AN APPROACH TO KNOWLEDGE ENGINEERING TO SUPPORT KNOWLEDGE-BASED SIMULATION OF PAYLOAD GROUND PROCESSING AT THE KENNEDY SPACE CENTER

by

Shawn McManus, Michael McDaniel
McDonnell Douglas Space Systems Company
Kennedy Space Center
P. O. Box 21233
Kennedy Space Center, FL 32815

ABSTRACT

Planning for processing payloads at the Kennedy Space Center (KSC) has always been a difficult and time-consuming task. With the advent of Space Station Freedom and its capability to support a myriad of complex payloads, the planning to support this ground processing maze involves thousands of man-hours of often tedious data manipulation. To provide the capability to analyze various processing schedules, McDonnell Douglas Space Systems Company-KSC is developing an object oriented knowledge-based simulation environment called the Advanced Generic Accommodations Planning Environment (AGAPE). Having nearly completed the baseline system, our emphasis in this paper is directed toward rule definition and its relation to model development and simulation. We focus specifically on the methodologies implemented during knowledge acquisition, analysis, and representation within the AGAPE rule structure. An example model is provided to illustrate the concepts presented. Our approach demonstrates a framework for AGAPE rule development to assist expert system development.

I INTRODUCTION

Space station payload ground processing at KSC requires a great deal of planning and analysis, and AGAPE was designed primarily to support this activity. As the system has matured, its capabilities have become quite robust, making the system adaptable to modeling development activity in a wide variety of domains. To highlight and clarify the discussions in the ensuing sections, the major sections will feature a discourse of the relevant topics, followed by two examples from KSC space station ground processing. A future facility at KSC, the Space Station Processing Facility (SSPF) will provide an example of the facilities under consideration in our work, and installing a payload into a rack will benchmark a typical processing sequence.

Some local KSC definitions are in order:

payload - in the context of this paper, this will mean any hardware processed at Kennedy to become part of Space Station Freedom

experiment - any user (i.e. non-system) payload

APAE - Attached Payload Accommodation Equipment, the equipment that allows an attached payload to interface with station resources

II SYSTEM OVERVIEW

AGAPE is an knowledge based, object oriented simulation environment. The system's objects are built in a frame structure, a convenient method to allow storage of object attribute values in structures called *slots*. The object oriented approach to programming defines objects into two categories, *classes* and *instances*. Class objects may have any level of descendents (children, grand-children), while instance objects can have no children. This parent-child relationship allows for *inheritance* of attributes and other structures to be discussed later. This hierarchy constructs a tree-like *genealogy* of objects, as seen in Figure 1.

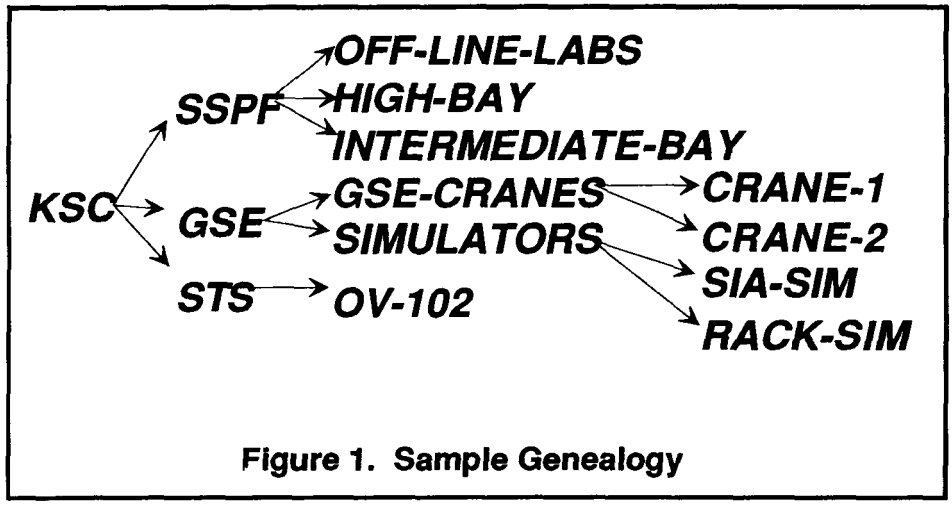

Figure 1. Sample Genealogy

The *knowledge base* resides in the HSKB, or *Hierarchical Segmented Knowledge Base*. The hierarchy comes from the object oriented nature of the system. Small portions of the knowledge base, called *rule sets*, are attached to appropriate objects (the segmentation of the HSKB), and these rule sets can be inherited from parent to child, just as with attributes. This structure differs from the typical expert system knowledge base, where all rules reside in a single unit. The segmentation of the rule sets allows the simulation to reason over only those portions of the knowledge germane to the situation.

