A PACKET-BASED CONCEPT FOR SPACECRAFT COMMAND PLANNING

Valerie B. Barnes

The Johns Hopkins University Applied Physics Laboratory
Laurel, Maryland/USA

ABSTRACT

The current generation of spacecraft being developed and operated by the Applied Physics Laboratory provides users with access to a broad spectrum of scientific instruments on maneuverable platforms that can be oriented for observation of both moving and stationary targets of interest. The capability of these increasingly complex spacecraft to perform data collection operations is approaching one observation per orbit. To enable both rapid configuration and generation of complex spacecraft command sequences, as well as reusability of command sequences among data collection operations, a packet-based concept for spacecraft command planning has been developed.

The configuration of the spacecraft for any operation is designed using "packets" where a packet represents a set of commands that is reusable. The packets can be combined in varying levels of functionality, and in varying time relationships, to create an observation timeline. At the lowest packet level are primitives. Primitives relate the details of command generation for a particular spacecraft to a "message template." Thus the packet concept itself is reusable from one spacecraft to the next.

Key Words: Operations planning, command, aerospace

1. INTRODUCTION

Complex satellites are built to support many different experiments. Because the same instruments and spacecraft subsystems are being used for those experiments, there is some degree of command commonality across different experiments. Often, a single experiment cannot be completed by one data collection event; rather, the experiment is designed as a series of several similar data collection events. The operations planning process developed at APL is designed to exploit command commonality across experiments and within each experiment class.

Spacecraft events are carried out by executing a series of commands that tell the spacecraft how to maneuver, what devices to use, how to configure them, and when to take each action. To streamline the event planning process, sets of spacecraft configuration information are defined. This configuration information is divided into two types: reusable building blocks called **planning packets** and event-peculiar structures called **event specifications**.

Figure 1 shows possible planning packets. The planning packets are used to simplify the planner's job. For instance, a specialist may make a low-level packet to partially or completely define a particular instrument configuration for several experiments. A mid-level packet may be set up to accomplish one type of maneuver, leaving details such as pointing angles and timing blank. Or, a packet may be constructed to define a spacecraft-wide function such as recording data in one of several formats.

A planner builds a top-level packet from existing lower-level planning packets to meet the sequence and configuration requirements for an experiment type rather than starting from scratch each time. When a data collection event for that experiment is scheduled, the planner sets up an event specification which points to the top-level packet.

