LARGE HUMAN-MACHINE INFORMATION SPACES*

Chuck Rieger, Richard Wood, Elizabeth Allen

Maryland Artificial Intelligence Group
Department of Computer Science
University of Maryland
College Park, Maryland 20742

ABSTRACT

An I-SPACE is a large human-machine information space, designed especially for environments where large amounts of distributed information, and tools for using that information, must be interfaced to a wide variety of users in real time. The important characteristics of an I-SPACE are that it presents a powerful, but uniform and simple interface to all users, that it permits users to synthesize information from diverse parts of the I-SPACE for simultaneous viewing and manipulation via their screens, and that it accommodates a cluttered desktop environment, where activities can be set aside (but kept up to date by real-time mechanisms), then later returned to. We describe the Goddard I-SPACE, a prototype system for Goddard Space Flight Center, an archetype of organizations in need of an I-SPACE. This 1-SPACE is presently running on a VAX, and will soon incorporate a large multiprocessor to handle real-time scheduling and event-driven pattern matchers.

1- The I-SPACE Concept

Information growth in modern society is outpacing the individual human s ability to access and synthesize information. While this problem applies to all segments of society, it is most noticeable in large commercial and governmental organizations, where teams of both technical and non-technical professionals must work cooperatively toward goals whose solutions demand the coordination of substantial amounts of information in a timely fashion. The very existence of such large societal entities and their often ambitious goals can be attributed, in part, to computers and advanced software tools. It seems appropriate, therefore, that computers be applied toward the solution of problems which result from this information explosion.

We describe here the I-SPACE project, a human-machine system designed to interface both technical and non-technical professionals to „the large, dynamic, and diffuse information space" in which tney conduct their daily professional activities. This project shares some goals with the PIE project [Goldstein 1960a] and the ZOC project [Newell 1977], both advanced human-machine information systems. However, in our project, more emphasis is placed on the real-time acquisition and synthesis of information from diverse, often geographically distributed sources.

Goddard Space Flight Center, an example of an organization with needs in these areas, is the environment for which we are designing the current I-SPACE. . ,At this center, as many in 20 data proceasing "empires" engage in project-specific activities involving the engineering, science, and administration of near-earth satellites. Data is everywhere: science databases, engineering information (some on-line, some not), administrative scheduler, system documentation, experiment schedules, and so forth. While each empire runs adequately, there is no central information hub which allows, for example, a bewildered visiting scientist to gain a grasp of what science data is available, or a Goddard administrator to ascertain the adrainistrative or technical status of some subproject. The dynamic nature of Goddard s mission operations, and the need for timely access of information, magnify the basic problem of a human's inability to deal with massive amounts of information.

The I-SPACE is a prototype information hub for Goddard. By logging into thi3 hub, a user enters the Goddard information space, a potentially vast collection of frames (akin to Smalltalk's classes and objects [Ingalls 1976|, as used in the PIE system I [Goldstein 1980b). Regions of *the* I-SPACE deal with various aspects of Goddard s operations (e.g., science, engineering, specific projects, administration, maintenance, experiment scheduling, specific spacecraft), and the user roams through this space intelligently assisted by the system. As he proceeds, the current frame generates a high resolution CRT representation of itself as a form with a collection of slots (the specific contents of the frame). When he reaches a frame. of interest. he requests that the I-SPACE system "focus' that frame. As we describe below, a focused frame puts the user in real-time touch with that frame's contents, providing in effect a constantly updated, synthesized view of selected I-SPACE information relevant to 3ome task. "Invoking" a focused frame's slot is tantamount to requesting that some computation be performed, so that the 1-SPACE is in fact a programming environment as well as an information system. In effect, the I-SPACE replaces the conventional operating system.

In the sections below, we describe the system's conceptual design, hardware and software architectures, human-machine interface, and current and potential qualities as an artificially intelligent manager of large amounts of information.

2. Conceptual Design

The main goal of an I-SPACE is to deliver a simple interface through which the user gains screen access to all information and tools appropriate to the solution of his task at hand. A second goal is that information be delivered in a dynamic fashion, which often means real-time. The user must be given the illusion of looking through powerful windows at data that may be changing fey the second or minute. At Goddard, such data could be orbital or engineering data from a spacecraft, data being routed from one administrator's screen to another's screen, newly updated science data, a bulletin about a hardware failure or schedule change, and so forth. The user must be given direct conduits to and from the information sources with which he must deal in performing his job.

An interesting image comes to mind at Goddard, where there are already many pools of information individually managed by existing equipment.

* This research was supported by the Goddard Space Flight Center under contract iNA35-25764.

Suppose there were a room at Goddard whose walls were covered with information conduits flowing to every reach of the organization. Suppose also that a user enters the room and, using "patch cords', simultaneously taps into some arbitrarily chosen set of conduits and interacts with the information sources at the other ends of those conduits in a wholistic fashion. For example, information received from various conduits might be combined to derive certain synthesized information, or information might be sent to several conduits to effect a complex task. Further suppose that each specific setup of patch cords is named, saved, and later reinstalled automatically to bring the user back to the same information context when he next enters the room. Finally, ima-?ine that many users are in this room, all simul- aneously patching to and sharing seta of conduits with minimal interference.