The *script-based* simulation utilized by AGAPE allows each object (or for similar processes, groups of objects) to contain its own activity capabilities and requirements. These scripts are built from three basic types of activities; *task* activities, such as servicing or testing, *transport* activities, to move an object from one location to

another, and *installation/deintegration* activities, where an object installs itself in or removes itself from another object. The scripts also allow objects to define resources needed for a particular activity, such as technicians and lifting devices for a transport activity. These resource requests also provide a method for the script to interact with the HSKB, allowing certain attribute requirements to replace specific object requests. To move a 13,000 pound payload, for instance, the script may call for a specific crane, or it may require any lifting device with a capability of more than 6.5 tons.

The *simulation* in AGAPE offers an animation feature. The specific processing areas can be viewed during the simulation. As the simulation begins, the modeler may wish to view processing within the SSPF. If certain activities are of interest, like the weight and center-of-gravity test area in the high bay, that object can be selected during the animation, and the display will focus on that particular area.

III KNOWLEDGE ACQUISITION AT KSC

3.1 GROUND PROCESSING AT KSC

KSC's function in the NASA payload community is generally considered to be that of a payload integration and test center, with other centers around the country performing the bulk of the operational and strategic planning. While this is true in many cases, a considerable amount of effort occurs regularly at KSC in creation and development of work plans and procedures for present and future NSTS operations. Space Station Freedom has increased the planning duties at KSC to a large extent, due to the number of unique pieces of flight hardware scheduled to pass through Kennedy's processing facilities. Because the planning for Freedom involves much data manipulation to produce models such as facility and equipment utilizations, it was determined that a knowledge-based simulation environment would provide a valuable resource in planning for many programs.

Payload ground processing at KSC involves a series of complex, tightly controlled assembly and test procedures, as the *launch package* is assembled. (the launch package is the set of payloads for one mission, assembled as they would appear in the orbiter's payload bay. As the payload arrives at the center, it is received and inspected for any shipping damage. The hardware is then taken to an assembly area for build-up to its on-orbit configuration. The payload is tested to assure its functionality, and then verified with the space station hardware it will interface with (rack, APAE, etc). All payloads are tested with space station systems to assure compatibility with such items as power distribution and the Data Management System (DMS). The payloads for a single mission are then assembled into a launch package, and interfaces with the orbiter are verified. The assembled launch package is finally installed into the orbiter and launched. The flow outlined typically lasts from three to eighteen months.

One of the major planning and analysis tasks being performed today at KSC involves the design and review of a new facility, the Space Station Processing Facility (SSPF). The reviews include standard items, such as power outlets and office sizes. Because of the limited resources to process the myriad of Freedom hardware, the review process also includes many utilization studies to assess the impacts of changes in the program and its material.

There are many thousands of single operations required to process any payload at KSC. To adequately model a given process, the proper modeling scope is required to define the precision of the simulation's products. The process of preparing a flight rack to accept a payload can be laid out in its exact detail (hundreds of steps), the major processes can be defined (10-12 activities), or the overall process can be laid out in one to two steps. The more detailed the model, the smaller the scope of the simulation needed to provide an accurate and meaningful representation. If the modeler wishes to simulate the entire 20-mission build-up of the space station, for example, it is obvious to use the very highest level of overview (the fewest steps). This forces the simulation to concentrate on the overall utilization of facilities and equipment, the obvious point of such a large model.

3.2 OUR MODEL - FLIGHT OF-1

In order to more effectively delineate the processes occurring in the knowledge base development for AGAPE, one of the Freedom assembly flights, OF-1 (OutFitting flight 1, number 6 in the series), was chosen to act as a candidate mission. This flight information is currently taken from the August, 1988 Space Station Trial Payload Manifest (TPM), the most accurate set of flight information available for space station planning. This mission will allow a demonstration of the types and quantities of information necessary to support KSC process planning.

Flight OF-1 has many payloads of several types (see Table 1). The station payloads include a hand and eye wash station, a glovebox for utilization by experiments, and payload racks. The user payloads found on the manifest include some science payloads (e.g. SAAX 307X, life science 1.8m centrifuge), technology development payloads (e.g. TDMXF, fluid behavior experiment), and commercial endeavors (COMM 1243, Electroepitaxial crystal growth).