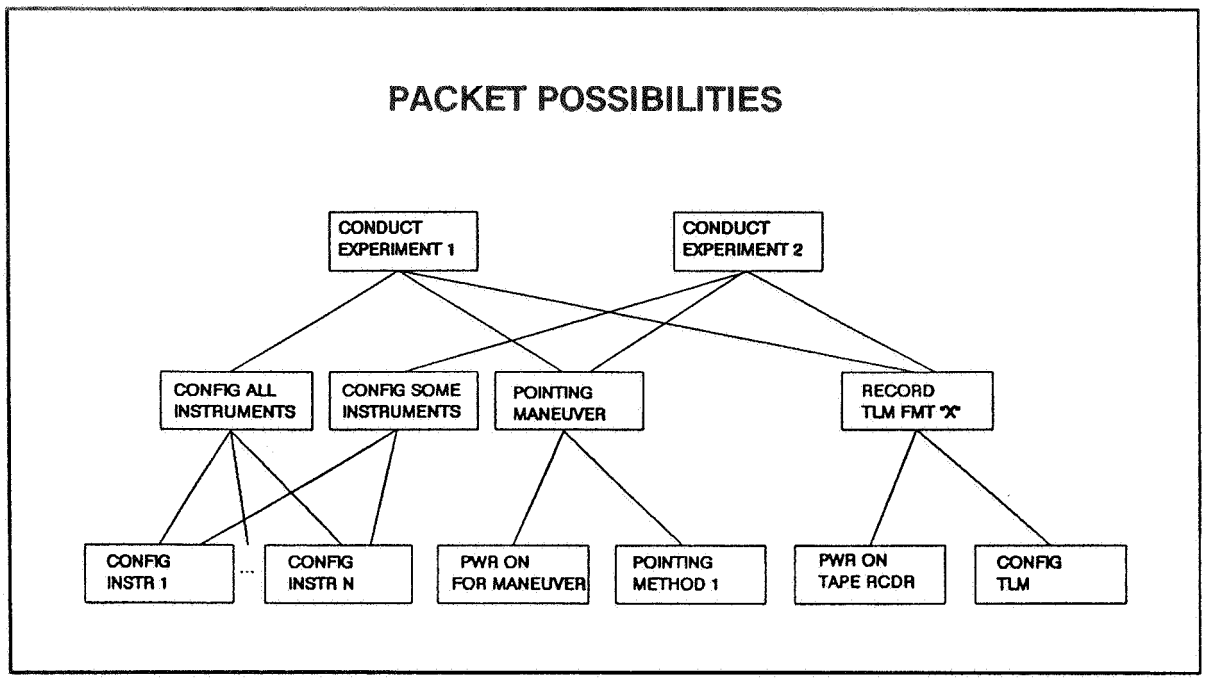

Figure 1

All the data and timing from the pre-built packet are used to initialize the event specification. The packet data are tailored for that particular data collection event by filling in each blank data value, adjusting timing, and storing the results in the event specification. When completed, the event specification contains all of the data necessary to formulate all the spacecraft commands required for the event.

2. RAW MATERIALS FOR PACKETS

APL satellites are operated via "messages," each consisting of one or more commands. In the operations planning system, **message templates** are used to define the characteristics of each message type. The information stored for each message template is used in building the reusable packets; in soliciting, validating, and interpreting event specification data; and in formulating messages in the spacecraft's command language (APLCOM). The message template library contains the following for each message template:

- Message Template ID
- Message Template description
- Command type (serial digital, pulse, relay, memory load)
- Number of data fields
- APLCOM format
- For each data field
  - name
  - description
  - type & length
  - category (list, constant, range, calculated)
  - units
  - default value (optional)
  - list of acceptable values or value range and accuracy
  - power delta corresponding to each "mode" change
  - locked command flag
  - telemetry response.

Thus, message templates form the foundation for the packet-based operations planning system.

Each message template only designates the valid alternatives for its data fields. To give a message template meaning for a particular function, at least one data field must be filled in. The next layer in the operations planning hierarchy is a **primitive.**

A primitive is a reusable instance of a message template in which at least one data field is filled in. A primitive can be defined to configure a specific device (e.g. channel 33 relay ON) or to configure a generic device (e.g. tape recorder "X", record at 25 Mbps). The user identifies the data fields in the primitive as constant or variable and sets fixed or default values in data fields. Making every data field in a primitive a constant means that no additional information will be entered during the event specification process.

Primitives are stored in a library for use in planning packets.

3. PLANNING PACKETS

There are three types of planning packets in the packet hierarchy: **event packets, action packets, and command packets**. Figure 2 depicts the packet hierarchy.

The lowest-level packets are the **command packets**. Command packets define reusable sequences of primitives in which timing is fixed for a particular component, sensor, or group of spacecraft components. These packets are used to perform some basic function such as a power-on sequence.

At the next level, **action packets** are used to define configurations that accomplish some broader function common to more than one event. Action packets are constructed from command packets. An action packet might be used to define a type of maneuver or a series of filter setting changes for data collection with an imager.

Event packets occupy the highest level of the packet hierarchy. Event packets are collections of action packets which define configuration sequences for every spacecraft component needed to perform an event.

Packets contain this information:

- Packet ID
- Packet name
- Packet description
- Spacecraft subsystem, instrument, subsystem component, instrument component or sensor
- Number of packet elements
- For each element of the packet
  - element ID (primitive or lower level packet)
  - element sequence number.

When a packet is defined, the user may set defaults (data values and time offsets). The defaults are for:

- primitive data field values
- device selections
- definition of a scheduling point for the packet
- relative time offsets from that scheduling point for everything else in the packet.

Some defaults may be defined to be constants; others may vary. A variable data field value may be changed at any higher packet level or in the event specification. Invariability flows up the packet hierarchy (primitive to command packet to action packet to event packet). The packet defaults are used as the starting points for an event specification.

Every packet must have exactly one scheduling point (time offset 0). The scheduling point is used to schedule a command packet in an action packet or an action packet in an event packet. If necessary, the user may use a dummy placeholder for the scheduling point. The placeholder will not be translated into a spacecraft command; it will only be used to establish relative timing within the packet.

Timing relative to the scheduling point is set for each element in the packet. If the element is itself a packet, the time offset applies to all internal sub-elements. Relative time offsets must be defined as constants in every command packet. Relative timing defaults may be assigned for higher-level packets; if they are set, they are variable. Packets are stored in libraries for reuse.

