Using Common Graphics Paradigms
Implemented in a Java Applet
to
Represent Complex Scheduling Requirements

by John Jaap, Elizabeth Davis, & Patrick Meyer
Mission Planning Division
Mission Operations Laboratory
George C. Marshall Space Flight Center
National Aeronautics and Space Administration

Abstract

The experiments planned for the International Space Station promise to be complex, lengthy and diverse. The scarcity of space station resources will cause significant competition for resources between experiments. The scheduling job facing the Space Station mission planning-software requires a concise and comprehensive description of the experiments' requirements (to ensure a valid schedule) and a good description of the experiments' flexibilities (to effectively utilize available resources). In addition, the continuous operation of the station, the wide geographic dispersion of station users, and the budgetary pressure to reduce operations manpower make a low-cost solution mandatory. A graphical representation of the scheduling requirements for station payloads implemented via an Internet-based application promises to be an elegant solution that addresses all of these issues.

Scheduling Requirements

For activities on the International Space Station, scheduling requirements are described by defining "activities" and "sequences of activities." Activities generally equate to the simplest or lowest-level tasks. A sequence of activities is usually required to represent scheduling entities. Consider the following example from everyday life of an Internet user. To check e-mail from home, one must perform a sequence of three activities: connect to an Internet Service Provider (ISP), download new messages, and terminate the modem connection. The "scheduling entity" is the sequence of three tasks. Doing the activities out of sequence or standalone does not accomplish the objective.

Activities define the resource requirements (with alternatives) and other quantitative constraints of tasks to be performed. In the e-mail example, resource usage (e.g., phone line, ISP account and mail program) is requested by the activities. Of course, there may be alternative resources; one may have multiple phone lines, multiple ISPs and multiple mail-client programs. Even the computer itself is a resource that may be shared with other household members. The resource requirements for these activities are shown in the following table.

Connect	Download	Disconnect
Phone line	Phone line	Phone line
ISP account	ISP account	ISP account
Computer	Computer	Computer
	Mail Client	

While this representation of the activities seems straightforward, it may not be complete. The "Connect" activity would look more like the following if we show alternative resources.

Connect

$\text{or}\begin{cases}\text{Home-office phone}\\ \text{Residence phone}\end{cases}$

$\text{or}\begin{cases}\text{Employer's ISP}\\ \text{AOL}\end{cases}$

Computer

If there is some hierarchical relationship between the choices, we might show the activity the following manner.

Connect

$$\text{or}\begin{cases}\text{and}\begin{cases}\text{Home-office phone}\\ \text{Employer's ISP}\end{cases}\\ \text{and}\begin{cases}\text{Residence phone}\\ \text{AOL}\end{cases}\end{cases}$$

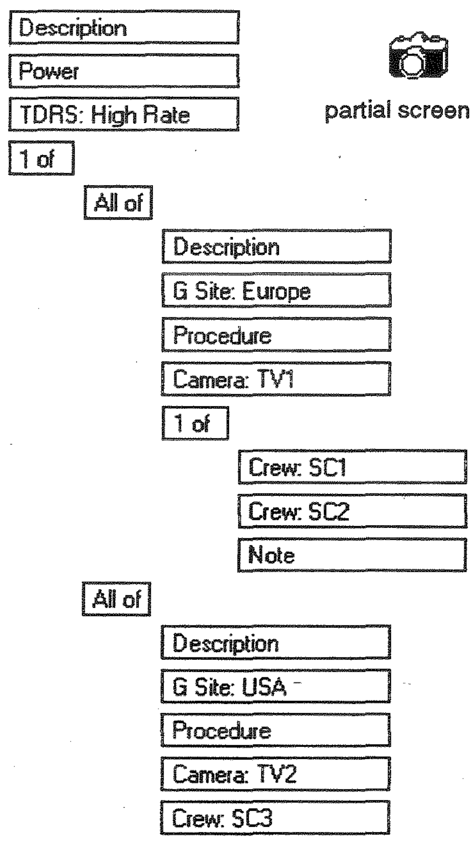
Computer
Electricity

Or, we might use the following "outline" paradigm.

```
Connect
    One of
        All of
            Home-office phone
            Employer's ISP
        All of
            Residence phone
            AOL
    Computer
    Electricity
```

The "outline" paradigm supports unlimited depth and can be intuitively manipulated by a drag-and-drop interface. The actual implementation of this method of representing activities uses dialog boxes that are activated by clicking on the constraints to enter constraint values.