Although the various payloads employ the same basic interfaces to the space station, many require modified or unique attributes to their processing flows due to owners' preferences or singular payload characteristics. As can be seen in the TPM, there are several items bound for the space station on each flight, all with some level of processing occurring at KSC. Multiply this example by the 20 assembly flights necessary to complete the station, and the enormity of the planning and scheduling at KSC becomes apparent.

Another variable has been added to the equation with the creation of the Payload Integration Centers (PICs), centers around the country able to perform some of the space station processing that is KSC's

TABLE 1. MANIFEST OF SPACE STATION FREEDOM FLIGHT OF-1	
STATION SUPPLED EQUIPMENT	
Manifest Code	Name
RACK	Payload Racks - 16
USLAB	General Lab Support Equipment
USLAB	PMMS Process Fluids
USLAB	Customer Thermal Control System
USLAB	Commode/Hand & Eye Wash
USLAB	ECLS (Hygiene Water)
USLAB	PMMS Ultrapure Water
USLAB	MPS Glovebox
USER SUPPLIED EQUIPMENT	
Manifest Code	Name
TDMX2411	Advanced Adaptive Control
COMM1255	Commercial Organic & Polymer Processing
LOG	Payload Consumables
COMM1243	Commercial Electroepitaxial Crystal Growth
SAAX401C	Modular Containerless Processing Facility
TDMXF	Two-Phase Fluid Behavior
COMM1230	Commercial Crystals by Vapor Transport
SAAX307X	1.8m Centrifuge, Animal Holding Facility
SAAX401A	Space Station Furnace Facility

responsibility. The impacts from these centers has yet to be fully understood, and KSC is responsible for the validation and acceptance of their processing capabilities. Since PICs are in the early concept stage of development, it behooves KSC modelers to make preparations for acceptance of these new processing facilities.

IV KNOWLEDGE ACQUISITION

Numerous information sources exist at KSC for outlining *current* standard processing flows. Operations and Maintenance Instructions (OMIs) and Work Authorization Documents (WADs) are two of the sources available to the studious KSC knowledge engineer. The difficulty in determining Freedom payload processing characteristics stems from the high rate of change of space station resource information. The on-orbit resource types and locations differ almost from day to day. The challenge for the modeler, as shown in Figure 2, is to find the information, as it exists at any given time, and compile it into a meaningful and realistic model of space station processing.