Would not such a system be a foundation for a very intelligent distributed information system? And would not such a system represent an incremental advance in interfacing humans to large computer-based information systems? We, of course, believe that both answers are yes!

The I-SPACE conceptual design follows this image closely (Fig. 1). The "room' is the central I-SPACE computer, currently our Departmental VAX 11/780 running C and LISP under UNIX. The walls filled with information conduits, which we call the I-BANK, will be realized by ZMOB [Rieger 1981] (Appendix), a 256 processor research computer under current construction by our AI group (ZMOB is currently being simulated by UNIX processes). The collections of patch cords that route, synthesize, and present information from the conduits are the I-SPACE frames. The ability for many users to use the room simultaneously translates to our implementation of the I-SPACE as a UNIX shell which coordinates several UNIX processes per user.

2.1. I-SPACE Frames

I-SPACE frames are LISP data structures that contain all the information required to set up, maintain, synthesize, and display information from a set of information "conduits" on regions of the user s screen. Structurally, a frame is a collection of named slots. Each frame corresponds to an activity, or local environment, upon which a user might wish to focus his attention. Each slot represents an aspect of the frame's environment, and generally will be responsible for bringing a tool, service, or information to the user via a region of his display screen. Visually, a frame resembles a form-like description of an associated I-SPACE activity. It is presented by displaying all its slots at an appropriate level of activation for that user. Each slot is presented as a three window display: the slot name, the slot value or work area, and a status region. The shapes and locations of windows on the screen, and whether or not all three are displayed at the current level of activation are initially determined by information in the frame's representation.

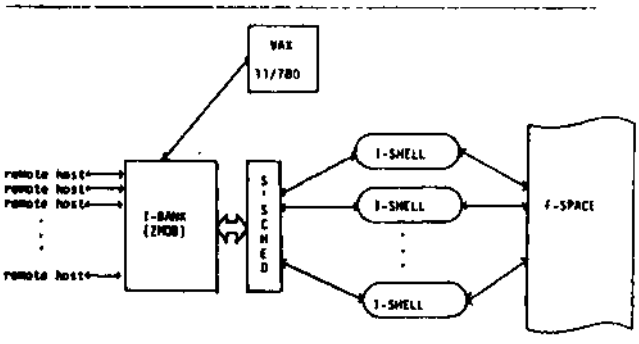

Figure 1. I-SPACE Overview

Categorically, I-SPACE frames include:

MENU FRAMES: Frames whose slots will move the user to other frames in the region of the I-SPACE where the user is presently situated.

REAL-TIME CONTROL FRAMES: Frames whose slots give the user access to real-time information about dynamic regions of the I-SPACE.

INFORMATION FRAMES: Frames whose slots provide windows into external databases.

DOCUMENTATION FRAMES: Frames that document Features and capabilities of the I-SPACE itself.

SYSTEM TOOL FRAMES: Frames that give the user access to standard system tools, such as editors, higher level languages, and utility packages.

I-SPACE DEVELOPMENT FRAMES: Frames that allow the use? to" extend, modify, and create new I-SPACE frames and procedures, and that deliver 1-SPACE status and user information.

Within each category there is considerable diversity. For example, administrative plans and schedules, science data, engineering schedules, and building and grounds maintenance are all examples of information frames, found in different regions of the I-SPACE. Real-time interpersonal communications, satellite science and engineering control channels, and ground-based equipment monitors and sensors are all examples of real-time control frames that reside in various reaches of the I-SPACE. Furthermore, most frames will have characteristics of several of these categories, since the slots which comprise the frame may address different needs. For example, one user may find it convenient to define a frame whose slots are: (1) a screen-screen link with a colleague, (2) a real-time conduit to a satellite, (3) a window to an external database, (4) a FORTRAN program ramming environment, and (5) a window into a ocal piece of documentation. Focusing such a frame would set up all the necessary communications channels, then present the user with this miniature world, which keeps him in touch with several parts or his environment simultaneously.

Slots

The logic of a slot depends upon the nature of the task it is designed to perform. The slots of a menu frame are generally descriptions of other frames to which the user can move. The slots of a real-time frame associated with the attitude control of a satellite would have names like "Current Attitude". "Alter Current Attitude", and so forth. The slots of an information frame about an administrator s weekly schedule would have names like "Person For Whom Schedule Desired", "Week Desired", and "The Schedule". Slots in a system tool frame for a class of utilities, might be such things as Run File Transporter , Rewind Tape", and so forth. Slots of a higher level language tool such as LISP might be "Run The Interpreter , Debug Some Code . Load Saved Code , and so forth. Generally, slots will follow an imperative style of programming, and the invocation of a slot always means "do your thing for me' .

