing is resting on the edge of a hole in the ring, and the spindle is flat, then the bearing will be pressed into the ring's hole. If, on the other hand, the bearing is already in the ring, the ring is supported, and the spindle is the shape of the bearing but smaller, then the bearing and ring will be pressed apart. The plan is invoked in basically the same way for

```
PRESS-TOGETHER(PRESS,STUFF) =
        FIND(PRESS)          I-WEDGE(PRESS) +   ←┐
                             I-MECH(PRESS)      ←┼── ①
                             I-CONTN(PRESS) +   ←┘
                             PTRANS(LOC(PRESS))
      + D-CONT(PRESS)        GRASP(PRESS)
      + FIND(STUFF)          PTRANS(LOC(STUFF))
      + D-CONT(STUFF)        GRASP(STUFF)
      + I-PREP(PRESS)        OPEN(PRESS) =
                               GRASP(EDGE-WHEEL) +
                               TURN(EDGE-WHEEL,CW)
      + I-PREP(STUFF)        PLACE(STUFF,INSIDE-CONTAINER)
      + PROPEL(PRESS)        TURN(EDGE-WHEEL,CCW)
             ↑                          ↑
    device-use template    subgoal resolution/specialization
```

Figure A.55 Pragmatic representation of press together.

both. The press-together plan is similar to the extrude-stuff plan in that motion of an object under pressure occurs, but in the former plan, motion is limited to the confines of the container. In the extrude plan, there is an opening in the container which allows a pressure release as long as the object changes shape and moves through the opening.

A.3.5 Crank Can-Opener

A crank can-opener is illustrated in Fig. A.56.

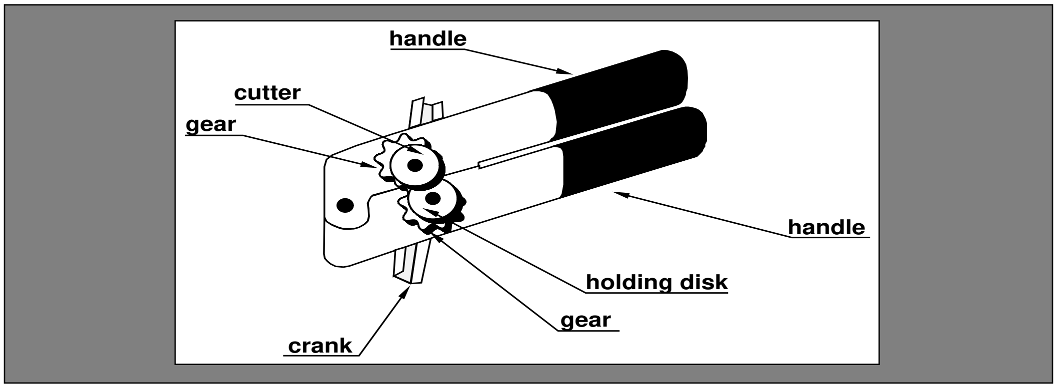


Figure A.56 Crank can opener illustration.

Crank Can Opener Statics

An idealized can opener and static device diagram are depited in Fig. A.57.

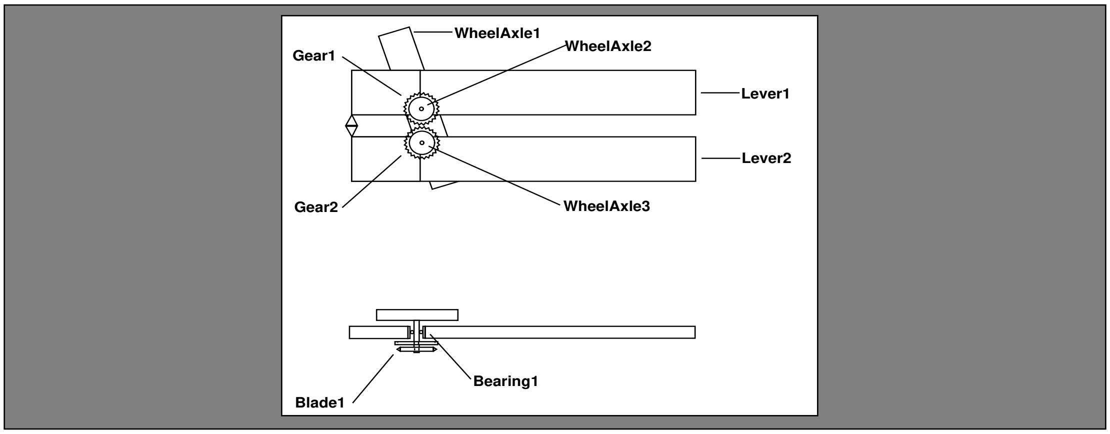


Figure A.57 Crank can opener object primitives.

The crank can-opener SDD is shown in Fig. A.57:

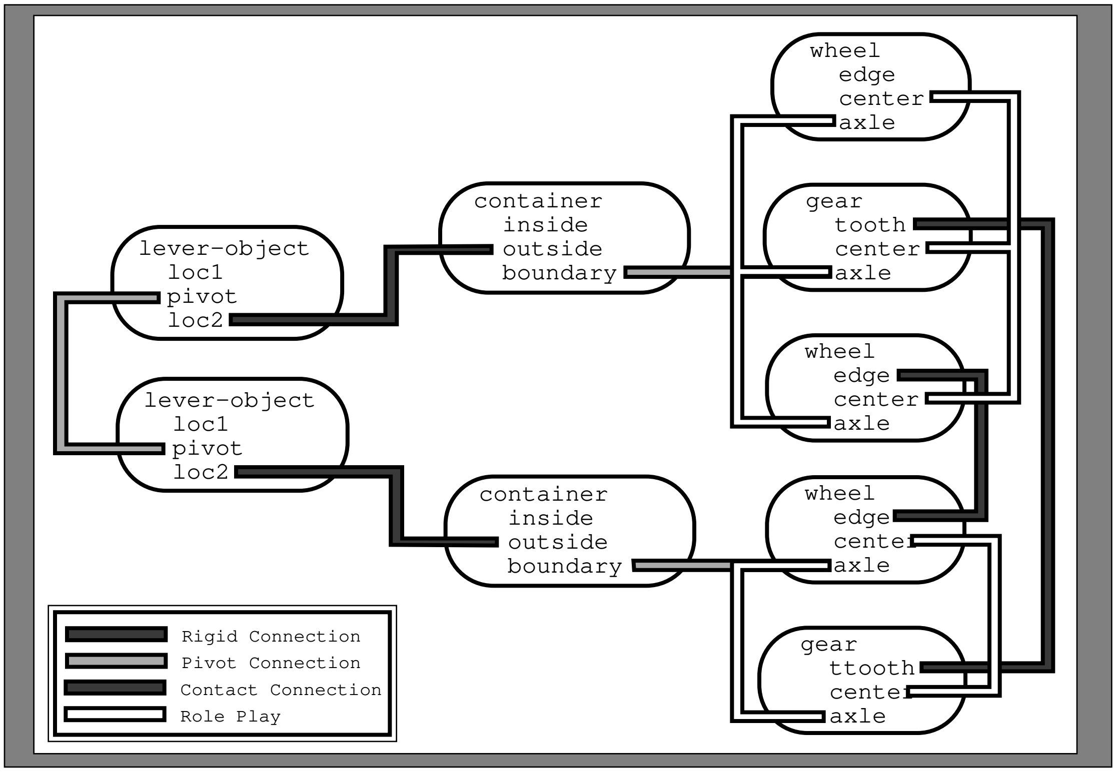


Figure A.58 Crank can opener static device diagram.

High-Level Crank Can-Opener Dynamics

The crank can-opener slicing function is represented as an MP-Diagram in Fig. A.59. The LINKAGE1 and LINKAGE2 components are represented with MP-LEVER-TYPE2, because of the common rivet. The crank assembly (CRANK, HOLD-GEAR, and HOLD-DISK) axle and the cutter assembly (CUT-GEAR and CUT-DISK) axle also share axle roles of MP-WHEEL-AXLE (at 3 and 4). Rotational force is applied to CRANK (2), whose center is supported, so it is represented with MP-WHEEL-AXLE. Because of the shared axle, the resulting force is transmitted to HOLD-GEAR, and the toothed disk HOLD-DISK. The gears, HOLD-GEAR and CUT-GEAR, and toothed disk HOLD-DISK are represented with MP-GEARs, which enable position control and force transmission. Finally, the cutting disk CUT-DISK is represented with MP-BLADE, which enables the puncturing and slicing.

