The Space Station Information System
and Software Support Environment

Clarence W. Pittman
Director, Software Technology Division
NASA Space Station Program Office

The Space Station will be a
large, permanent, multi-
purpose facility with
comfortable living quarters
for a crew of up to eight
people. It will consist of
four habitable modules and a
truss to which payloads which do
not require human involvement
can be attached. It is intended
to accommodate a wide variety of
science and technology projects
(astronomical and geophysical
observations, materials and life
sciences, etc.) and, as it
evolves, to serve as an assembly,
servicing, and departure point
for other space missions. The
truss, utility services, and two
of the modules will be provided
by NASA, one module will be
provided by the National Space
Development Agency of Japan, and
the other by the European Space
Agency. The entire manned base
will be carried into orbit and
assembled in nineteen increments
over a period of two years around
1995.

A large part of the value of the
Space Station will reside in the
information exchanged between it
and the related ground
facilities. To insure that
this exchange is effective,
the Space Station
Information System (SSIS)
has been identified as an
explicit element of the program
and a management structure has
been put in place to oversee the
development of the end-to-end
system, including those elements
not provided by NASA. This will
involve identifying all the
components of the SSIS and the
people responsible for each,
comparing their schedules,
establishing interface and
protocol agreements, and,
providing for testing the overall
system. The SSIS management
intention is that the non-NASA
participants, including the
international partners, will have
complete freedom in the design of
their hardware and software,
providing that they satisfy the
agreed upon interfaces.

The SSIS is an extensive
collection of hardware
(computers, networks, facilities)
and software, whose primary
purpose is to carry data between
a space-based source and a
ground-based user. The data
source could be a scientific
instrument on the manned base or
on a platform; it could be a
piece of on-board equipment; or
even a crew member. The
ground-based user could be an
experimenter operating from his
home institution or an operator
based in a spacecraft control
facility. The data itself might
be scientific data; housekeeping
data used to monitor equipment
health and safety; a database
query; or audio and video data
forming part of a space-ground
teleconference. While the
situation is not completely
symmetrical, most of these
data-flows will also occur,
simultaneously, from ground to
space.

The collection of hardware
elements involved in this process
is large and varied, consisting of
more than ninety space and
ground elements which combine to
form the SSIS. Some of these
elements are new and unique to
the SSP while others (such as the
Tracking Data Relay Satellite
System (TDRSS)) are extended
versions of existing NASA
institutional capabilities. Some
elements are outside of SSP (and
even NASA) control, such as
customer facilities and public
communications networks.
However, being connected to the
SSIS and using its services, they
can be considered as SSIS
elements.
In the past, space mission data systems tended to be rather inflexible, schedule-driven systems which were labor-intensive to operate. Typically, users' data would be delivered several days (or longer) after reception, frequently in formats unique to NASA. The SSIS concept involves some significant improvements:

1. The trend is away from schedule-driven and towards data-driven systems - within certain prearranged limits, the system will autonomously accommodate changes in data rates and data destinations. This results in operations which are more flexible, less labor-intensive, and give better service to users.

2. A necessary building block in the construction of data-driven systems is the autonomous data unit - instead of appearing to the data system as anonymous bit streams, basic data units will be packetized and self-identifying. This allows many of the data system functions, such as routing data to users and removing data transport artifacts, to be entirely automated and generic in nature.

3. Packetized data can be further packed into 'virtual channels' so that many logically distinct data streams can be multiplexed into the same physical medium (for example, the space-ground link via TDRS).

4. In addition to the program benefits of these techniques, the adoption of packetized, data-driven methods has equal benefit to the user community in terms of increased operational flexibility and reduced software development effort.

Another way in which the SSIS differs from previous data systems is in the concepts of tele-operations and tele-analysis. The tele-operations concept is often described as the ability for a customer to interact with his payload as if that payload were in a laboratory 'next door'. The realization of this concept imposes some stringent requirements on the intervening data systems:

1. They must provide constant, standard interfaces so that the payload/user interface looks the same regardless of whether the payload is on a testbench in the manufacturer's facility, in an integration facility at the launch site, or in orbit attached to the station or a platform.

2. They must provide data transparency - transporting data with minimum interference, regardless of the data content.

3. They must transport data with minimum delay so that the investigator can interactively change the course of an observation based on current information.