Frame and Slot Activation Levels

Three levels of frame/slot activation are possible: browse", "focus.'' and invoke . Up to hree "activation procedures"' can be associated with each slot of a frame, each procedure

corresponding to one of the three levels of activation. When the user initially brings a frame to his screen, the entire frame is activated at the browse level by individually executing each slot's browse level procedure (if any), which then performs a computation (typically one which displays the name of the slot In the name window and, perhaps, other information in the value window.) Since activation procedures can also be sensitized to the user s priority and clearance levels, different events may occur for different users when an activation procedure is applied. Typically, browse level procedures will attempt to present only enough information to give the user an overview of the frame s contents or capabilities, under the assumption that the user is not interested in frames he passes through in reaching his target frame in the 1-SPACE.

When he reaches a frame in the 1-SPACE with which he wishes to interact, the user focuses' that frame. This causes all browse level activation procedures superceded by focus level procedures to be terminated, and all superceding focus level procedures to be applied. Conceptually, this results in more detailed or accurate services being run at each slot.

In either browse or focus mode, the user can move to a certain slot and invoke it. If the invoke level activation procedure exists for that slot and if it is determined that the user is qualified to invoke that slot, the invoke level procedure supercedes the running browse or focus level procedure at the slot, typically resulting in some control or sensing event within the I-SPACE. When the invoke procedure relinquishes control, the slot is returned to its previous activation level. When the entire frame is left, it may be left either in a focused or defocused state (i.e., left active, or discarded), permitting the user to interact sequentially with several focused frames (which are visually overlaid and restored on the screen). As in PIE, the I-SPACE user interface provider for the 'cluttered desktop' display control, where partially occluded frames may actually still be running in focus mode and keeping themselves up to date visually even though they are not at the top of the pile.

Invoking an I-SPACE frame's 3lot is tantamount to computing. Since all activation procedures are unrestricted LISP functions, the effectB on the I-SPACE of an invoked slot can be arbitrary. Simple examples of invocation results are such tilings as one-time data acquisition and display (e.g.. a thermal value for a satellite, a fact from a aata base), and frame changing (e.g., by invoking a "menu" frame's slot, the user runs a procedure which moves to another frame). More complex examples include such activities as repetitive real-time data acquisition and display updating, and the running of an entire processor within the slot's value window. In this last case, invoking, say, the Run the Editor' slot of the current frame can cause the system editor to be invoked within the slot's value window. Hence, conventional computing is considered a subset of the larger 1-SPACE organization.

2.2. Example Scenario

In the context of the applications at Goddard, a sample session of a visiting scientist employing the I-SPACE to obtain data or effect experiments would proceed as follows. Since the scientist is not an experienced I-SPACE user, he depends on the system s expertise to assist him in locating the desired information and accomplishing his tasks. The flavor of a session with an experienced user differs from the following scenario in that users familiar with the I-SPACE can move quickly from one frame to another without passing through intermediate frames.

Initially the system presents to the scientist a series of MENU frames which describe the information available in the Goddard 1-SPACE. When each frame is browsed the title of an available category of information is displayed. Focusing the slot associated with a category produces a brief, but more detailed description of the category. When a slot in a MENU frame ia invoked, the scientist moves through the I-SPACE to another frame, presumably more specific than the previous one, which is then browsed. Our scientist is interested in experimenting with the Space Telescope. He traverses the I-SPACE from general frames describing Goddard at large through 3cience-specific frames to the "Near Earth Satellites frame and arrives at the "Space Telescope" frame. Browsing this frame he discovers thart the following categories of information about the Space Telescope are . ,available: "Documentation", Satellite Status , "Experiment Data Base , Control Operations , and Personnel Contacts . Focusing on the 'Satellite Status" slot of this frame produces the information that the Space Telescope is operational and available for experimentation. The scientist then invokes the "Experiment Date Base" slot to browse through the results of previous experiments, discovers that the results he is interested in are not available and decides to perfonn the experiment himself. He returns to the Space Telescope frame to examine the available documentation and then moves from the Documentation frame to the Control Operations frame. The scientist then invokes the appropriate slot in the Control Operations frame intending to accomplish his goal, unfortunately the experiment doe3 not run, perhaps because scientist did not specify the task correctly. Returning again to the Space Telescope" frame, he invokes the "Personnel Contact" slot creating a screen-to-3creen conversation with the project manager, determines his initial error in operating the Space Telescope and retries his experiment. Now successful in his quest, our scientist movea on to other uses of the I-SPACE, perhaps investigating topics mentioned in other frames that were visitea during this session.