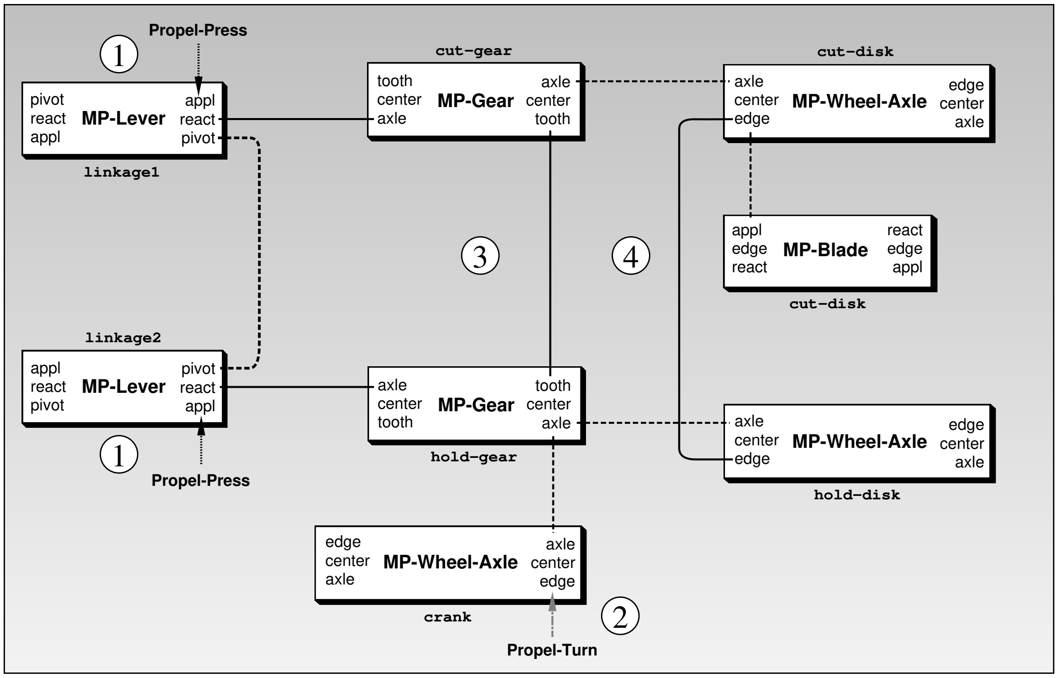
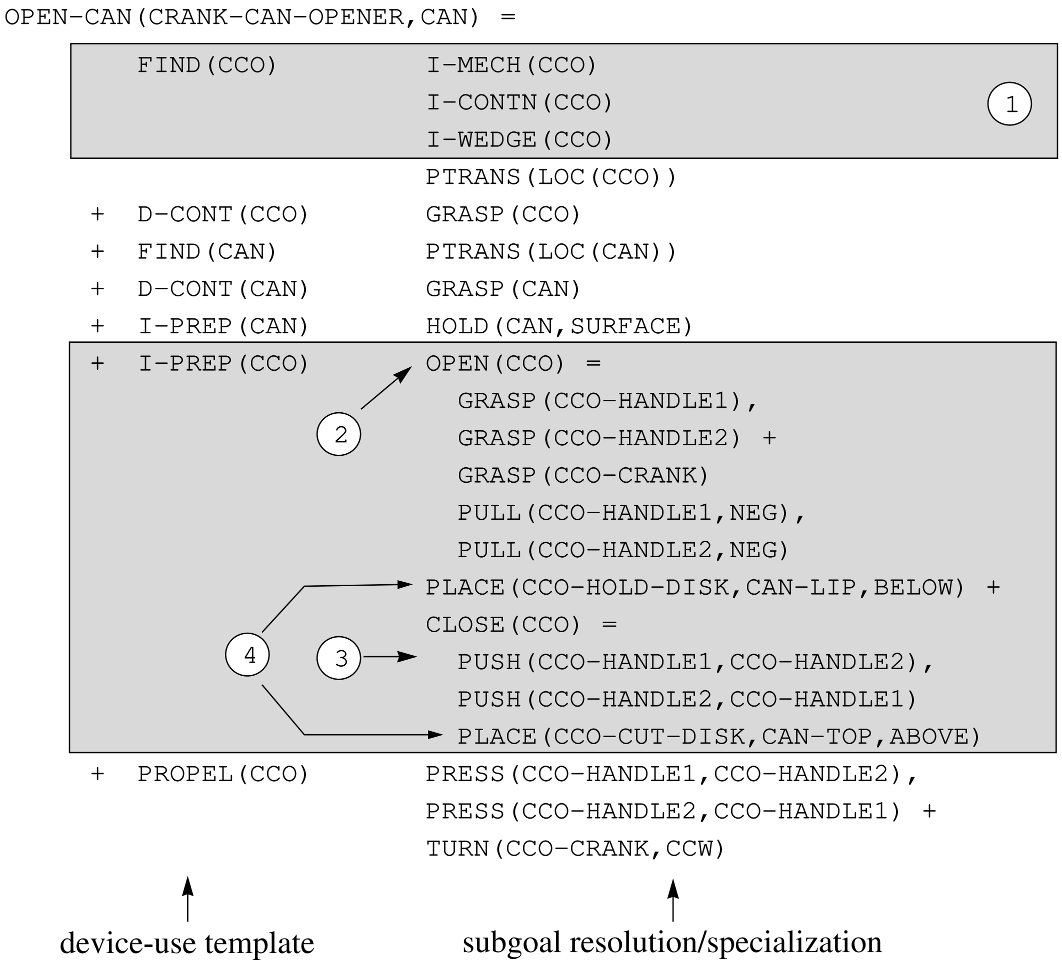


Figure A.59 MP-Diagram for crank can opener function.

The CRANK-CAN-OPENER can also be used to pound nails or crack nuts. These applications can be represented as generalized machine primitives. When the CRANK-CAN-OPENER is used to pound nails, the entire device is associated with a Type 3 MP-LEVER. When the CRANK-CAN-OPENER is used as to crack nuts, the entire device is associated with a Type 2 MP-LEVER. The CRANK-CAN-OPENER will always be associated with the subset of machine primitives applied to produce the desired effect, regardless of the design-intended device function. This notion is identical to what Schank and Abelson called the *main conceptualization* (maincon) of a script or plan. In FONM, a device's maincon is associated with a single behavioral process primitive for a machine primitive, such as DEFORM for MP-BLADE.

Crank Can-Opener Pragmatics

A crank can opener is designed for removing lids from cans (i.e., an SOP plan in support of a CROM plan). The crank can opener has many independently moving components, which form small systems which perform specific functions. For example, the handles provide leverage to initiate the cut in the can, and they house the remaining components. The crank, cut-gear and cut-disk transmit force to the can. The hold-gear and hold-disk restrain the can's motion and transmit motion to the can. Each of these components and systems is activated in the crank can opener function and enabled through its use, as shown in Fig. A.60.To prepare and use the crank can opener, five subtasks must be performed: (a) the handles must be opened (2), (b) the HOLD-DISK must be oriented under the can lip (4), (c) the handles must be closed (3), so that the two gears engage and so that the CUT-DISK contacts the top of the lid (4), (d) the handles must be pressed together so as to puncture the lid, and (e) the crank must be turned so as to force the cut (i.e., SLICE) around the lid. The notation used to describe the direction the handles are pulled is based on the direction being away from the handle CG. The placement of the can opener about the can is represented as a preparation of the device rather than the can, since, once the can is held stable, it is prepared for cutting. The actual application of the device involves two kinds of input: (a) to initiate the cut, and (b) to continue the cut. The first enables the behavior BPP-DEFORM-PUNCTURE, and the second enables the behavior BPP-DEFORM-SPLIT, both conceptualization of MP-BLADE.



```
OPEN-CAN(CRANK-CAN-OPENER,CAN) =
       FIND(CCO)           I-MECH(CCO)
                           I-CONTN(CCO)                    ①
                           I-WEDGE(CCO)
                           PTRANS(LOC(CCO))
    +  D-CONT(CCO)         GRASP(CCO)
    +  FIND(CAN)           PTRANS(LOC(CAN))
    +  D-CONT(CAN)         GRASP(CAN)
    +  I-PREP(CAN)         HOLD(CAN,SURFACE)
    +  I-PREP(CCO)         OPEN(CCO) =
                             GRASP(CCO-HANDLE1),
                     ②       GRASP(CCO-HANDLE2) +
                             GRASP(CCO-CRANK)
                             PULL(CCO-HANDLE1,NEG),
                             PULL(CCO-HANDLE2,NEG)
                           PLACE(CCO-HOLD-DISK,CAN-LIP,BELOW) +
                           CLOSE(CCO) =
                ④    ③       PUSH(CCO-HANDLE1,CCO-HANDLE2),
                             PUSH(CCO-HANDLE2,CCO-HANDLE1)
                             PLACE(CCO-CUT-DISK,CAN-TOP,ABOVE)
    +  PROPEL(CCO)         PRESS(CCO-HANDLE1,CCO-HANDLE2),
                           PRESS(CCO-HANDLE2,CCO-HANDLE1) +
                           TURN(CCO-CRANK,CCW)
```

Figure A.60 Pragmatic representation of crank can opener can opening.

A.4 Complex Devices

Complex devices involve objects which may or may not be permanent components of the device, have complex motions, or have large numbers of components. One such example is a toy gun, which has a projectile which is only connected to the device when being armed and preparing to fire. Another complex device is a mousetrap, since the hook and baitplate motion are both confined to the limits imposed by their connectors, but free to move in every other respect. In this section, these two devices and two others are presented. The intent is to address some of the issues in representing such devices, rather than to represent them explicitly.

(1) corkscrew
(2) egg beater
(3) mouse trap

A.4.6 Corkscrew

A corkscrew is illustrated and labeled in Fig. A.61:

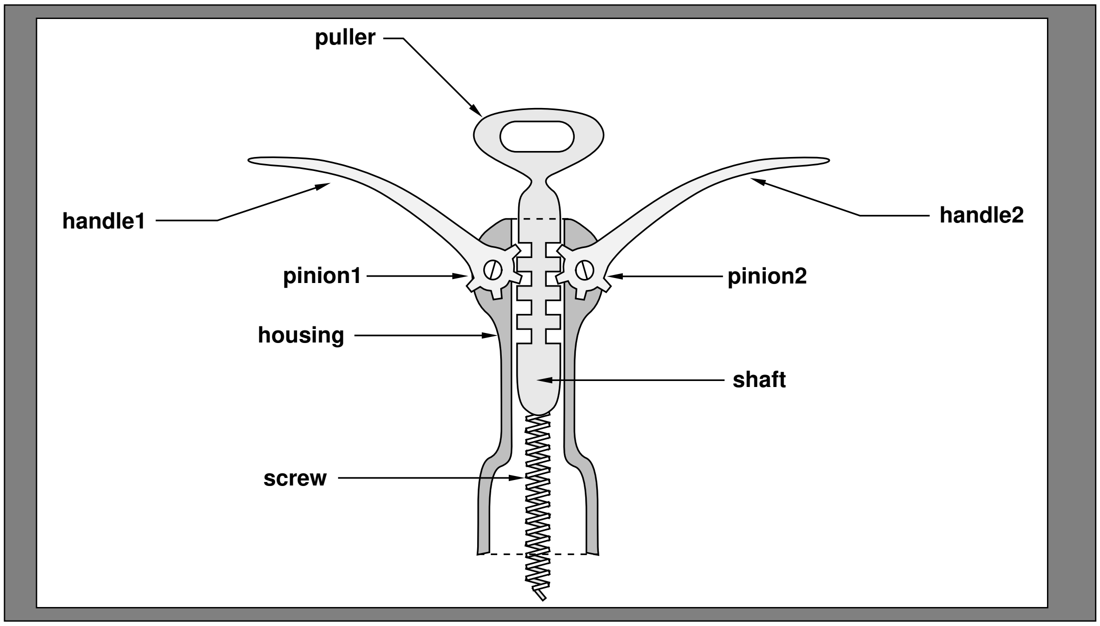


Figure A.61 Corkscrew illustration.

Corkscrew Statics

An idealized corkscrew and its static device diagram is illustrated in Fig. A.64.

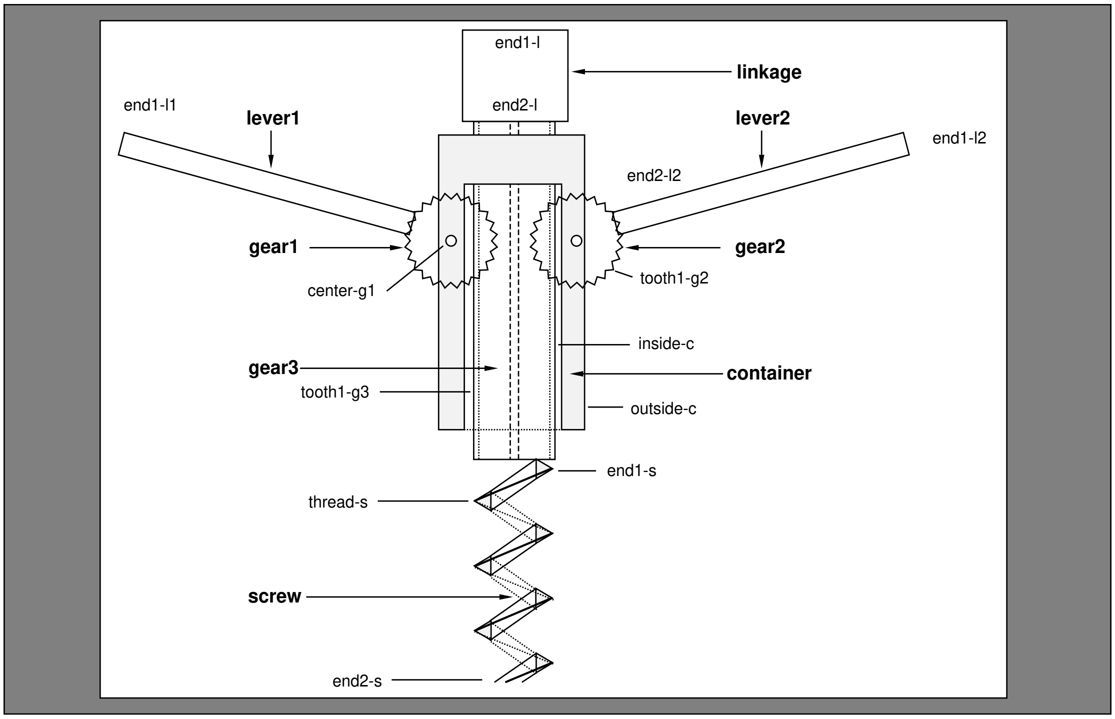


Figure A.62 Idealized corkscrew representation.