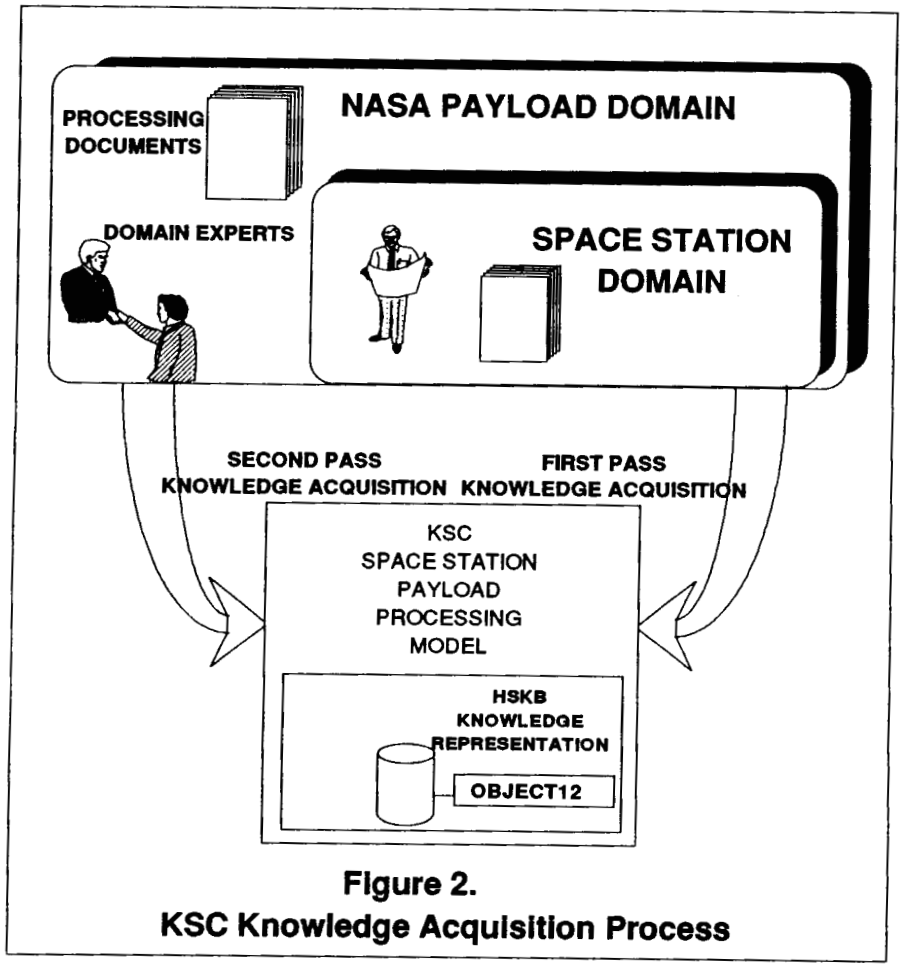

Figure 2.
KSC Knowledge Acquisition Process

4.1 ACQUISITION. Because the models being developed represent a program still in its infancy, many of the documents needed to complete the model have not been created. These new space station documents, however, are nearly all based on one or more *existing* documents used to support NSTS processing. For those items in the model requiring a nonexistent document, relevant current documents can be obtained and used as a basis. Generally, once the space station equivalent of the document has received final approval, the changes necessary to update the 'old' model are relatively minor, and quickly implemented and verified. The benefits of developing a model in an object-oriented environment become apparent at this time, as one change propagates through many descendents, making the change almost instantaneous.

At the time of this writing, the SSPF is undergoing tertiary review of its design. As such, specific items like equipment locations and utility port sizes and types are in a nearly constant state of flux. This makes the modeler's job rather difficult, because attribute values are constantly in need of revision. The basic layout of the building, on the other hand, has remained nearly constant over the past few months.

The data for the SSPF comes from two main types of sources, written and personnel. Documents are released in increments as the review process develops, so changes are manifested only when the next document revision is released. Data from the personnel contacts change almost daily while the review process continues. This makes for frustration as a knowledge engineer, since values and concepts are modified from minute to minute, depending on the person being interviewed. Our experience shows that using values from the documentation leads to the fewest problems. Generally, the people involved with the design reviews know when a change in the documents is about to occur, but these alterations should only be incorporated after they have been approved by the appropriate personnel.

Ground processing data and procedures are being formulated at this time. All these documents depend a great deal on the actual configuration of space station hardware, which remains in a state of constant change. The data presented is based on the most recent versions of the information.

4.2 ORGANIZATION. Because of the characteristics of the HSKB utilized by AGAPE, the organization of data easily falls into a familial grouping, much in the same manner as object-oriented programming. For the most part, rules necessary to accomplish a simulation follow the payload type groups: Payloads in general have different scripts than GSE, rack payloads use different rules than attached unpressurized payloads, etc. Similarities in certain aspects of processing allow for use of the same rules, provided that the rules were written in a generic fashion.

4.3 REPRESENTATION. The method by which knowledge is input into the AGAPE system tightly constrains the method of knowledge representation, but in no way limits the modeler in developing a useful and accurate model. Because of the breadth of information necessary to model KSC as a payload processing center, many rules are required for each aspect of payload integration. These rules tend to apply to only one or two types of payloads, or to experiments rather than elements, and thus the knowledge base grows at rather an astounding rate.

The grouping of rule sets is actually controlled in a large manner by the architecture of the system. Unlike traditional expert systems, the reasoning does not involve the entire knowledge base. Instead, as shown in Figure 3, the simulation reasons over only that part of the knowledge attached to the scripts under consideration. This automatic segmentation of the knowledge occurs in such a way that the modeler often fails to notice.

As an example of the representation of the knowledge gathered for the models at KSC, a rule is needed to acquire a crane in the SSPF to relocate a payload as it moves through its processing activities. This rule looks like:

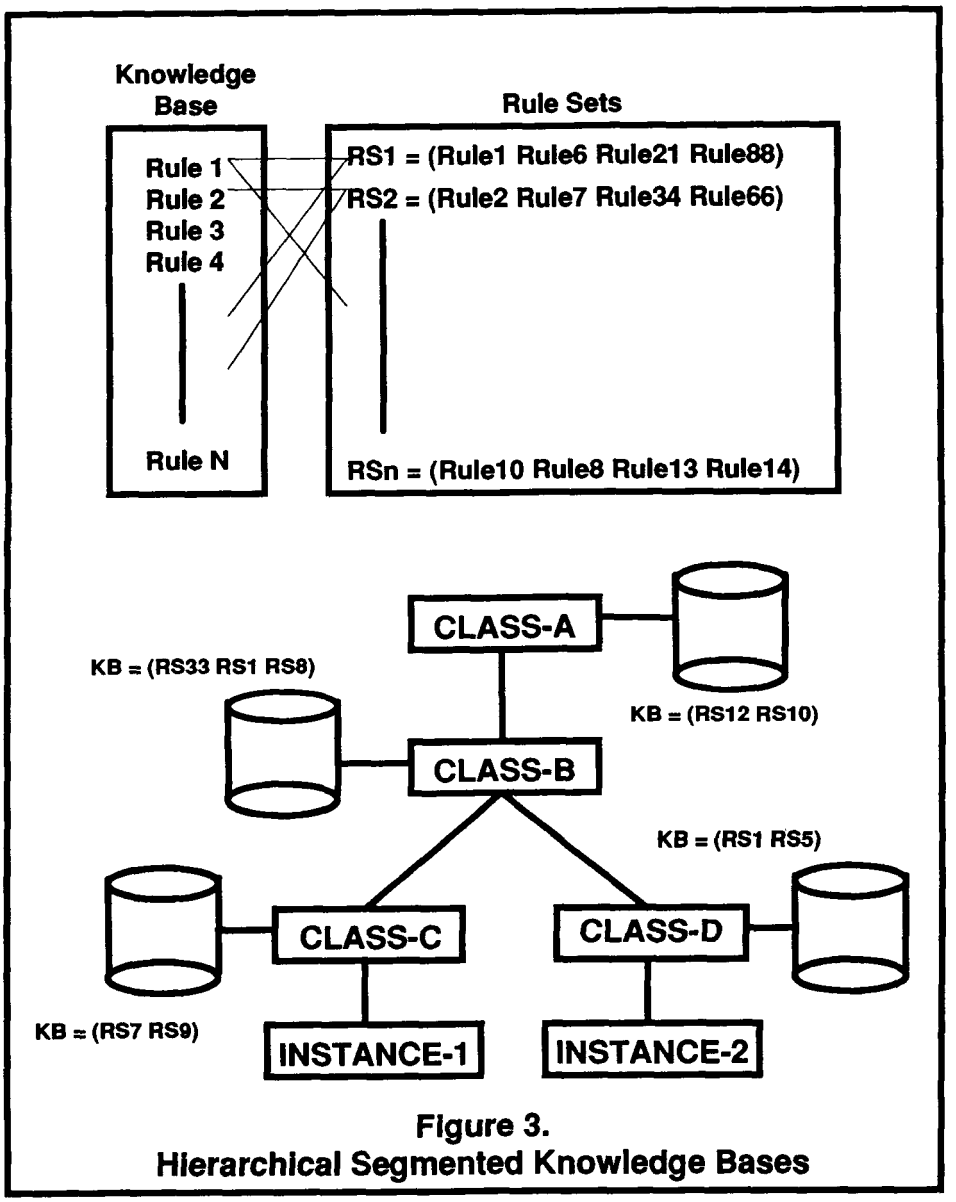

Figure 3.
Hierarchical Segmented Knowledge Bases

```
(define-hskb-rule P1
   :lhs (ACQUIRE-CRANE ?PAYLOAD ?CRANE
   :rhs ((BIND-IN-LIST ?CRANE (ASK 'GSE-CRANES CHILDREN))
         (> (ASK '?CRANE REQUEST 'LIFT-CAPACITY 'IS)
            (ASK '?PAYLOAD REQUEST 'MASS 'IS))
         (RETURN-VALUE ?CRANE))
```

This rule finds a crane whose lift capacity is simply greater than the mass of the payload. With the backward chaining process used in reasoning, the left-hand side (:lhs) or *consequent* of the rule is true if the right-hand side (:rhs) or *antecedent* is satisfied. All the statements in the rhs of the rule must be satisfied for that side to be true. In the case of this rule, a crane object (variable ?crane) is capable of lifting a payload if the rhs is complete. The BIND-IN-

LIST command assigns the ?crane to each of the children (first-order descendents) of the class GSE-CRANES. Each of the cranes is queried to determine the lift capacity of the device *(ASK '?CRANE REQUEST 'LIFT-CAPACITY 'IS)*. This is compared to the mass of the payload *(ASK '?PAYLOAD REQUEST 'MASS 'IS)* to assure a simply greater-than relation. The first crane to satisfy these requirements is returned to the caller *(RETURN-VALUE ?CRANE)*.

This rule could be expanded to include function calls or calls to other rules, to assure the accuracy of the model. By replacing the greater-than *(>)* line above with:

```
(BIND ?X (ASK '?CRANE REQUEST 'LIFT-CAPACITY 'IS)
(CRANE-CAPACITY (ASK '?PAYLOAD REQUEST 'MASS 'IS) '?X)
```

This section would call a function or a rule *CRANE-CAPACITY* to calculate the proper amount of over-capacity for a crane to safely lift a payload. For example, if KSC requires the lifting device to exceed the payload's mass by 40%, or use a more complicated formula, or tabular values, the rule could accommodate the requirements.