A few points of interest should be made. At no point in the scenario did our scientist realize that the information he requested, or the operations he invoked, might have been accomplished on remote hosts. Encoded in the procedures contained in the individual frames of the 1-SPACE is the expertise to accomplish any activity, be it locating pertinent data or performing specific calculations. Our scientist'8 screen-to-screen conversation may have been with a project manager located in the next office or an individual on a remote host. Information describing the current status of the telescope may have been stored in a data base, or it may have been the result of an actual request to the satellite via a controlling computer to report its current status. The individual I-SPACE frames determine the source of the information and via the I-SHELL schedule reauests to retrieve the information or execute procedures to produce the result.

<L'2.' I-SPACE heal-Time Scheduling and Processes

Consider the genera} character of an I-SPACE session from one user s point of view. At any given moment, the user will have a currently browsed or focused frame (some of whose slots may have been invoked), and possibly a collection of other focused and partially invoked frames in the background. Each slot of each of these frames has been activated at some level. If the slot s activation procedure is one which is concerned with repetitive or real-time updating, or is on the lookout for certain events or data in the I-SPACE at-large, then it will be making demands on the I-SPACE for periodic attention. For this reason, the system must include a scheduler process per user, as well as a system-level scheduler for mediating the user schedulers' requests of the I-BANK (i.e., ZMOB).

The architecture of the I-SPACE scheduling system is shown in Fig. 2. As the I-SPACE shell interprets user desires, and slots become activated, slot activation procedures run and send scheduling tasks to the O-SCHEDULfR, a separate process that manages its one user s scheduling needs. The U-SCHEDULEH in turn determines the frequency by which a procedure should be run and makes appropriate demands on the S-SCHEDULER. The S-SCHEDULER is a system-wide process which synthesizes all U-SCHEDULER requests to the I-BANK,

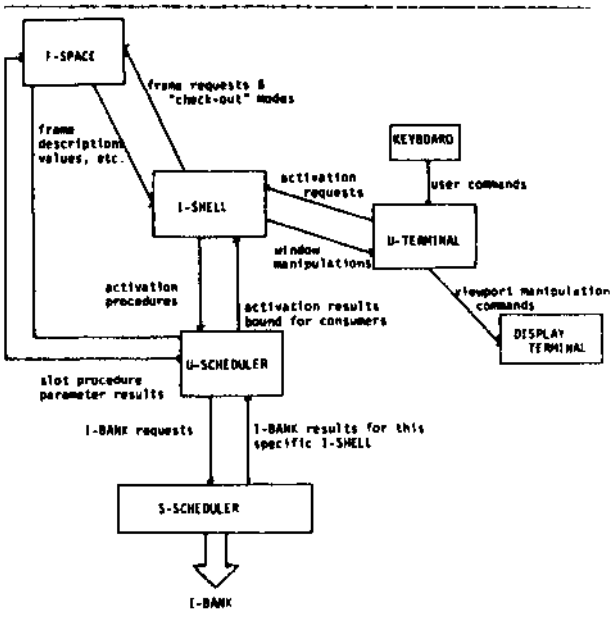

Figure 2. A Single I-SHELL

prioritizes them (from user priority and real-time requirements of the scheduled event), then passes them to the I-BANK. which finally carries out the requested interactions with the outside. Results are returned as named message packets, which are sorted out by the S-SCHEDULER and sent to the various requesting U-SCHEDULERS, which in turn route results to their I-SHELLs.

Status Information Display

Because of the design goal of presenting a user interface that has appearances of both a PIE-like programming/personal environment and real-time windows into dynamic regions of the I-SPACE, the I-SHELL must take care in how it processes responses. First, associated with any slot that displays real-time information, or that serves as a user comnand window to a real-time device somewhere in the I-SPACE, is a separate status or synchronization window. Using inforrnation—from the U-SCHEDULER, this § window is illuminated when the main window's real-time scheduling falls behind, and a real time clock begins displaying the delay time until the currently scheduled event is completed. For example, if the U-SCHEDULER has been instructed to acquire updated thermal information from a spacecraft every 30 seconds, and there has been a delay beyond this period, the synchronization window illuminates and begins displaying the overrun time until the data becomes available. On output, if, say, the user'has requested that text be sent to another host, and there are I-SPACE delays in getting the information out (and returning the confirmation that it was sent) beyond the time deemed reasonable by the sending slot, the status window will also be activated.

Multiple Active Frames

Another topic of interest that arises because of the I-SPACE's blend of a PIE-like environment with real-time scheduling is that of routing incoming real-time results to the user s display in meaningful ways. Because the user may have focused a frame that has given rise to recurring real-time scheduling events, then moved to another frame without having defocused the current one, the I-SHELL must properly manage the arrival of scheduled information associated with the set-aside, but not defocuaed frame (or frames). This requirement demands that the logic of each frame be isolated from its display manifestation, so that the I-SHELL can maintain a pool of currently running and up-to-date frames, even if not all are completely visible on the display screen, because of this separation of frame logic and frame display in the I-SHELL, the user is free to focus or invoke slots of a frame, then move on to another frame before those effects are fully felt. Visually, the effect is a cluttered screen of the current frame and other partially occluded frames, all of whose visible parts may be changing to reflect real-time updates from completed scheduled events. Additionally, completions of invoked slots for set-aside frames^ will be noted by a flashed message in the display's status area.