The corkscrew SDD is shown in Fig. A.63:

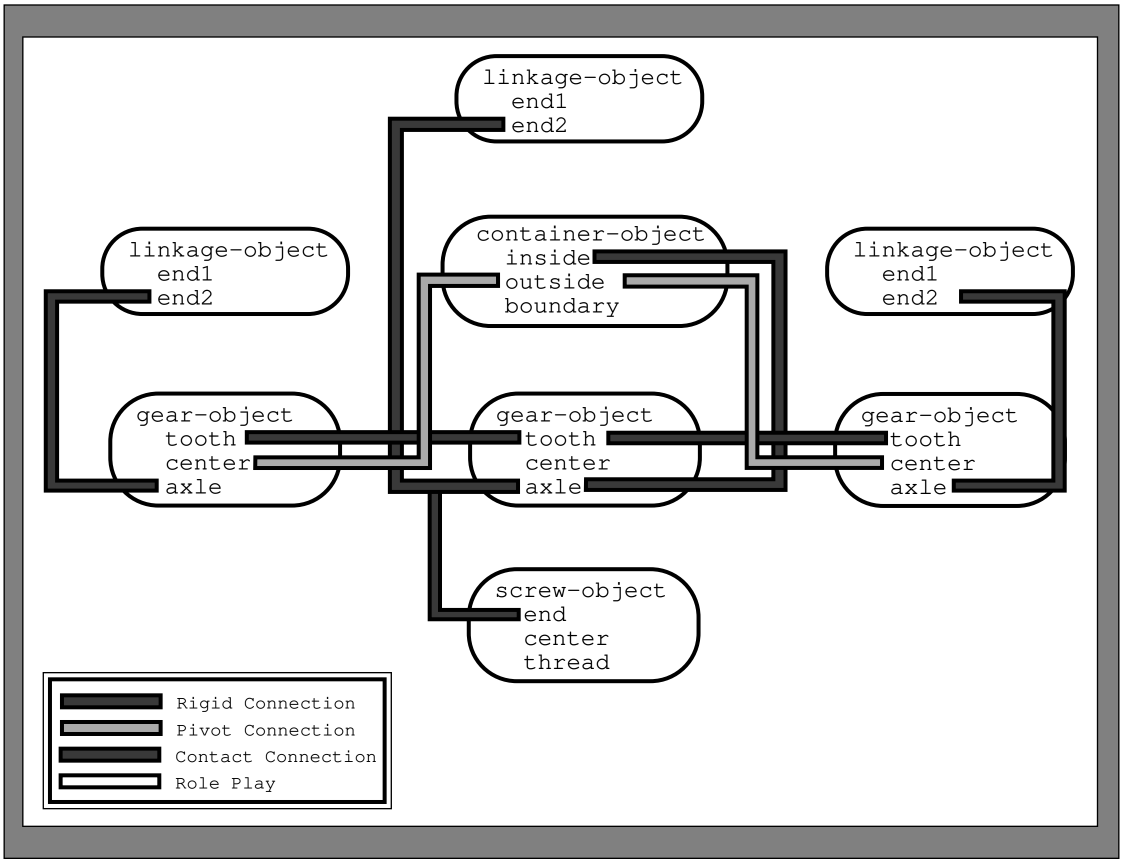


Figure A.63 Corkscrew static device diagram.

High-Level Corkscrew Dynamics

The corkscrew has two normal functions: a screwing function and a pulling function, both of which are represented in Fig. A.64.

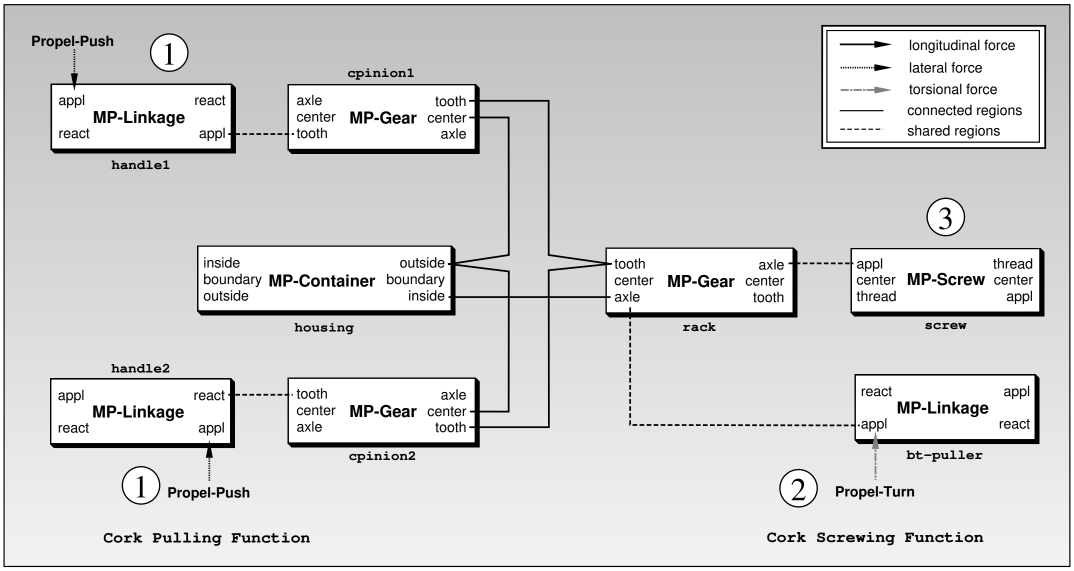


Figure A.64 MP-Diagrams for corkscrew functions.

Corkscrew Pragmatics

The cork screw has a single design-intended use: (1) remove corks, which is a control-range-of-motion (CROM) plan. The corkscrew is able to control the cork and open the bottle, because it utilizes a screw to burrow into and hold the cork, and then uses leverage to pull the cork from the bottle, as shown in Fig. A.65.

```
OPEN-CORKED-BOTTLE(CORK-SCREW,BOTTLE) =
     FIND(CORK-SCREW)       I-CONTN(CORK-SCREW) +  ← ①
                            I-WEDGE(CORK-SCREW) +  ← ②
                            I-MECH(CORK-SCREW) +   ← ③
                            PTRANS(LOC(CORK-SCREW)) ← ④
  +  D-CONT(CORK-SCREW)     GRASP(CORK-SCREW)
  +  FIND(BOTTLE)           PTRANS(LOC(BOTTLE))
  +  D-CONT(BOTTLE)         GRASP(BOTTLE)
  +  I-PREP(BOTTLE)         PLACE(BOTTLE,SURFACE)
  +  I-PREP(CORK-SCREW)     OPEN(CORK-SCREW) =
                              PUSH(CS-LEVER1,DOWN),
                              PUSH(CS-LEVER2,DOWN) +
                              PLACE(END2-CSBODY,TOP-BOTTLE,ABOVE) +
                              PLACE(CS-POINT,CORK,ABOVE) +
                              TURN(CS-HEAD,CCW)
  +  PROPEL(CORK-SCREW)     PUSH(CS-LEVER1,DOWN)
                            PUSH(CS-LEVER2,DOWN)

             ↑                              ↑
    device-use template          instantiation of subgoals
```

Figure A.65 Pragmatic representation of cork pulling.

A.4.7 Egg Beater

An eggbeater is depicted in Fig. A.66:

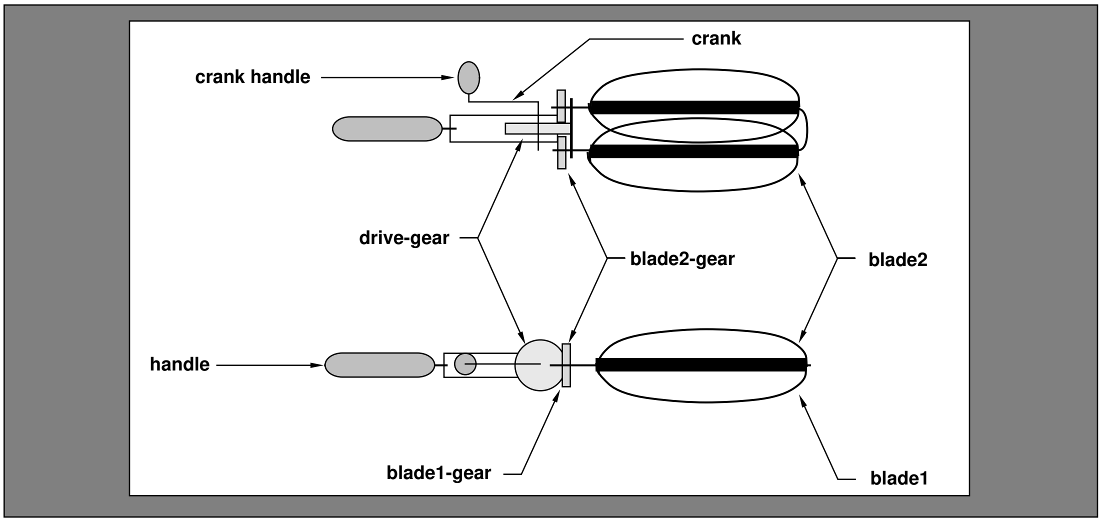


Figure A.66 Egg beater illustration.

Egg Beater Statics

An idealized egg beater is illustrated in Fig. A.67.