Figure 4 contains a copy of the rule editor screen for AGAPE. The rule set shown checks a rack to assure that a payload will conform to the proper type and dimensions. The pressurized-module-code refers to the on-orbit residence of the rack payload (US Laboratory module, ESA, Japanese).

These rules can be utilized at many points in the simulation. Any time in a script that a certain rule set is needed, an HSKB message can be attached to activate rule usage. To get a crane to relocate a payload within the SSPF, for instance, a message could be used. The text of the message would appear as:

```
(DEFINE-HSKB-MESSAGE 'PAYLOAD 'MESSAGE26
      ()
      (ACQUIRE-CRANE ?SELF ?CRANE))
```

The rule *ACQUIRE-CRANE* is passed the values SELF (the payload calling the message) and CRANE, which will contain the crane returned by the rules. This will allow the script to acquire a crane to perform its current activity.

Figures 5 and 6 display the code which expands to write rules and messages. This code, along with the rest of AGAPE, is public domain software available in COSMO.

V CONCLUSION AND FUTURE RESEARCH

As the models under development at KSC become more robust and flexible, many varied processes could be simulated. The current processing of STS payloads could be studied, along with the processing of the shuttles themselves. KSC's budgetary cycle could also be modeled. Impacts of future NASA programs, such as lunar base concepts