Since the logic of I-SPACE frames is what binds together the scheduling and user interface aspects of the I-SPACfc, we turn next to a description of the structure and contents of I-SPACk frames.

1' Organization of jthe 1-SPACE

2.'L' Frame Logic and Scheduling

A frame in the 1-SPACE is a LISP data structure containing a set of slots which correspond to each subactivity of a task in the 1-SPACE. A slot is a description of a particular aspect of a frame and contains the slot's name, value, display descriptor, activation procedures, and access channels. Associated with each slot are three levels of activation which correspond to the three ■m-o"aes-o"f processing in the 1-SPACE.

For each activation level (browse, focus or invoke), a slot specifies (I) the channel to be used, (2) the procedure to be run on that channel. (3) the parameters passed to the procedure, (4) the schedule by which the procedure is run. (5) the consumers that handle the results returned from tne procedures, and (6) the priority required to activate the slot at this leveT: SIoT specifications may be explicitly included in the frame or implicitly refer to system defaults. For instance, the absence of a channel indicates that the 3lot is to be executed by the I-SHELL itself (without interaction with the I-BANK). An example of this is the slot description for browsing the Space Telescope frame visited by OUT scientist during the scenario.

```
slot orbit
  browse null,
         consume ■ L>isplay( name, value) ;
```

This slot specification causes tfje contents of the name and value< windows of the orbit" slot to be sent to the user's terminal for Display. (The null procedure is used to indicate that no computation is necessary before executing the consumer function.) Another common slot procedure is MoveToFrame, the mechanism which moves the user to a different frame as the result of invoking *H* slot of a ilEMJ frame.

An 1-SPACE information conduit is called a channel. Channels access various objects in the external world (e.g., mainframe computers, satellite controllers, administrator's desktop computers and so forth) as well as the internal computing environment (e.g., UiJIX tools, such as, editors, high level languages and program development tools). Currently channels refer to specific resources (or connections) in the I-LAi\K, and have associated capabilities (or sets of procedures that can be executed on the channel). We are currently developing plans for decoupling the relationship of channel and procedure and replacing specific references to a channel with descriptions of the information requirements of the channel. This in turn will change the nature the S-SCHEDULEH to resemble more a problem-solver than a resource scheduler.

Each slot contains three window descriptors that specify the visual presentation of the slot's

name, value, and status windows to the user. This description includes each window's location (two-dimension subregion) within the frame and, possibly, the initial contents of the window. Typically, the name window contains only the slot s name. e.g., Near Earth Satellites , or Orbit of Satellite , the value window contains the slot s value as obtained by running one of the slot's procedures, and the status window contains information describing the current status of the running procedure. This includes messages such as: "Connecting to Remote Host", "Remote Host is Down', or Result Late, 5 sec". Both the value window and the status window of a slot are optional. The value window is not specified for slots that correspond to activities in the I-SPACE that do not return specific values, but are executed for their side-effects (e.g., slots in MENU frames). The status window is optional for slots that do not monitor recurring events.

A partial frame description corresponding to some of the activities our scientist may have viewed during the scenario is shown below:

```
frame SpaceTeleControlOper
type control

slot orbit
  browse null
         consume = Display(name,value);
  focus  GetOrbitInfo,
         channel = SpaceTeleChan,
         schedule = 10 sec,
         consume = Update(value)
                   & Display(value);
  name   "Orbit of Space Telescope",
         window descriptor = (3,1) (1,25);
  value  34000km 0d,
         window descriptor = (3,30) (1,25)
  endslot;

slot attitude
  browse null,
         consume = Display(name,value);
  focus  GetAttInfo,
         channel = SpaceTeleChan,
         schedule = 5 sec,
         consume = Update(value)
                   & Display(value);
  invoke AdjustAttitude
           (user ("Enter degrees (r,p,y):")),
         channel = SpaceTeleChan;
  name   "Attitude",
         window descriptor = (5,1) (1,25);
  value  22r 36p 15y,
         window descriptor = (7,1) (2,25)
  endslot

slot SpaceTeleFrame
  browse null,
         consume = Display(name);
  invoke MoveToFrame(SpaceTeleFrame);
  name   "Space Telescope Frame",
         window descriptor = (10,1) (1,25);
  endslot

endframe
```

The arguments passed to slot procedures are constants, values taken from a slot's value field in the I-SPACE, or supplied by the user. If an argument is used from a slot's value field, it may be taken from the current slot, or another slot in the current frame, or a slot in a different frame. This supports a form of communication between slots situated in different frames and can be employed to form complicated slot procedures that synthesize information from varying sources.

Each result of a procedure execution is passed to the consumers specified in the slot-Consumers are taken from a library of consumers, which includes procedures that update the contents of fields in a slot, that send requests to the U-TERMINAL to display the contents of a field, or that pass the new results to other slots in the frame or to slots in other frames. This, too, supports communication to other sections in the I-SPACE.