The figure below shows an example of an activity that might be flown on the space station.

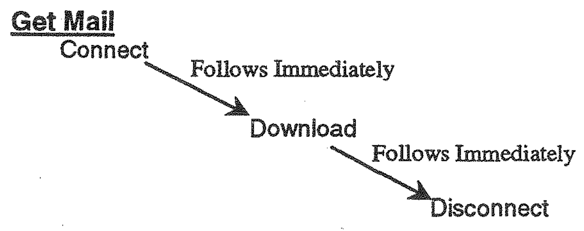


partial screen capture

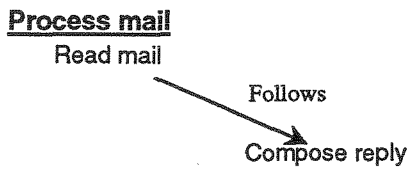
In this example, power and high-rate TDRS are always needed, and the activity must be performed over Europe or the USA. If done over Europe, a specified procedure is used to operate camera 1 by either crewman SC1 or SC2. If done over the USA, then a different procedure is used by crewman SC3 to operate camera 2. Of course the details of each constraint (for example, the amount of power) are entered into a dialog panel that appears when a box is clicked with the mouse. This example also shows that ancillary information such as procedures and descriptions can be associated with the constraint hierarchy.

For space station, the users are not allowed to create resource definitions; they are created by an administrator in negotiation with the station systems experts and with the user community. When defining an activity the user selects from a list of predetermined constraints. The dialog boxes for the values are tailored to the type of constraint. For space station, ten constraint types (and thirteen dialog panels) are currently defined.

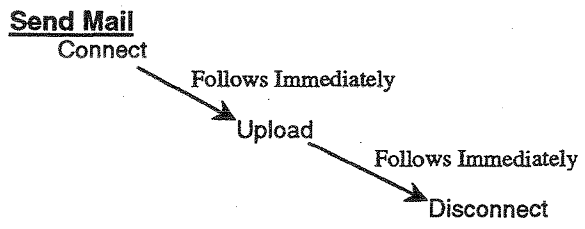
Sequences define the time relationships between activities. In our e-mail example we discussed three activities which would be done one after the other (i.e., in a sequence). This can be represented pictorially something like the following.



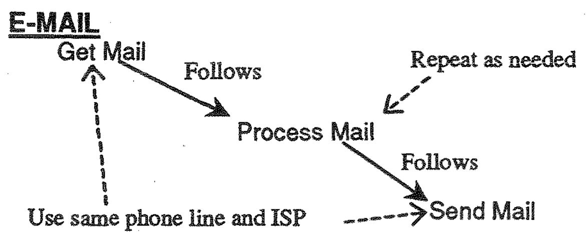
Consider two more sequences, one for processing mail (reading and composing replies to the mail) and another for uploading the replies to the ISP. To support these sequences three new activities (read mail, compose reply, and upload) are assumed but are not defined. The sequence for reading each message and composing a reply looks like the following.



The sequence for sending mail looks like the following.

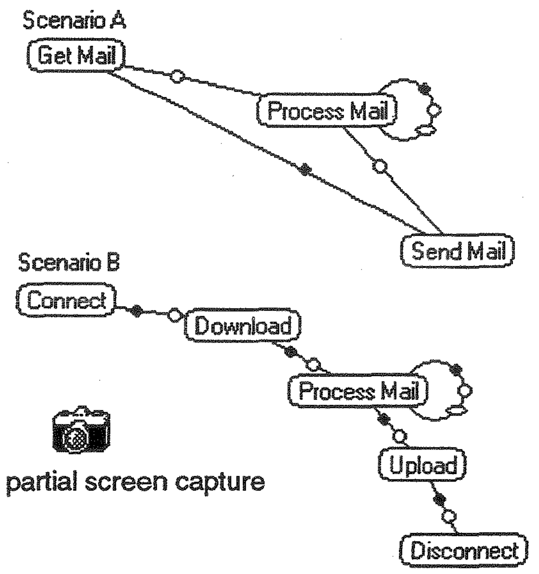


The whole process of getting mail, processing, and uploading mail would be expressed in a meta-sequence like the following.



In addition to the main sequence (Get Mail, Process Mail, Send Mail), instructions that multiple messages are to be processed and the same ISP and phone line are to be used for getting and sending mail are shown.