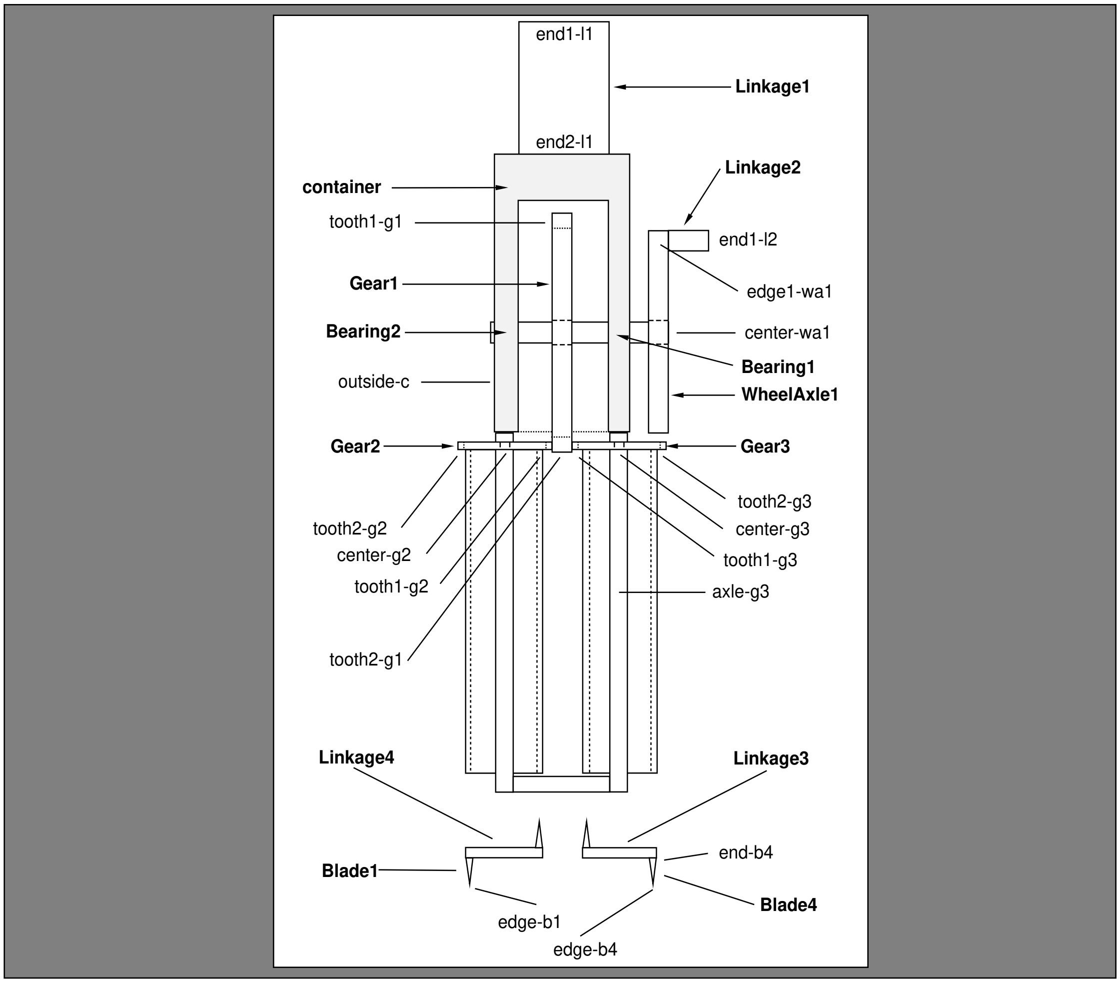


Figure A.67 Idealized representation of egg beater.

The egg-beater is a complex gear composition where the gears move in different rotational planes and effect the motion of counter-rotating blade-whips. The static device diagram for the egg beater is shown in Chapter A.68.

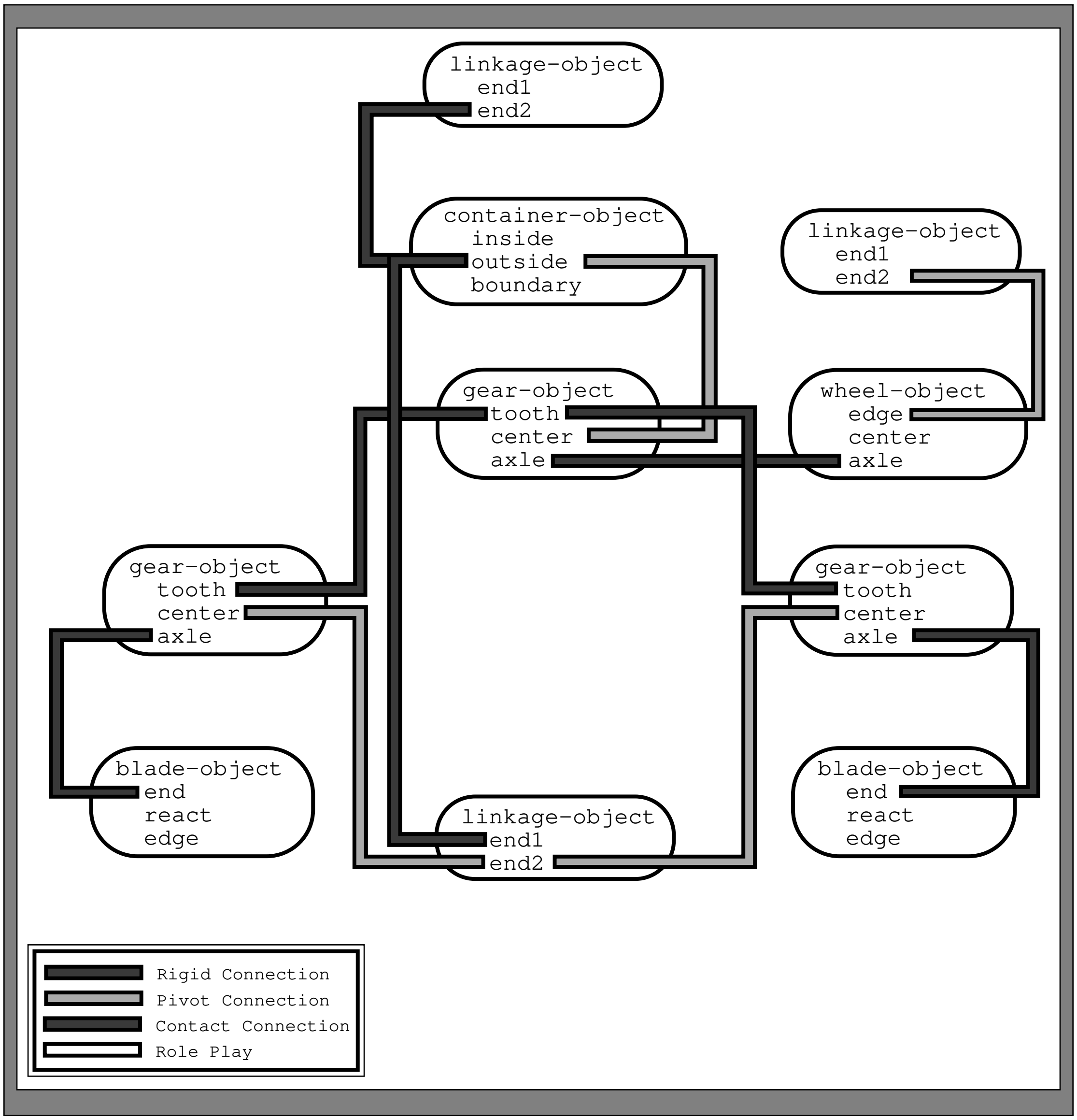


Figure A.68 Static device diagram for idealized eggbeater.

High-Level Egg Beater Dynamics

The MP-Diagram which represents the egg beater whipping function is shown in Fig. A.69.

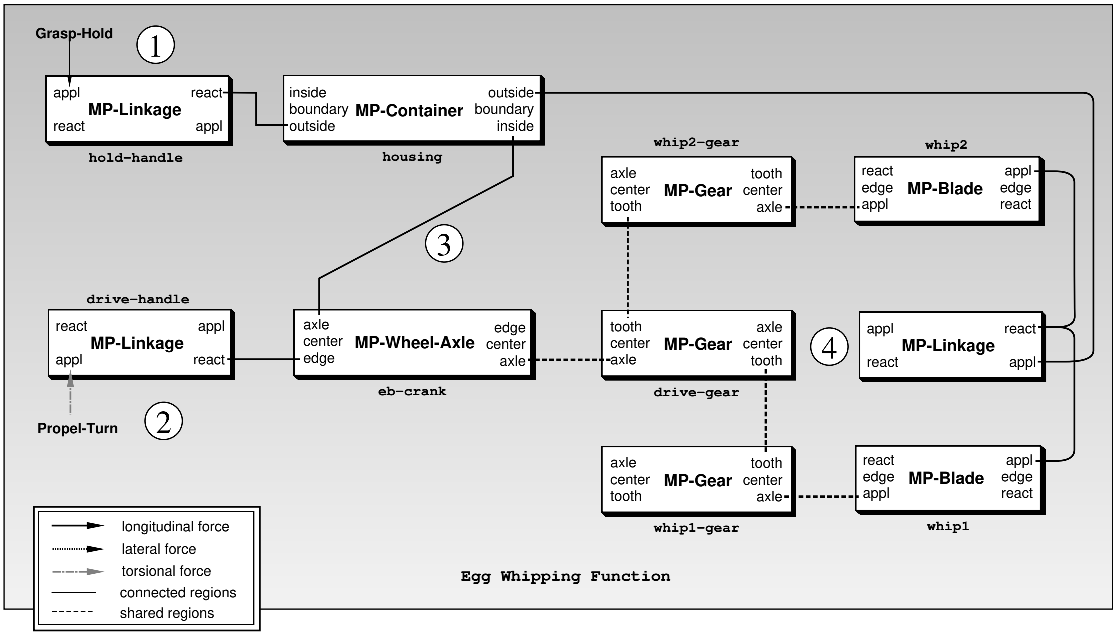


Figure A.69 MP-Diagram for egg beater function.

The egg beater presents a number of problems for mechanical representation. First, the counter rotation of the whips, and the complex shape, make geometric modeling and simulation of the device difficult. However, the MP-Diagram is straight forward. This implies that the function is simple, but that implementing it might take many forms. Since the primary notion of whipping eggs is nothing more than separating them and adding air, it is clear than even a fork can implement the function.

The egg beater combines MP-GEARs and MP-BLADEs differently than in the CRANK-CAN-OPENER. The former device uses gears both to drive the cutter and to drive the soupcan. In the egg beater the blades, and not the stuff, are turned. By using a single MP-GEAR, driving a separate MP-BLADE on either side of its center, results in counter-rotating MP-BLADE motions.

Egg Beater Pragmatics

The egg beater has a single design-intended use: (1) whip eggs, as shown in Fig. A.70. Egg whipping can be seen as a separation or a combination, but both are SOP plans.

```
WHIP(EGG-BEATER,EGGS) =
           FIND(EGG-BEATER)          I-WEDGE(EGG-BEATER) +
                                     I-MECH(EGG-BEATER) +
                                     PTRANS(LOC(EGG-BEATER))
        +  D-CONT(EGG-BEATER)        GRASP(EGG-BEATER)
        +  FIND(EGGS)                PTRANS(LOC(EGGS))
        +  D-CONT(EGGS)              GRASP(EGGS)
        +  I-PREP(EGGS)              CRACK-OPEN(EGGS,BOWL)
        +  I-PREP(EGG-BEATER)        HOLD(EB-HANDLE1) +
                                     PLACE(EB-BLADES,BOWL) +
                                     GRASP(EB-HANDLE2)
        +  PROPEL(EGG-BEATER)        TURN(EB-HANDLE2,CCW)
```

↑ device-use template

↑ instantiation of subgoals

Figure A.70 Pragmatic representation of egg beater whipping.