As an illustration of the frame-to-scheduler link suppose the following slot from the SpaceTeleControlOper frame above is focused:

```
focus GetOrbitInfo,
      channel - SpaceTeleChan,
      schedule ■ 10 sec,
      consume = Update(value)
                & Display(value)
```

This leads to scheduling activities that result in running CetOrbitInfo on SpaceTeleChan every 10 seconds by the S-SCHEDULED each time the —the orbit information is returned, the consumers are run. Update stores the value in the value window of the slot, and Display tells the U-TERMINAL to display the value window. The user then sees the orbit position updated every 10 seconds. If the orbit information is delayed or cannot be obtained the slot s real-time window is activated informing the user of the difficulty. In general, numerous scheduled slots such as this one will be running simultaneously as the result of user browse, focus, and invoke requests.

3.2. Frame Creation and Editing

At various times, the user may decide that he would like to create a new frame to perform a different set of operations or to synthesize and display a new set of information, or he may want to modify an existing frame (by making a copy of the frame and then changing it to meet his own needs). These operations are done through a special set of I-SPACE development frames which allow the user to edit frames and add new frames to the I-SPACE.

The I-SPACE editor accepts new frame definitions via a special description language akin to KRL [Bobrow 1977] called ISFL (the I-SPACE Frame Language [Allen 1981], the examples in this paper are written in ISFL). The editor aids the user by providing templates of the various types of frames in the I-SPACE. These templates contain default slot values as well as variables whose values are supplied by the user when he creates a new frame. The ISFL templates allow the user to enter frames into the system without undue concern with the ISFL details. This method of creating frames is similar to the one used in the BROWSE system of ZOG [Palay 1980]. The frame editor also provides access to libraries of procedures with channels and consumers from which the user may choose in creating hi3 frame. The editor also knows the syntax of the language and can direct the user in designing a frame without the use of a template. Thus, the user may create a frame completely from scratch or create a frame using one of the given templates. He may also create his own templates, specifying variables that will be supplied when a frame is created using the template. In order to edit an existing frame, the Reverse Parser automatically translates the LISP frame structure into ISFL and then invokes the I-SPACE editor. If a template was used in creating the frame, this template is once again used to aid the user in modifying the frame. After the user is finished editing a frame, the ISFL specification is translated by a special parser into its corresponding LISP structure and the frame is added to the 1-SPACE frame library.

3.3. The F-SPACE and Frame Librarian

The repository of all I-SPACE frames is called the F-SPACE. This library of frames and associated procedures is, in effect, the knowledge base for the 1-SPACE. The F-SPACE is managed by a separate process which is responsible for coordinating the insertion and modification of I-SPACE frames, and for overseeing the "checking out" of frames by individual user 1-ShELLs as their users roam the space. As the central I-SPACE librarian, the F-SPACE manager must also be responsible for coordinating information flow among multiple users of a checked out frame, in case several I-SHELLa are actively manipulating it, and for performing global F-SPACE searches for users looking for fast access to a relevant frame by description rather than normal local browsing (i.e., global F-SPACE

browsing).

There are three ways a user can "check out" a frame from this library: by copy, by update, and by command. The user automatically accesses a frame by copy when he is browsing since browsing returns essentially no new information. Also, the user only needs a copy of a frame if he is only using it to accomplish certain tasks (e.g., editing) that do not generate any new information that should be stored back into the F-SPACE for later use. If, however, the user is generating new information to be stored back into the F-SPACE, then he will have to access the frame by update. In update mode, another user sees the updates, but is not allowed to manipulate the frame himself. If the user needs to run certain procedures that require exclusive use (e.g., procedures to control a satellite at 3oddard), he will have to access the frame by command. In this mode, another user can see the frame as the first manipulates it, but he cannot run the procedures that require access by command.

4. User Interface

4.1. The I-SHELL

The I-SHELL is the main per-user operating system in the I-SPACE, and is responsible for processing user commands and coordinating the system responses. The main functions in the I-SHELL are browseframe, focusframe, focusslot, and invokeslot which are run in response to user requests. When the user first signs onto the system or moves to another frame. two functions are called. The first is displayframe which sends the initial visual description of "the frame to the U-TERMINAL. This description includes a list of the slots and where the windows associated with the slots should be located. After displayframe is run, the second function, browseframe, Is called. For each slot in the frame, browserrame runs the browse procedure associated with"that slot. When the user focuses a frame, focusframe is called. It first checks which slot's" nave focus procedures. For all of those, it checks if there are already active procedures running in the slot. An active procedure is any recurrent procedure or any procedure that is only run once but has not yet completed its task. If focusframe finds no active procedures, it runs the focus procedure. If it finds an active browse procedure, it terminates that procedure and runs the focus procedure. If it finds and active focus procedure, it will allow the procedure to continue. If it finds an active invoke procedure, it waits and focuses the slot when the invoke procedure is done.