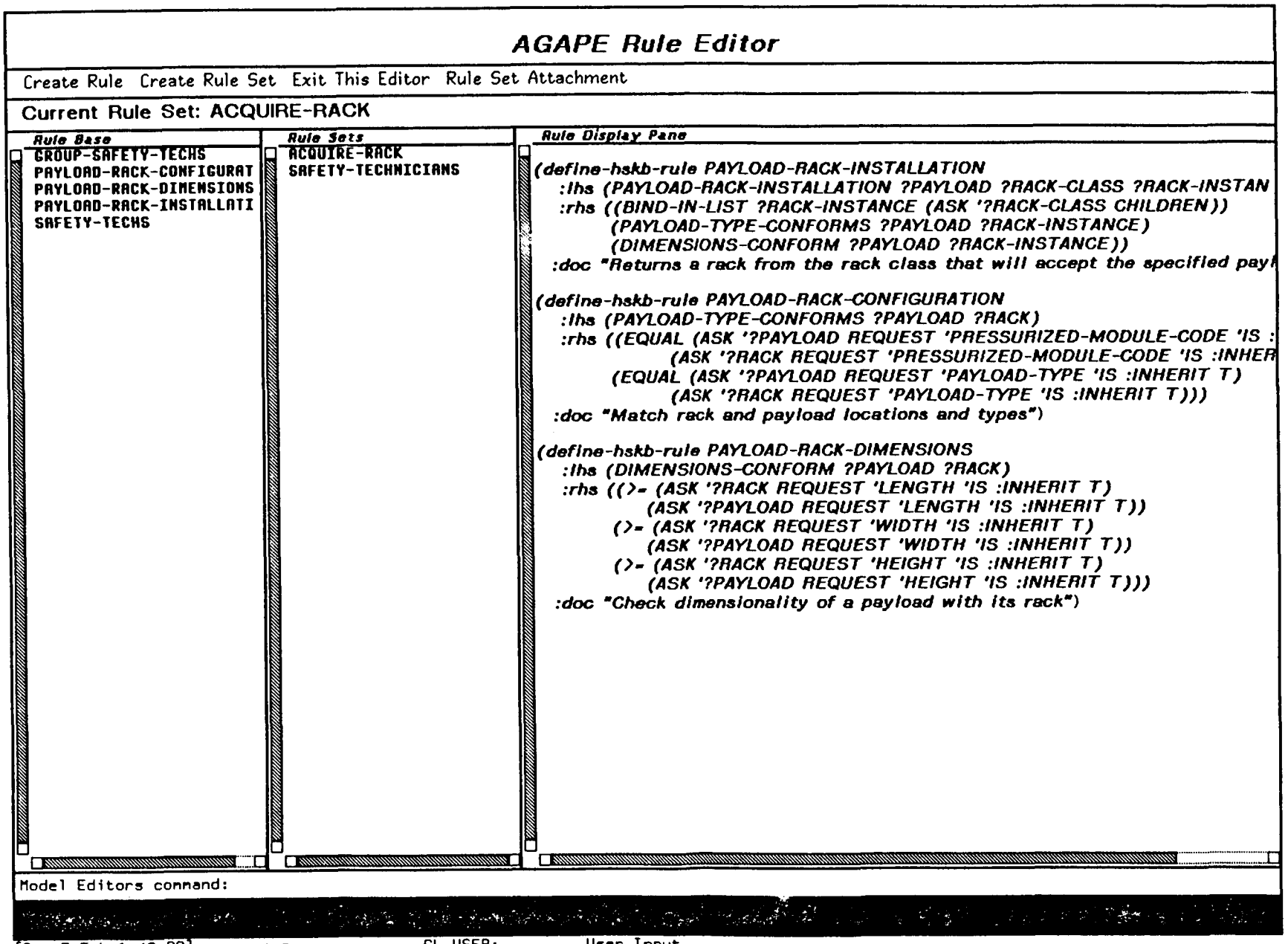

Figure 4. AGAPE Rule Editor With Rules Shown

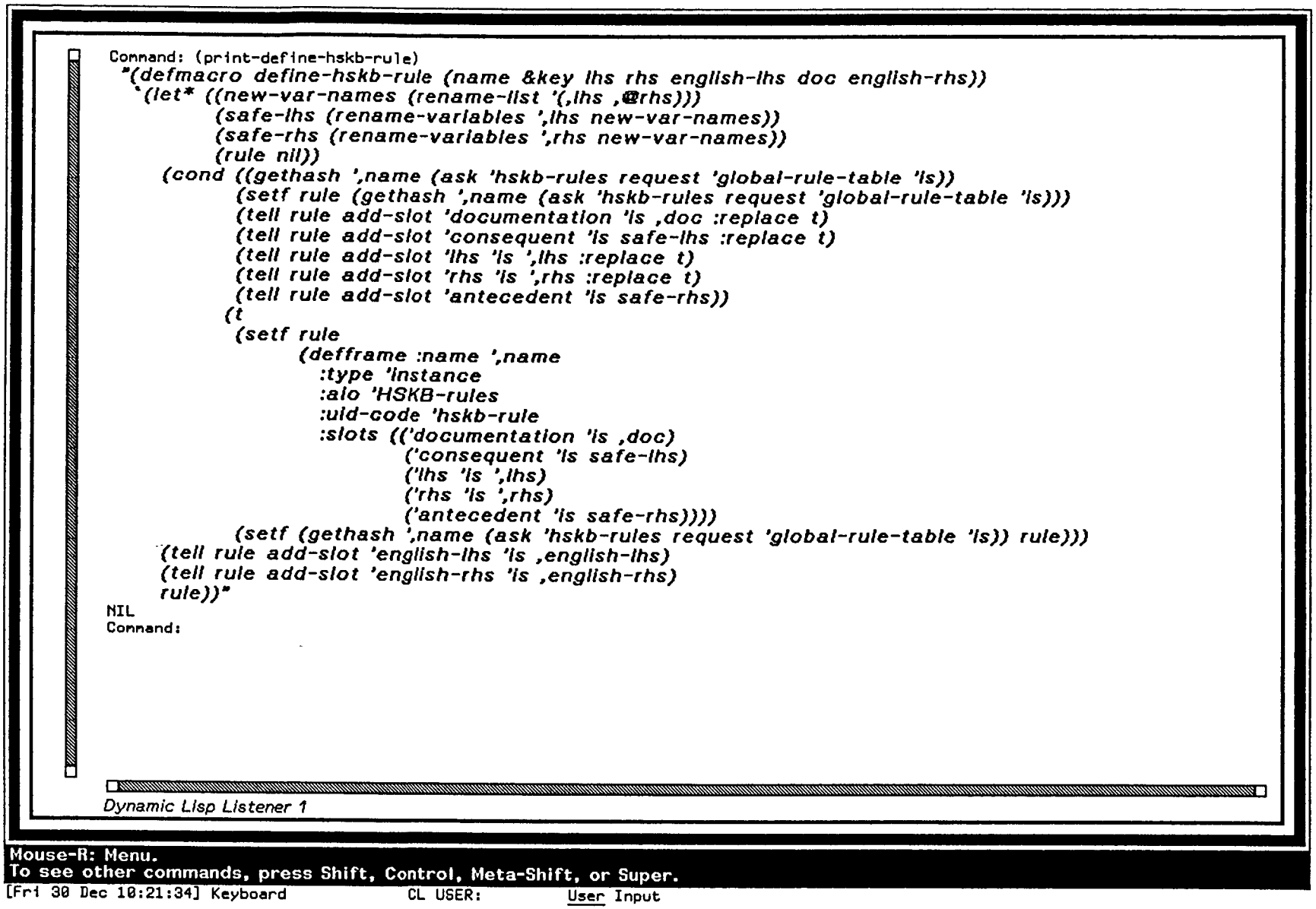

Figure 5. Expanded Rule Macro

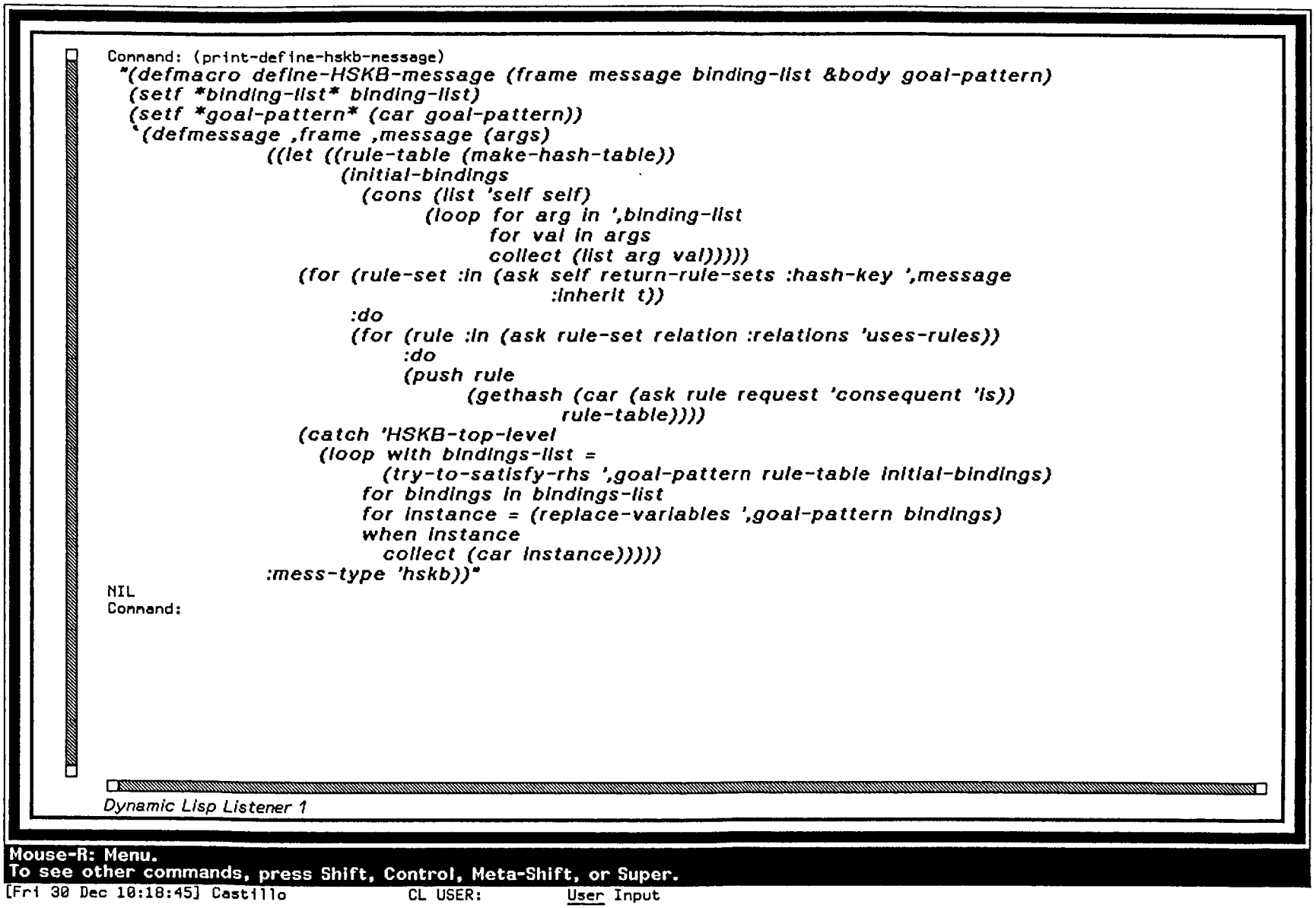

Figure 6. Expanded Message Macro

and planetary missions, could be assessed to determine long-range goals.

With the user-oriented capabilities of the AGAPE system, coupled with its robustness, focus the process of knowledge engineering into a single, continuous technique. Allowing the models to be developed *directly* by the people performing the payload processing brings the actual knowledge one step closer to the simulation. This system could assist many studies at KSC, not to mention processing scenarios at other NASA centers.