A.4.8 MouseTrap

A mouse trap is illustrated in Fig. A.71:

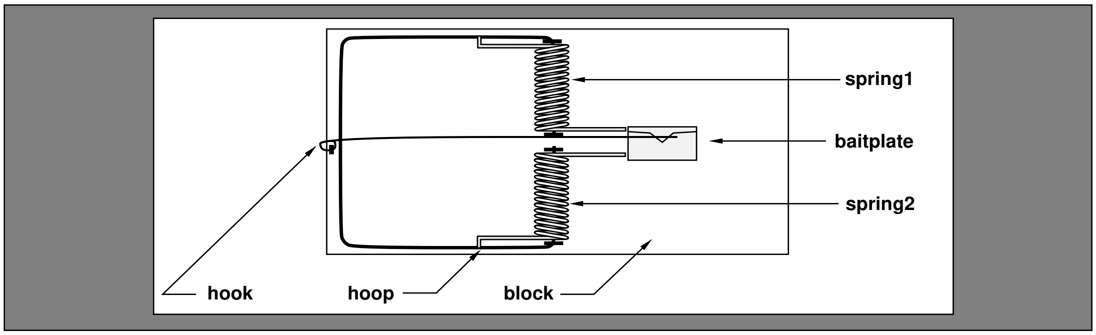


Figure A.71 Mouse trap illustration.

Mousetrap Statics

An idealized mousetrap and its static device diagram are depicted in Fig. A.72.

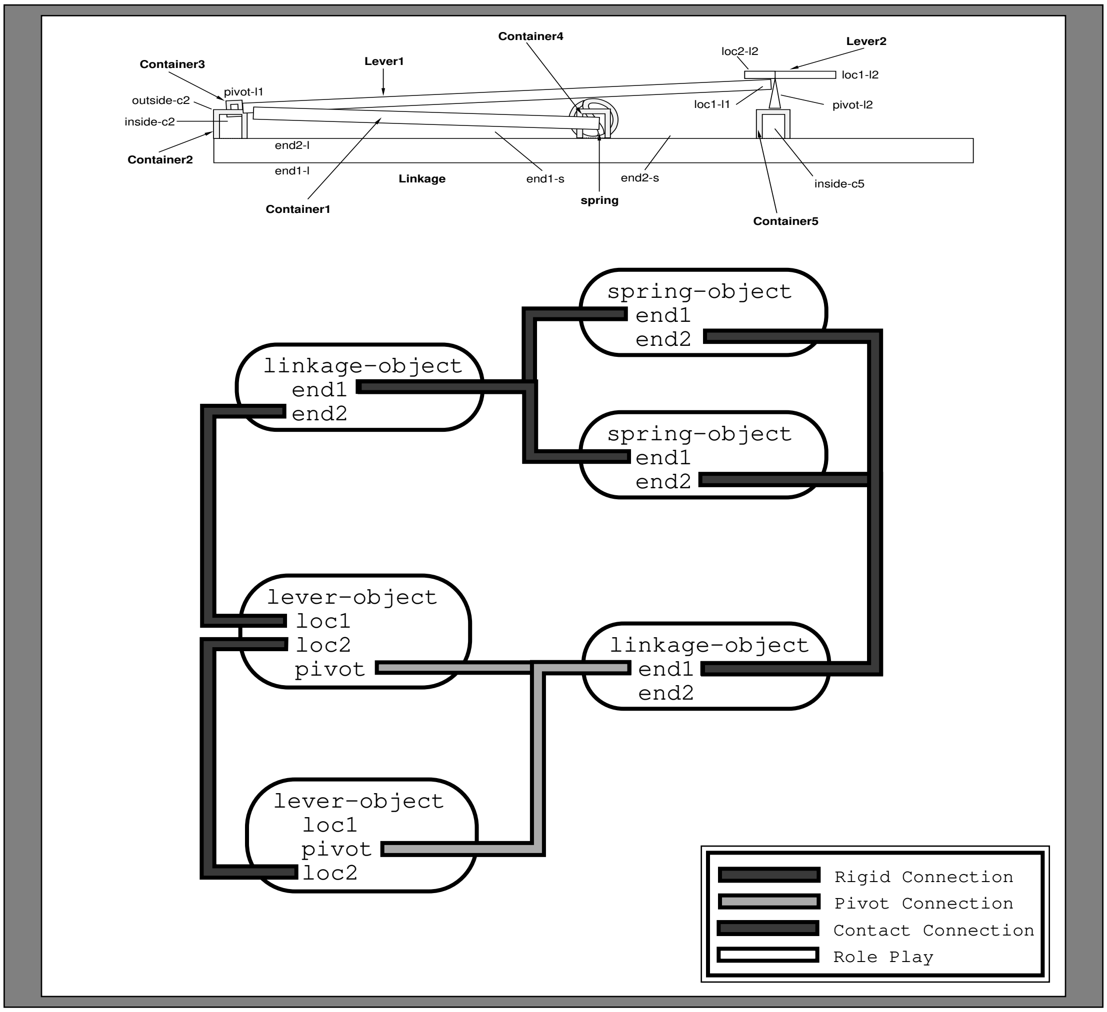


Figure A.72 Mousetrap static device diagram.

High-Level Mousetrap Dynamics

The mouse-trap has two primary functions, arming and firing, like the toy gun. These functions make use of the same primitives, and can be combined into a single MP-Diagram with the flow of force and motion traced through the affected primitives, as shown in Fig. A.73.

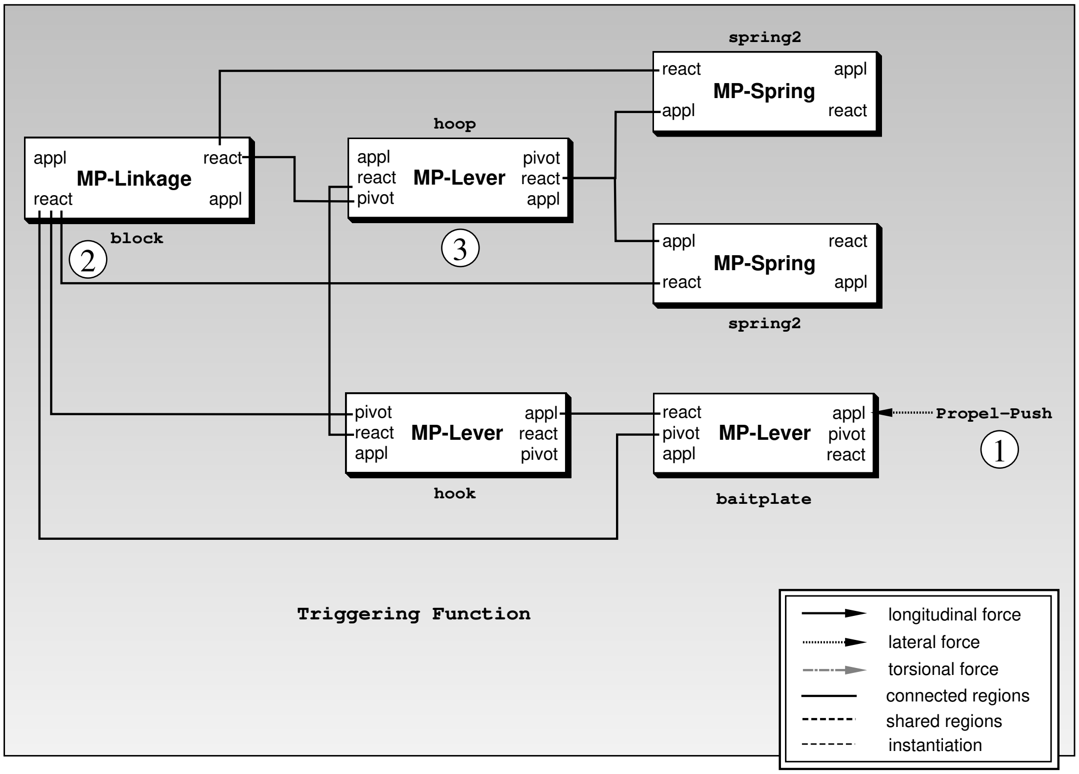


Figure A.73 MP-Diagram for mousetrap function.

Mousetrap Pragmatics

The mouse trap a single design-intended use: (1) to kill mice, which is a CROM plan. The use is illustrated in Fig. A.74

```
TRAP-MOUSE(MOUSE-TRAP,MOUSE) =
         FIND(MOUSE-TRAP)         I-CONTN(MOUSE-TRAP) +
                                  I-MECH(MOUSE-TRAP) +
                                  I-RESIST(MOUSE-TRAP) +
                                  PTRANS(LOC(MOUSE-TRAP))
      +  D-CONT(MOUSE-TRAP)       GRASP(MT-BLOCK)
      +  FIND(BAIT)               PTRANS(LOC(BAIT))
      +  D-CONT(BAIT)             GRASP(BAIT)
      +  I-PREP(BAIT)             PLACE(BAIT,CLAW-MTBAITPLATE,ABOVE)
      +  I-PREP(MOUSE-TRAP)       ARM(MOUSE-TRAP) =
                                    HOLD(MTBLOCK) +
                                    GRASP(MTBAR) +
                                    PULL(MTBAR,MTBAITPLATE) +
                                    HOLD(MTBAR,MTBAITPLATE) +
                                    GRASP(END1-MTLOOP)
      +  PROPEL(MT-LOOP)          PUSH(MT-LOOP,CATCH-MTBAITPLATE)
                  ↑                                ↑
```

device-use template

instantiation of subgoals

Figure A.74 Pragmatic representation of mouse trap use for trapping mice.

Glossary of Terms

A

Applied force: An object behavioral property associated with motion and force transmission. (22)

B

Behavioral delta goal: A goal which describes a state change needed to recover from a failed goal. (209)

Behavioral preconditions: BPP preconditions which are based on behavioral object state changes. (79)

Behavioral Process Primitives: A knowledge structure which represents fundamental mechanical behaviors and can be sequenced to represent object function. (3, 78)

Black box: An object whose function is known but nothing about how the function is achieved is known. (147)

Blackboard: A memory which is available to a variety of processing mechanisms. In the case of conceptual analysis, the working memory is a blackboard since it can be accessed by all the demons. (271)

Bondgraph: A representation approach which describes the ways that a device can be connected to other devices and by what behavioral interactions each connection can support. (327)

Boundary region: Describes the junction of surface regions. An edge is a boundary between two faces, between a face and an indentation, or between a face and a protuberance. (28)

C

Canonical form: A structure into which knowledge can be captured in such a way that two concepts which have the same meaning are represented with the same structure. (8)

Cartesian axis dimensions: The three translational dimensions - X, Y, and Z and their rotational equivalents - P, Q, and R. (22)

Causal keywords: Words which causally connect physical events and enable inference about intervening relationships. For example, "X until Y," or "when X Y." (279)

Clamped connection: A state in which both objects are fully (i.e. all dimensions) restrained. (61)

Complex device: A device comprised of systems, each of which can be a multiple compound device. (62)

Compound device: A device where at least two components can move relative to one another. (62)

Continuity consistency: A model which predicts continuous behavior when applied mechanics would predict continuous behavior. (79)

Cylindrical axis dimensions: The dimensions associated with cylindrical bodies, namely radius and length. In addition, the rotational equivalents for these dimensions complete the set of cylindrical degrees of freedom. (22)