In an unfocused frame, the user may focus particular slots. When he focuses a slot, focusslot is called. Focusslot checks to be sure there is a focus procedure for that slot, and, if there is. stops any active browse or invoke procedure for the slot and runs the focus procedure. When the user invokes a 3lot, invokeslot is called which stops any active browse or locus procedure for the slot and runs the invoke procedure. When the invoke procedure terminates, the browse or focus procedure will be restarted if it was active at the time of invocation. In every case, these ?rocedures are-treated as if they were not there ?f the priority of the user is not high enough to allow him to run them.

Browseframe, focusframe, focusslot, and invokeslot call an internal functiontha t actually taxes tne action necessary to run the procedures. This function first gets any user-supplied arguments, and then evaluates the other arguments. If there is a channel for the procedure, the I-SHELL sends a request to the U-SCHEDULER to run the procedure. The I-SHELL is interrupted whenever the U-SCHEDULER has a result. The 1-SHELL takes this result and runs the consumers for the 3lot and activation level. If there is not a channel, the I-SHELL runs the procedure locally and then calls the consumers. If the procedure is to be run in the window, the I-SHELL passes the procedure to the U-TERMINAL to process it there.

The consumers for a given slot and activation level are run every time the procedure for the slot and activation level returns a result or finishes execution. When they are run, they know the frame, the slot, and the activation level information that caused them to be called. They also know what value, if any, was returned from executing the procedure. With this information, they are able to update the I-SPACE and tell the U-TERMINAL to display the new results.

4.2. The Display Screen

Given the power of the I-SHELL to coordinate the accessing of information from the diverse resources of the I-SPACE, it is crucial that this information be presented to the user in an efficient manner. The U-TERMINAL is an intelligent terminal handler that has similar characteristics o the display systems of [Lantz 1979] and Teitelman 1977]. It supports the modes of operaions of the I-SHELL by providing sophisticated displays, called viewports, through which the user examines a section of the'I-SPACE. Each viewport is a two-dimensional subsection of the total CRT screen area and corresponds to a specific active frame in the I-SPACE. For each slot in a frame, the corresponding viewport contains three windows (or subregions) that are associated with the three fields of a slot [i.e., Name, Value, and Status) Each of the windows of a viewport may be manipulated independently of the other windows on the CRT and supports a full spectrum of operations normally associated with a computer terminal (e.g., partial and total clearing, scrolling). Each window can be displayed with a variety of graphical renditions, which are normally employed to indicate the status of the associated slot (i.e., whether it is browsed, focused, or invoked).

Viewports may be manipulated in a manner akin to sheets of paper on a desk. They can be stacked on top of one another, moved to various locations on the screen, etc. This allows a user to place interesting information in different locations on the screen, much like a scientist spreads the several journals he is currently reading over the surface of his desk. Information addressed to viewports whose windows are covered by other viewports will be stored in individual window buffers so that when the overlapping windows are removed, the viewport that was partially obscured is refreshed without loss of information.

The user operation of the U-TERMINAL has been simplified and uses short (normally a single keystroke) commands to issue requests to the I-SHELL. Graphic enhancements of the display windows provide the user with instantaneous status information about previously issued requests. Information addressed to obscured viewports updates the covered windows without causing undue interference with the user's operations on another frame.

Finally a panic facility is provided that displays the currently available options to the user, as well as the capability to invoke a system HELP frame for more complex problems. The U-TERMINAL has been integrated into the I-SPACE system and specific terminal drivers for low resolution CRT and Tektronix terminals have been installed. While adequate for use during the development of the 1-SPACE these devices will be replaced by memory-mapped high resolution displays in the final development phase of the project.

I-SHELL to U-TERMINAL Interface

The communication protocols between the I-SHELL and U-TERMINAL parallels the interface between the I-SPACE and tne user. From the 1-SUELL perspective information corresponding to events occurring in the I-SPACE must be communicated to the user. These activities include operations such as browsing, focusing, or invoking a slot in a frame as well as returning the results of user initiated requests. Additionally, the I-SHELL must have a mechanism for obtaining values from the user that are needed by the functions used by the 3lots.

The user on the other hand wishes to communicate his requests for specific frame manipulation operations to the I-SHELL. This includes focusing the current frame or specific 3lots within the frame, invoking an interesting slot in a frame, or moving to u different point of interest within the I-SPACE. This could be either a previously activated frame, or an untouched area in the I-SPACE. Additionally the user wants local control over the way that the information is displayed on his terminal, including the ability to dynamically modify the arrangement of active frames.

The U-TERMINAL and the I-SHELL communicate using a message passing, paradigm (Implemented using the IPC facility of [Rashid 1980] in which requests to the I-SHELL are constructed by the U-TERMINAL as a response to a user command in terms of frame manipulation functions. Information directed from the I-SHELL to the U-TERMINAL is first napped to the corresponding viewports and windows and then sent to the CUT where, the information nay be displayed, determined by whether the window is covered or not.