Of course, there is another scenario for doing one's e-mail; one could connect, download the mail, process it (read and reply), upload and then disconnect. The implementation allows the user to define multiple scenarios of a sequence on the same drawing panel. Below is the actual representation of the e-mail sequence. Notice that both scenarios are shown and that sequences can be nested (Process Mail is a sequence) and mixed with activities.



The program shows the relationships between the items by showing a placeholder (circle for temporal relationship, oval for repetition relationship, or spot for resource persistence). The details of each relationship are entered via dialog panels that appear when a relationship placeholder is clicked with the mouse. Often, the relationship is more complex than simply "follows" of the e-mail example. A sequence may also show temporal relationships to station events which are not part of the experiment. For example, a payload activity or sequence might have a temporal relationship to a station event like reboost or shuttle docking.

Implementation

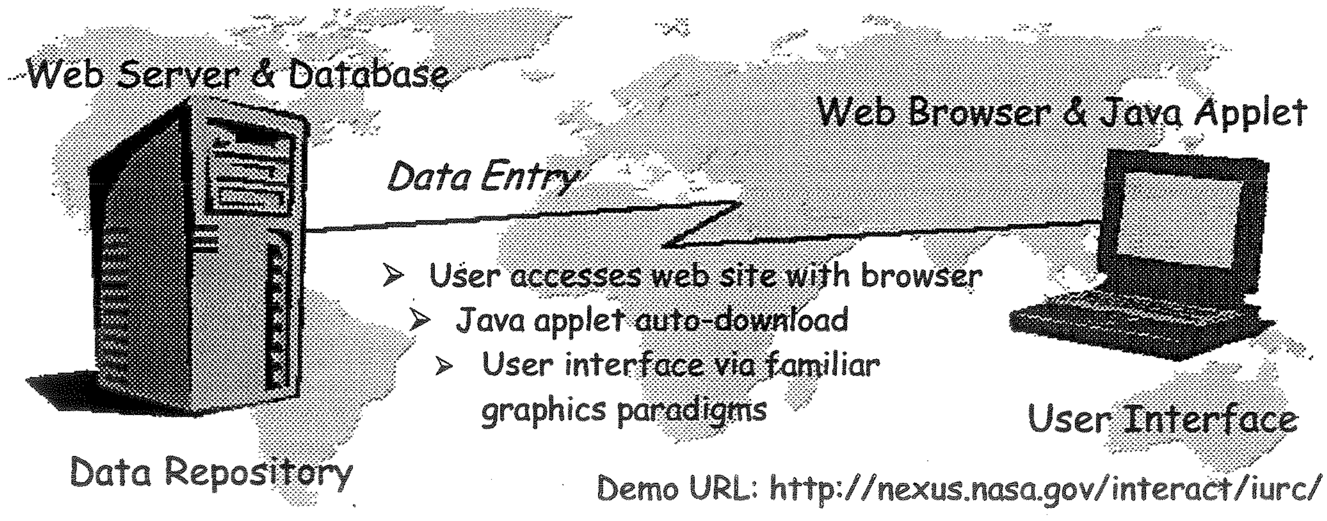
In support of space station, the Mission Planning Division at the Marshall Space Flight Center is implementing a Java-based remote data entry system to collect user requirements for payload planning and scheduling.

The data collection system has several components as shown in the figure below. The web server and the database are located at Marshall. A user connected to the Internet supplies a web browser. When the user points his browser to the web site, a Java applet is downloaded and executed (after user authentication). The applet provides the graphics and dialog interfaces which allow the user to enter/edit his scheduling requirements, and it uploads and downloads the requirements from the database on the web server.

Summary

In the e-mail example, we saw that simple graphics paradigms can encapsulate a wealth of information about hierarchies and relationships. However, using graphics paradigms to represent scheduling requirements has even greater advantages. Graphics paradigms are less susceptible to misinterpretation and can overcome language barriers; they can represent requirements concisely while capturing scheduling flexibility; and, with the recent advances in technology, they can be effectively implemented.

A Java applet, to run in a web browser, has been developed to support the graphical representation of payload scheduling requirements. Implementing the entry and editing of requirements via the web solves the problems introduced by the geographic dispersion of users. Reducing manpower is accomplished by developing a concise representation which eliminates the misunderstanding possible with verbose representations and which captures the complete requirements and flexibility of the experiments.


Web Server & Database
Web Browser & Java Applet
Data Entry
➢ User accesses web site with browser
➢ Java applet auto-download
➢ User interface via familiar graphics paradigms
User Interface
Data Repository
Demo URL: http://nexus.nasa.gov/interact/iurc/