D

Degree of freedom: A dimensional reference. There are six degrees of freedom in space, one each for the three translational axes and one each for the three rotational axes. (23, 330)

Demon agenda: A stack onto which demons are placed and polled against a current or working memory of concepts. (271)

Demons: Delayed procedures with test-act parts that implement rules. (271)

Device appearance: The shape and size of a device's components and their features. (21)

Device construction: The composition and connectivity of a device's components. (21)

Device dynamics: The combination of low-level and high-level behavioral interactions between objects. (77)

Device function: What a device does, with respect to perturbation and perceived results, as opposed to how the result comes about. (147)

Device pragmatics: How a device functions, how it shares similarities with other devices, and how the application of device function is affected by the environment in which it is used. (207)

Device statics: Composition, appearance, orientation and connectivity of a device's components. (21)

Device-use plan: A plan in which a device function is applied. (207)

Dynamic consistency: When a model is based on the same behavioral principles upon which applied mechanics is based. (78)

E

Elastic deformation: Changes in object size which are reversed when the applied force is removed. (98)

Elastic limit: In Applied Mechanics the elastic limit is 0.2% strain on the stress-strain curve. Beyond the elastic limit some strain becomes permanent deformation. (97)

Episodic memory: A memory of experiences and events. (271)

External nodes: Boundary nodes of a finite element. (330)

F

Face: A region representing a flat surface and associated with support. (27)

Finite element: An engineering theory which represents objects as compositions of discrete objects with linear properties. A finite element object appears like a wire frame. (44, 47)

Fixed connection: A state (identical to clamped) in which both objects are fully (i.e. all dimensions) restrained. (61)

Fixed-axle device:A device whose axle region location hasadifferent radius than the appl region location, so force magnification is produced on the device itself. (167)

Free-axle device: A device which rotates about its axle without axle rotation. Force magnification is generated by cascading free-axle devices of different radii. (167)

G

Generalized machine: A device which takes on the functional characteristics of one or more of its component functions. (193)

Generic process: The minimum definition which allows recognition of a process class. (83)

Global behavior: High-level object behavior associated with device function. (77)

Grounded object: An object which is anchored to the inertial frame and cannot move in any dimensional axis. (245)

H

High-level behavior: Behavior which is observed as the function of an object. (77)

I

In-plane: Force or Motion resulting from a process is in the same dimension as the one which enabled the process. (159)

Indentation: A region representing a concave volume and associated with containment. (27)

Individual object: A single object and its regions. An individual object is the primitive building block in the FONM representation (see object primitive). (21)

Individual View: A collection of objects and the relations between their quantities. Generally a bounding state for recognizing behavior or function. (310)

Internal energy: An object which is perturbed but unable to move must absorb, store, or internalize the force. The internal force is used to change the size and shape of the object and, when the applied force is removed, to return the object (98)

Internal force: An object behavioral property associated with energy storage. (22)

Inter-object dynamics: Interactions which are not explicitly a part of an object primitive but, rather, with a device. Usually motion of a device component or mechanism. (149)

Intra-object dynamics: Dynamic interactions which are associated with an object primitive. (148)

K

Knowledge Dependency Graph: A diagram illustrating the class dependencies of a KR approach. (9)

Knowledge primitive: Represents the lowest level of inference in the knowledge domain, and is the inferential building block of the system. (8)

L

Linkage: An elastic object, capable of transmitting mechanical force in at least one dimension and direction. (30)

Local behavior: Low-level behavior associated with device components and their interactions. (77)

Location: A knolwedge structure with two roles -- ref and dimv for representing region locations on objects. (28)

Location value: Locations on an object are described with a discrete set of values (FAR-LEFT, LEFT, NEAR-LEFT, MIDDLE, NEAR-RIGHT, RIGHT, and FAR-RIGHT). (28)

Low-level behavior: Behavior of and between components of a device that are not observed as the function of the device. (77)

M

Machine plans: Plans which are used to achieve mechanical instrumental goals, and which are associated with particular machine functions. (216)

Machine Primitive: A primitive machine function, which is represented as a schemata associated with a specific sequence of behavioral processes. (3)

Material division: Any DEFORM process which results in a new region and releases internal energy, such as a cut or tear. (105)

Material stiffness: An object property associated with the ability to transmit mechanical force. (22)

McDypar: The micro version of Dyer's demon parser (see Dyer, 1983). (271)

Mechanical advantage: Mechanical force magnification brought about as a tradeoff between applied force and the distance through which it acts. (95)

Mechanical instrumental goal: Goals which relate a desired function to the devices which can instantiate it. (212)

Mechanical understanding: The ability to recognize and comprehend the relationship between the use of an object and the manner in which the object behaves. (4)

MP Diagram: A diagram of device function composed of MP blocks and links representing interobject connectivity. (156)

N

Naive Mechanics: A description of mechanical devices which is based on knowledge shared by members of a culture through experience with objects in their everyday lives. (3)

Naive reasoner: Someone whose knowledge is based on experience and whose reasoning is based on commonsense or culturally shared principles. (3)

Nodal points: Discrete points which define the boundaries of a finite element. All behavior and states of the finite element are defined at nodal points. (330)

O

Object behavior: How and object works. The internal interactions between object components which is ignored by object function. Behavior is described most simply as a state history for any particular object property. (3)

Object deformation: In mechanics this is called strain, or the ratio between the change in length to the original length. The theory of elasticity is based on small deformations, those which do not exceed the 0.2% strain limit. (98)

Object function: What an object does when perturbed, represented with a schemata which has initiating and terminating states, and I/O regions associated with the function. (3)

Object motion: The time history which describes an object's change of position and how it is brought about. Object motion is represented with BPP-MOTION. (83)

Object primitive: An object which is commonly recognized and associated with a particular function but is not decomposed to more primitive functional objects. (43)

Object regions: General appearance of an object, described by shape, size, and location. (22)

Object restraint: An object behavioral property associated with contact between objects and motion constraint. (22)

Off-axis forcing: A force which is applied to an object which is not directed through the object CG. (85)

Offset angle: The angle between two surface regions. (59)

Offset ratio: The ratio of vertical travel to horizontal travel in inclined planes. Used as a measure of planar motion advantage. (184)

On-axis forcing: A force applied through the object's center of gravity. (85)

Orientation: An object relational characteristic that defines dimensional alignment between two objects. (58)

P

Parse Diagram: A graphical representation showing how textual entries are associated with each other with demons. (274)

Physical state: A knowledge structure which represents object material properties and behavior. (22)

Pivot: A means of modifying how force is applied to object is contact. Three pivot types are defined in FONM, a fulcrum pivot, a fixed pivot, and an offset pivot. (95)

Plastic deformation: Changes in object size and shape which remain when the applied force is removed. (98)

Predictive consistency: A model which, when given the same object and initial conditions, makes the same predictions of behavior as would applied mechanics. (79)

Process: A knowledge structure which forms the backbone for behavioral process primitives. (79)

Property history: A sequence of object states for a particular behavioral property. (78)

Protuberance: A region representing a concave volume and associated with interference. (27)

Q

Qualitative Process: A dynamic structure (Forbus) that control the change of object state and individual views. A qualitative process describes object time histories. (311)

Quant: A knowledge structure for representing property values (quantities). The quant KS has three slots - state, unit, and value. (25)

Quantity space: A one-dimensional array of values for a particular unit type. The array describes a continuum from the minimum possible value to the maximum possible value, but is represented only with the minimum and maximum known values. (25)

R

Radius ratio: The ratio of wheel, pulley, and gear radii used to determine relative motion. (166)

Reacted force: When force is applied to an object at one location, and the object is in contact with another object at a second location, then the applied force is reacted at the second location. (93)

Relational characteristic: A spatial relationship between two primitive objects, typically orientation and placement. (58)

Representational granularity: The degree of specificity of a set of representational constructs, and hence the degree of inference associated with them. (79)

Rigid: The object has no mobility but can transmit all forces. (245)

Role-play link: A link between two knowledge structures which associates the role of one with the role of the other. Whichever filler is defined for one will also be defined for the other. (357)

S

Semantic and Situation Memory: An inviolate or template memory of problem-solving situations and concepts. (236, 270)

Shared dimensions: When two object regions lie on the same dimension but have different locations they share the dimension. Two ends of a stick share the longitudinal axis. (160)

Simple device: A device where all the components are connected together and move as a unit. (62)

Situation interpretation: Recognizing and understanding goals within a problem-solving context. (4)

Spawning demons: Placing demons onto an agenda. The term spawn is associated with a demon's activation because of its context specificity and competitive nature. (271)

Static device description: The device characteristics which describe a device's construction and appearance. (21)

Static Device Diagram (SDD): An idealization of the connectivity of device components as FONM primitive objects and arcs representing the associated RESTRAIN relationship between the components. (62, 64)

Static equilibrium: When all forces in a specified dimension balance. (81)

Store: The process primitive which represents elastic energy storage in objects. (98)

Subgoal resolution: A subgoal specialization of a plan which identifies a set of goal types which satisfy the plan but is not instantiated with a particular goal. (214, 218)

Surface region: Represents an area or volume. There are three surface region types - an indentation, a face, and a protuberance, depending on whether the object is concave, flat, or convex over the volume. (27)

T

Taxonomy: In knowledge representation, a taxonomy refers to a finite set of knowledge structures which span a knowledge class. (8)

Temporally sequential: Actions, scripts, or plans which occur in order. (219)

Temporally simultaneous: Temporally Parallel. Actions, scripts, or plans which occur at the same time. (219)

Transient behavior: Behavior which occurs over short periods of time immediately following a perturbation. FONM is a steady state approach to representing device dynamics. (79)

Type-1 lever: A lever which has the pivot or fulcrum between the location where force is applied and force is reacted. (161)

Bibliography

Allen, J. (1984). Toward a general theory of action and time. *Artificial Intelligence*, 23(2):123–154.

Allen, J. F. (1985). Maintaining knowledge about temporal intervals. In *Readings in Knowledge Representation*, pages 502–522. Morgan Kaufman.

Alonso, M. and Finn, E. J. (1970). *Physics*. Addison-Wesley, Reading, MA.

Alvarado, S. J. (1989). *Understanding Editorial Text: A Computer Model of Argument Comprehension*. PhD thesis, University of California at Los Angeles.

Archer, R. R., Cook, N. H., Crandall, S. H., Dahl, C. N., Lardner, T. J., McClintock, Frank A. Rabinowicz, E., and Reichenbach, G. S. (1972). *An Introduction to the Mechanics of Solids*. McGraw-Hill, New York, 2 edition.

Arens, Y. (1986). *Cluster: An Approach to Contextual Language Understanding*. Report ucb/csd 86/293, University of California at Berkekley, Computer Science Division.

Arkes, H. and Harkness, A. (1980). Effect of making a diagnosis on subsequent recogniton of symptoms. *Journal of Experimental Psychology: Human Learning and Memory*, 6:568–575.

Bateman, J. A. and Paris, C. L. (1989). Phrasing a text in terms the user can understand. In *proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, Vol. II, pages 1511-1517, Detroit, MI.

Bramwell, M. and Mostyn, D. (1984). *How Things Work*. Usborne Publishing.