4. They must provide all of the above regardless of the customer's physical location: at his home institution, at a NASA facility, or at an international location.

Tele-analysis extends the concept to the ground processing of payload (and other) data. It includes the ability to locate useful data in distributed databases and archives, to extract and receive sub-sets regardless of the physical location of the data, and finally, to combine and reprocess datasets (possibly from different investigations) to produce new datasets of increased utility.

Similarly, the software for the SSIS will be large and complex, developed by many groups of people at many locations, and expected to operate safely and undergo extensive evolution over the life of the Space Station. The Software Support Environment (SSE) is being developed to control the development and maintenance of the SSIS software. Its purpose is to minimize the cost of the software over its life cycle while ensuring a high degree of safety and reliability. It embodies the "tools and rules"
i.e., the computer programs, procedures, and standards for the management, development, configuration control, integration, test and verification, and maintenance of the SSIS software. Virtually all software developed for the Space Station will use the SSE.

The SSE will reside in numerous facilities: the Software Support Environment Development Facility (SSEDF); a number of Software Production Facilities (SPF's); and the Multi System Integration Facility (MSIF). As its name implies, the SSEDF, located in Houston, is where the SSE will be developed. The SPF's will be at the locations where SSIS software is being produced. A tailored subset of the total SSE will be assembled at the SSEDF and installed at each SPF. The MSIF is the location at which the software elements produced and tested at the SPF's will be integrated and tested. It will contain sufficient simulations/emulations and/or prototypes of the Space Station Systems to allow the software to be tested in a realistic environment.

The SSE will control all phases of the software process, from development project initiation to operation and maintenance. When a software development project is initiated, the Project Manager will interact with the SSE to identify the project, the organization which will execute it, and the lines of authority within the organization. The SSE will subsequently report status and problems appropriately. He will also identify the tasks within the project and the personnel assigned to the tasks and the SSE will then control the access and activities of the personnel, i.e., testers cannot alter code, designers cannot perform formal verification testing, etc. The SSE will not permit the project to proceed until these task/personnel relationships are defined. Also during project initiation, the SSE will assist the project manager to tailor the software life cycle model to fit the particular project and provide him with templates of the required documentation.

The SSE provides an extensive set of tools for the design, development, and integration of the software. For design and development, these include tools for requirements and traceability analyses, preliminary design, detailed design, coding and documentation standards compliance, code analysis, interactive de-bugging, etc. For test and verification, the SSE will have tools to evaluate requirements and design adherence, verify test completeness and comprehensiveness, configure the system for the test, perform the test and record the results, and identify and record deficiencies.

Throughout the process, the SSE will use a DBMS to retrieve all the data objects produced by the project and it will support configuration management by retaining the currently approved versions of all controlled products.

The SSE software can be thought of as a collection of functional elements distributed in an architectural framework. The functional elements are: process management; software management support; software production; modeling, simulation, and software checkout; integration, test, and verification; data reconfiguration; and training. These functional elements are distributed among four architectural elements, which are called the Framework; Tools; Host System Software; and SSE Management. The Framework provides the functionality and control of the SSE. Users access the life-cycle products, tools, tests, and data through the Framework. The Tools element provides what its name implies. The Host System Software contains functions typical of host operating systems and utilities. These include executive, data base management, host configuration control, and communications. The SSE management element provides non-development management functions such as incorporation of non-SSE inputs, system performance monitoring, system management and resource analyses.
In addition, the SSE software includes a Human/Computer Interface; a Test and Tools Harness, which provides the interface between the Framework and the Tools; and a Distributed Ada Interface Set, a standard layer of interface services that bridges between the logical and physical views of the system.

Physically, each SPF will consist of one or more host computers, a collection of work stations, a local area network, and access to wide area networks. It will be designed to be hardware-independent and to accommodate technology upgrades transparently. Initially, the host computers will be from the VAX 8000 series and the IBM 3090 series, and the work stations will be Apple Macintosh II, IBM PC/AT compatible, and Apollo Series 3000. The local network will be an Ethernet-based TCP/IP LAN. In addition, there will be a specially designed Simulation Interface Buffer to provide an interface between the SPF and local simulations/emulations/prototypes of other Space Station systems.