5. Global Event-Driven Pattern Matching

Longer range I-SPACE research calls for the inclusion of a global, event-driven pattern matching system which will more fully automate the flow and dissemination of information throughout the I-SPACE. This system will have access to both the F-SPACE manager s activity queues (to watch the flow of frames to and from the F-SPACE), and selected 1-BANK channels. Users wishing to define personal or group-related event patterns will do so bv running special I-SPACE frames which are experts at thi3 activity. Event patterns will make reference to frame-checking-out events in the F-5PACE, to slot-filling events bv individual users, to general announcements from the 1-BANK (e.g., an I-BANK processor reporting that its remote host has just come up or gone down, or that a real-time satellite event of community-wide interest has just occurred), and to a real-time clock. This event-driven layer will enhance the character of the I-SPACE as expert assistant, by watching for its user in off moments, and by automatically initiating activities for its user.

6. Summary

An I-SPACE is a new form of intelligent operating system and human-machine interface wnich. we believe, holds promise for large 3cale, real-time information systems. Without the abilities to locate, access, manipulate, and synthesize information fragments efficiently and in user-idiosyncratic ways, managers and "technical staff in large organizations, and programmers in large teams, will become increasingly less productive as members of their large organization. An I-SPACE gives these abilities to a wide user population by allowing a user to bring to his screen a number of conduits to what would otherwise be unrelated information sources or services. The result for the user is a heightened ability to synthesize information relevant to decision making in complex tasks, or critical to large 3cale development projects. The I-SPACE concept is proving to *he* a workable one, and establishes a framework in which more advanced tools, including advanced event-driven pattern matchers and more intelligent database searchers, can be developed.

APPENDIX: A Brief Overview *of* ZKOB

ZMOB is a 256 processor multiprocessor research computer under current construction with funding from AFOSR. Architecturally it is 256 Z8OA microprocessors, each having G4K ,bytes of memory, and all connected as mail stops . ,around a 257 stage, 48 bit wide "conveyor belt" (shift register) which shifts at 10 mhz. Each processor is completely autonomous and is ,given the illusion of non-blocking, instantaneous" communication with another arbitrary processor or (subset of processors) by the high data switching capacity of the conveyor belt. At large, the machine will have an instruction throughput of 100 million instructions per second, a cumulative main memory of 16 million bytes, and an I/O throughput of 20 million bytes per second. Additionally, each processor individually has a high speed parallel and serial interface to the outside world, making possible remote data acquisition and information switching. Individual ZMOB processors will run one of three (assembly, PASCAL, LISP) operating systems, and will collectively communicate with the outside world via the 257th mail stop, which is a DMA interface to the VAX. The machine will be partially running by Summer 1981, and fully running by the end of 1981.

REFERENCES

[Allen 1981]
Allen, E.M., "The I-SPACE Frame Language" Univ of Maryland Technical Report in preparation, 1981.

[Bobrow 1977]
Bobrow, D.G., Winograd, T., "An Overview of KRL, A Knowledge Representation Language", Cognitive Science, Vol. 1(1), 1977.

[Goldstein 1980a]
Goldstein, I.P., Bobrow, D.G., "Descriptions for a Programming Environment", Proceedings of The First Annual National Conference on Artificial Intelligence, Stanford, CA, Aug. 1980.

[Goldstein 1980b]
Goldstein, I.P., Bobrow, D.G., "Extending Object Oriented Programming in Smalltalk", Conference Record of the 1980 LISP Conference, Stanford, CA, Aug. 1980.

[Ingalls 1976]
Ingalls,D.H.H., "The Smalltalk-76 Programming System Design and Implementation", Conference Record of the Fifth Annual ACM Symposium on Principles of Programming Languages, 1976.

[Lantz 1979]
Lantz, K.A., Rashid, R.F., "Virtual Terminal Management in a Multiple Process Environment", Proceedings of the Seventh Symposium on Operating Systems Principles Asilomar, CA, Dec. 1979.

[Newell 1977]
Newell, A., Notes for a Model of Human Performance in ZOC, Carnegie-Mellon University, Aug. 1977.

[Palay 1980]
Palay, A.J., Fox, M.S., "Browsing Through Data Bases" Proceedings of the Symposium on Research and Development in Information Retrieval, Cambridge, England, June 1980.

[Rashid 1980]
Rashid,R.F., "An Inter-Process Communication Facility for UNIX", CMU-CS-80-124, Carnegie-Mellon Univerait, Jun. 1980.

[Rieger 1981]
Rieger, C.,Trigg, R.,Bane, B., "ZMOB: A New Computing Engine for A.I.", Proceedings of IJCAI-81, Vancouver, BC, Aug. 1981.

[Teitelman 1977]
Teitelman, W., "A Display Oriented Programmer's Assistant", Proceedings of IJCAI-77, Cambridge, MA, Aug. 1977.