Brown, D. C. and Breau, R. (1986). Types of constraints in routine design problem-solving. In Sriram, D. and Adey, R., editors, *Applications of Artificial Intelligence in Engineering Problems*, volume 1, pages 383–390, Southampton, United Kingdom. 1st International Conference, Springer Verlag.

Bullock, M., Gelman, R., and Baillargeon, R. (1982). The development of causal reasoning. In Friedman, W. J., editor, *The Developmental Psychology of Time*, pages 209–254. Academic Press.

Cammarata, S. J. and Melkanoff, M. A. (1988). An information dictionary for managing cad/cam databases. *Database Programming and Design*, 1(3):26–35.

Caney, S. (1984). *Invention Book*. Workman Publishing.

Carbonell, J. (1982). Metaphors and physical envisionment. In Lehnert, W. and Ringle, M., editors, *Strategies for Natural Language Processing*, chapter 15, pages 415–434. Lawrence Erlbaum Associates.

Carbonell, J. (1983). Learning by analogy: Formulating and generalizing plans from past experience. In Michalski, R., Carbonell, J., and Mitchell, T., editors, *Machine Learning: An Artificial Intelligence Approach*. Tioga, Palo Alto, CA.

Carbonell, J. (1986). Derivational analogy: A theory of reconstructive problem solving and expertise acquisition. In Michalski, R., Carbonell, J., and Mitchell, T., editors, *Machine Learning: An Artificial Intelligence Approach*, volume II. Morgan Kauffman, San Mateo, CA.

Chittaro, L. and Tasso, C. and Toppano, E. (1993). Putting Functional Knowledge on Firmer Ground. In *Proceedings of the 1993 AAAI workshop on Reasoning about Function*. July, 1993.

Clough, R.W. (1960): The Finite Element Method in Plane Stress Analysis. In *Proceedings of the Second Conference on Electronic Computation,* ASCE, Pittsburgh, Pa, Sept 8-9 1960.

Coyne, R. D., Rosenman, M. A., Radford, A. D., and Gero, J. S. (1987). A logic model of creativity in knowledge-based cad. Unpublished research.

Cullingford, R. E. (1978). *Script Application: Computer Understanding of Newspaper Stories*. PhD thesis, Yale University, New Haven, Connecticut. Department of Computer Science, Research Report 116.

DeBono, E. (1980). *Children Solve Problems*. Penguin.

Dejong, G. and Mooney, R. (1986). Explanation-based learning: An alternative view. *Machine Learning*, 1(2):145–176.

DeKleer, J. (1984). How circuits work. *Artificial Intelligence*, 24:205–280.

DeKleer, J. and Seeley Brown, J. (1983). A qualitative physics based on confluences. *Artificial Intelligence*, 24:7–84.

Desai, C. S. and Abel, J. F. (1972). *Introduction to the Finite Element Method: A Numerical Method for Engineering Analysis*. Van Nostrand Reinhold, New York.

Dietterich, T. (1986). Learning at the knowledge level. *Machine Learning*, 1(3):287–315.

diSessa, A. A. (1983). Phenomenology and the evolution of intuition. In Gentner, D. and Stevens, A., editors, *Mental Models*, chapter 2, pages 15–34. Lawrence Erlbaum Associates, Hillsdale, N.J.

Doyle, R. (1989a). *Hypothesizing Device Mechanisms: Opening Up The Black Box*. PhD thesis, Massachussetts Institute of Technology. Report TR-1047.

Doyle, R. (1989b). Reasoning about Hidden Mechanisms. In *Proceedings of the Eleventh International Joint Coference on Artificial Intelligence*, Vol. 2, pages 1343-1349. Detroit, MI.

Duncker, K. (1945). On problem solving. *Psychological Monographs*, 58(5).

Dyer, M. G. (1983). *In-Depth Understanding*. MIT Press, Cambridge, Massachusetts.

Dyer, M. G. and Flowers, M. (1984). Automating design invention. In *Proceedings of Autofact 6*, Anaheim, CA.

Dyer, M. G., Flowers, M., and Hodges, J. (1985). Mechanical invention in edison. *IEEE Proceedings*.

Dyer, M. G., Flowers, M., and Hodges, J. (1986). Edison: an engineering design invention system operating naively. In D. Sriram and R. Adey (eds), *Applications of Artificial Intelligence in Engineering Problems*, Springer-Verlag, pp 327–361.

Dyer, M. G., Flowers, M., and Hodges, J. (1987a). Naive mechanics comprehension and invention in edison. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, volume 2, pages 696–699. Morgan-Kaufman.

Dyer, M. G., Flowers, M., and Hodges, J. (1987b). Computer comprehension of mechanical device descriptions. Technical report UCLA-AI-87-7, University of California at Los Angeles, Artificial Intelligence Laboratory. 1987.

Falkenhainer, B., Forbus, K., and Gentner, D. (1986). The structure-mapping engine. *Science*, pages 272–277.

Faltings, B. (1987). Qualitative kinematics in mechanisms. In *Tenth International Joint Conference on Artificial Intelligence*, pages 436–442. American Association for Artificial Intelligence (AAAI), Morgan Kaufman.

Forbus, K. (1984). Qualitative process theory. *Artificial Intelligence*, 24:85–168.

Forbus, K. D. (1985). Qualitative process theory. In Hobbs, J. R. and Moore, R. C., editors, *Formal Theories of the Commonsense World*, chapter 5, pages 185–226. Ablex Publishing Corporation.

Gentner, D. (1983a). Mental models. In Gentner, D. and Stevens, A., editors, *Mental Models*. Lawrence Erlbaum Associates, Hillsdale, N.J.

Gentner, D. (1983b). Structure-mapping: A theoretical framework for analogy. *Cognitive Science*, 7(2).

Goel, A. K. (1989). *Integration of Case-Based and Model-Based Reasoning for Adaptive Design Problem Solving*. PhD thesis, Ohio State University.

Granacki, J. J. and Parker, A. C. and Arens, Y. (1989). Understanding System Specifications Written in Natural Language. *Eleventh International Joint Conference on Artificial Intelligence*, pages 688–691. American Association for Artificial Intelligence (AAAI), Morgan Kaufman.

Granacki, J. J. and Parker, A. C. (1986). A natural language interface for specifying digital systems. In Sriram, D. and Adey, R., editors, *Applications of Artificial Intelligence in Engineering Problems*, volume 1, pages 215–226, Southampton, United Kingdom. 1st International Conference, Springer Verlag.

Granada Publishing, L. (1972). *How Things Work 1*. Granada Publishing Limited, London, UK.

Hammond, K. J. (1989). *Case-Based Planning: Viewing Planning as a Memory Task*. Perspectives in Artificial Intelligence. Academic Press, New York, NY.

Hayes, P. J. (1979). The naive physics manifesto. In *Expert Systems in the Microelectronic Age*, pages 242–270. Edinbugh University Press, Edinburgh, UK.

Hayes, P. J. (1985). The second naive physics manifesto. In *Readings in Knowledge Representation*, pages 467–486. Morgan Kaufmann.

Herskovits, A. (1986). *Language and Spatial Cognition*. Cambridge University Press, Great Brittain.

Hodges, J. (1989). Device representation for modeling improvisation in mechanical use situations. In *Proceedings of the 11th Annual Meeting of the Cognitive Science Society*, pages 643–650, Ann Arbor, Michigan.

Hodges, J. (1992). Naive mechanics: A computational model of device use and function in design improvisation. *IEEE Expert*, 7(1):14–27. cover article.

Hodges, J., Flowers, M., and Dyer, M. G. (1987). Knowledge representation for design creativity. In Liu, C. R., Requicha, A., and Chandrasekar, S., editors, *ASME Winter Annual Meeting*, volume PED-Vol. 25, pages 81–93. American Society of Mechanical Engineers, Boston, MA.

Hodges, J., Flowers, M., and Dyer, M. G. (1992). Knowledge representation for design improvisation. In Sriram, D. and Tong, C., editors, *Artificial Intelligent in Engineering Design*, volume I, chapter 6, pages 193–217. Academic Press.

Holyoak, K. and Glass, A. (86). *Cognition*. Random House, New York.

Johnson-Laird, P. (1983). *Mental Models*. Cambridge Unversity Press, New York.

Kahneman, D., Slovic, P., and Tversky, A. (1983). *Judgement Under Uncertainty- Heuristics and Biases*. Cambridge University Press, New York, NY.

Kannapan, S. and Marschek, K. M. (1989). An algebraic and predicate logic approach to representation and reasoning in machine design. Mechanical Systems and Design TR 212, University of Texas, Austin, Austin, Texas 78712-1063.

Kannapan, S. M. (1993). Functional Evaluation of Patentability. In *Proceedings of the 1993 AAAI workshop on Reasoning about Function*. July, 1993.

Kempton, W. (1986). Two theories of home heat control. *Cognitive Science*, 10(1):75–90.

Keuneke, A. M. (1989). *Machine Understanding of Devices Causal Explanation of Diagnostic Conclusions*. PhD thesis, Ohio State University.

Kintsch, W. (1986). Learning from text. *Cognition and Instruction*, 3:87–108.

Kintsch, W. (1987). Knowledge organization and text organization. *Cognition and Instruction*, 4(2):91–115.

Kintsch, W. (1988). The role of knowledge in discourse comprehension: A construction-integration model. *Psychological Review*, 95(2):163–182.

Kintsch, W., Kozminsky, E., Streby, W., McKoon, G., and Keenen, J. (1975). Comprehension and recall as a function of content variables. *Journal of Verbal Learning and Verbal Behavior*, 14:196–214.

Kintsch, W. and Van Dijk, T. (1978). Toward a model of text comprehension and production. *Psychological Review*, 85(5):363–394.

Klahr, D. and Wallace, J. (1976). *Cognitive Development: An Information Processing View*. Lawrence Erlbaum, Hillsdale, NJ.

Kolodner, J. A. (1984). *Retrieval and Organizational Strategies in Conceptual Memory: A Computer Model*. Lawrence Erlbaum Associates, Hillsdale, NJ.

Kuipers, B. (1986). Qualitative simulation. *Artificial Intelligence*, 29:289–388.

Lakoff, G. and Johnson, M. (1980). *Metaphors We Live By*. University of Chicago Press, Chicago, IL.

Lange, T., Hodges, J., Fuenmayor, M., and Belyaev, L. (1989). Descartes: Development environment for simulating connectionist architectures. In *Proceedings of the Eleventh Annual Meeting of the Cognitive Science Society*, pages 698–705.

Lebowitz, M. (1980). *Generalization and Memory in an Integrated Understanding System*. PhD thesis, Yale University. 186.

Lebowitz, M. (1985). RESEARCHER: An experimental inelligent information system. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pages 858-862. Los Angeles.

Lehnert, W. G. (1978). *The Process of Question Answering*. Lawrence Erlbaum Associates.

Lehnert, W. G. (1982). *Strategies of Natural Language Processing*, chapter 14, pages 375–414. Lawrence Erlbaum Associates.

Lenat, D. B. (1976). *AM: An Artificial Intelligence Approach to Discovery in Mathematics as Heuristic Search*. PhD thesis, Stanford University. STAN-CS-76-570.

Lenat, D. B. (1983). Eurisko: A program that learns new heuristics and domain concepts. *Artificial Intelligence*, 21(1,2):61–98.

Lenat, D. B. and Seeley Brown, J. (1984). Why am and eurisko appear to work. *Artificial Intelligence*, 23(3).

Lind, M. (1990). Representing Goals and Functions of Complex Systems: An Introduction to Multilevel Flow Modelling. Technical note 90-D-381, Institute of Automatic Control Systems, Technical University of Denmark. November.

Lind, M. and Larsen, M. and Osman, A. (1992). Applications of Multiple Flow Modeling. In *Proceedings of the ANP-92 International Conference on Design and Safety of Advanced Nuclear Power Plants.* Tokyo, Japan, Oct. 25-29.

Macaulay, D. (1988). *The Way Things Work*. Houghton Mifflin Company.

McCloskey, M. (1983). *Naive Theories of Motion*, chapter 13, pages 299–323. Lawrence Erlbaum, Hillsdale, NJ.

Metz, K. E. (1985). The development of children's problem solving in a gears task: A problem space perspective. *Cognitive Science*, 9(4):431–472.

Minsky, M. (1975). A framework for representing knowledge. In Winston, P., editor, *The Psychology of Computer Vision*, pages 211–277. McGraw-Hill, New York.

Minsky, M. (1981). A framework for representing knowledge. In Haugeland, J., editor, *Mind Design*, pages 95–128. MIT Press, Cambridge, MA.

Mostow, J. (1985). Toward better models of the design process. *AI Magazine*, 6(1):44–57.

Murphy, G. and Medin, D. (1985). The role of theories in conceptual coherence. *Psychological Review*, 92(3):289–314.

Narasimhan, S. and Navinchandra, D. and Sycara K. (1993). An Explicit Qualitative Representation of Function and Behavior for Conceptual Mechanical Design. In *Proceedings of the 1993 AAAI workshop on Reasoning about Function*. July, 1993.

Navinchandra, D. (1989). *Exploration and Innovation in Design: Towards a Computational Model*. Springer-Verlag, New York.

Navinchandra, D. and Marks, D. H. (1986). Intelligent use of constraints for activity scheduling. In Sriram, D. and Adey, R., editors, *Applications of Artificial Intelligence in Engineering Problems*, volume 1, pages 369–379, Southampton, United Kingdom. 1st International Conference, Springer Verlag.

Navinchandra, D., Sriram, D., and Kedar-Cabelli, S. (1987). The role of analogy in engineering problem solving. In *Second International Conference on Applications of Artificial Intelligence in Engineering*.

Navinchandra, D. and Sycara, Katia, P. (1989). A process model of experience-based design. In *The 11th Annual Conference of the Cognitive Science Society*, volume 1, pages 283–290, Hillsdale, NJ. Cognitive Science Society, Lawrence Erlbaum.

NAVY, U.S. (1971). *Basic Machines and How They Work*. Dover Publications, Inc.

Nielsen, P. (1989). The role of abstraction in place vocabularies. In *Proceedings of the Eleventh Annual Meeting of the Cognitive Science Society*, pages 267–274, Ann Arbor, Michigan.

Norman, D. A. (1983). Some observations on mental models. In Gentner, D. and Stevens, A., editors, *Mental Models*, chapter 1, pages 7–14. Lawrence Erlbaum Associates, Hillsdale, N.J.

Oswalt, W. H. (1976). *An Anthropological Analysis of Food-Getting Technology*. John Wiley & Sons.

Paris, C. (1987). Combining discourse strategies to generate descriptions to users along a naive/expert spectrum. In *proceedings of the 1987 international joint conference on artificial intelligence*, pages 626-632, Milan, Italy.

Pazzani, M. (1988). *Learning Causal Relationships: An Integration of Empirical and Explanation-Based Learning Methods*. PhD thesis, University of California at Los Angeles. UCLA-AI-88-10.

Pittges, J. and Eiselt, K and Goel, A. and Gomez, A. and Mahesh, K. and Peterson, J. (1993). Representation and Use of Function in Natural Language Understanding. In *Proceedings of the 1993 AAAI workshop on Reasoning about Function*. July, 1993.

Pylyshyn, Z. (1985). *Computation and Cognition*. MIT Press, second edition.

Qian, L. and Gero, J. (1993). Design Part Classification by Goal Achievement. In *Proceedings of the 1993 AAAI workshop on Reasoning about Function*. July, 1993.

Rajamooney, S., DeJong, G., and Faltings, B. (1985). Towards a model of conceptual knowledge acquisition through directed experimentation. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*. AAAI, Morgan Kauffman.

Rieger, C. (1976). An organization of knowledge for problem solving and language comprehension. *Artificial Intelligence*, 7(2):89–127.

Rieger, C. and Grinberg, M. (1977). The declarative representation and procedural simulation of causality in physics mechanisms. In *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, pages 250–255. AAAI.

Riesbeck, C. and Schank, R. (1989). *Inside Case-Based Reasoning*. Lawrence Erlbaum, Hillsdale, NJ.

Rivlin, E. and Rosenfeld, A. and Perlis, D. (1993). Recognition of Object Functionality in Goal-Directed Robotics. In *Proceedings of the 1993 AAAI workshop on Reasoning about Function*. July, 1993.

Rosch, E. (1978). *Principles of Categorization*. Lawrence Erlbaum Associates, Hillsdale, NJ.

Rubinstein, M. F. (1986). *Tools for Thinking and Problem Solving*. Prentice-Hall.

Sacerdoti, E. (1974). Planning in a hierarchy of abstraction spaces. *Artificial Intelligence*, 5(2):115–135.

Schank, R. (1973). Identification of the conceptualizations underlying natural language. In Schank, R. and Colby, K., editors, *Computer Models of Thought and Language*. W.H. Freeman, San Francisco, CA.

Schank, R. (1975). *Conceptual Information Processing*. American Elsevier, New York.

Schank, R. (1982). *Dynamic Memory: A Theory of Learning in Computers and People*. Cambridge University Press, New York.

Schank, R. (1986). *Explanation Patterns: Understanding Mechanically and Creatively*. Lawrence Erlbaum, Hillsdale, NJ.

Schank, R. and Abelson, R. (1977). *Scripts, Plans, Goals, and Understanding*. The Artificial Intelligence Series. Lawrence Erlbaum, Hillsdale, NJ.

Schank, R. C. and Birnbaum, L. (1980). Memory, meaning, and syntax. Research Report 189, Department of Computer Science, Yale University.

Schmalhofer, F. and Glavanov, D. (1986). Three components of human understanding a programmer's manual: Verbatim, propositional and situational representations. *Journal of Memory and Language*, 25:279–294.

Sembugamoorthy, V. and Chandrasekaran, B. (1986). Functional representation of devices and compilation of diagnostic problem-solving systems. In Kolodner, J. and Riesbeck, C., editors, *Experience, Memory, and Reasoning*, chapter 4, pages 47–73. Lawrence Erlbaum Associates.

Skorstad, G. and Forbus, K. (1989). Qualitative and quantitative reasoning about thermodynamics. In *Proceedings of the Eleventh Annual Meeting of the Cognitive Science Society*, pages 892–899, Ann Arbor, Michigan.

Sriram, D. (1987). *Knowledge-Based Approaches for Structural Design*. Computational Mechanics, Southampton, UK.

Sriram, D. and Maher, M. (1986). The representation and use of constraints in structural design. In Sriram, D. and Adey, R., editors, *Applications of Artificial Intelligence in Engineering Problems*, volume 1, pages 355–368, Southampton, United Kingdom. 1st International Conference, Springer Verlag.

Stahovich, T. and Davis, R. and Shrobe, H. (1993). An Ontology of Mechanical Devices. In *Proceedings of the 1990 AAAI workshop on Qualitative Vision*. July, 1993.

Stark, L. and Bowyer, K. (1990). Achieving generalized object recognition through reasoning about association of function to structure. In *Proceedings of the 1993 AAAI workshop on Reasoning about Function*, pp 137-141, 1990.

Sticklen, J. (1987). *MDX2: An Integrated Medical Diagnostic System*. PhD thesis, Ohio State University, Columbus, OH. Department of Computer and Information Science.

Swinney, D. (1979). Lexical access during sentence comprehension: (re)consideration of context effects. *Journal of Verbal Learning and Verbal Behavior*, 18:645–660.

Swinney, D. (1981). The process of language comprehension; an approach to examining issues in cognition and language. *Cognition*, 10:307–312.

Sycara, K. (1987). *Resolving Adversial Conflicts: An Approach to Integrating Case-Based and Analytic Methods*. PhD thesis, Georgia Institute of Technology, School of Information andComputer Science.

Thagard, P. and Holyoak, K. (1989). Why indexing is the wrong way to think about analog retrieval? In *Proceedings of the Second DARPA Case-Based Reasoning Workshop*, pages 31–35.

Thomson, D. and Tulving, E. (1970). Associative encoding and retrieval: Weak and strong cues. *Journal of Experimental Psychology*, 86:255–262.

Thorndyke, P. (1977). Cognitive structures in comprehension and memory of narrative discourse. *Journal of Cognitive Psychology*, 9:77–110.

Tulving, E. (1972). Episodic and semantic memory. In Tulving, E. and Donaldson, W., editors, *Organization and Memory*. Academic Press, New York.

Tulving, E. (1983). *Elements of Episodic Memory*. Oxford Psychology Series. Oxford University Press, Oxford, UK.

Turner, S. and Reeves, J. (1987). The rhapsody manual. Technical report, UCLA Artificial Intelligence Laboratory, Los Angeles, CA. UCLA-AI-87-3.

Tversky, A. and Kahneman, D. (1983). *Causal Schemas in Judgements Under Uncertainty*, pages 117–128. Cambridge University Press, New York, NY.

Ulrich, K. (1986). The use of objects in problem-solving. Personal Communication.

Ulrich, K. and Seering, W. (1987). Conceptual design: Synthesis of systems of components. In *Intelligent and Integrated Manufacturing Analysis and Synthesis*, volume PED Vol. 25, pages 57–66. American Society of Mechanical Engineers (ASME), ASME.

Van Dijk, T. and Kintsch, W. (1983). *Strategies of Discourse Comprehension*. Academic Press, New York.

Wasserman, K. and Lebowitz, M. (1983). Representing complex physical objects. *Cognition and Brain Theory*, 6(3):259–285.

Waterman, D. and Hayes-Roth, F. (1983). An investigation of tools for building expert systems. In Hayes-Roth, Waterman, and Lenat, editors, *Building Expert Systems*. Addison-Wesley.

Weiss, H. (1983). *Machines and How They Work*. Thomas Y. Crowell.

Wickelgren, W. A. (1974). *How to Solve Problems: Elements of a Theory f Problems and Problem Solving*. W.H. Freeman and Company.

Wilensky, R. (1983). *Planning and Understanding*. Addison-Wesley, Reading, MA.

Wilensky, R. (1987). Some problems and proposals for knowledge representation. Technical Report UCB/CSD 87/351, Computer Science Division, University of California, Berkeley, CA.

Wilks, Y. (1975). An intelligent analyzer and understander in english. *Communications of the Association of Computing Machinery*, 18(5):264–274.

Winograd, T. (1972). *Understanding Natural Language*. Academic Press, New York.