4. EVENT SPECIFICATIONS

The **event specification** is created when an

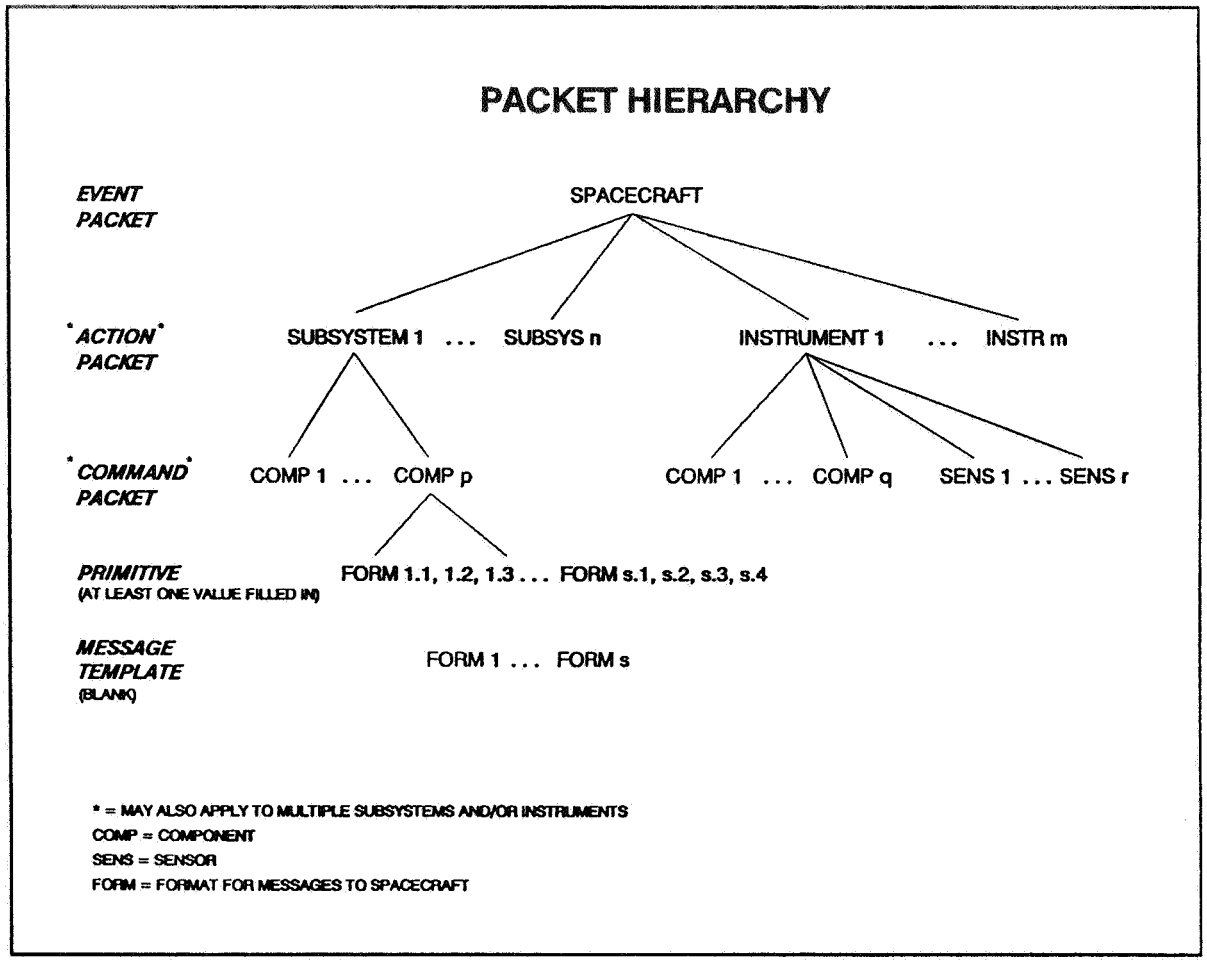

Figure 2

event packet is linked to a particular data collection, maintenance, or a downlink event. The constant, default, and timing values associated with the planning building blocks comprising the event packet are then copied into the event specification. An event packet must exist before the event specification can be created. Figure 3 illustrates the relationships among packets and shows the kinds of data associated with an event specification.

The event specification data may be divided into these categories:

- Event-level data
  - planned T-0 (time used in analysis of the event)
  - scheduled T-0 (time at which the event has actually been scheduled)
  - description
  - relationships to other events
- Relative time offsets
- Primitive data values.

The event-level data in the event specification are unique to a particular event. The remaining data are initialized based on the values set in the default section of the packet. The defaults were first set as constants or variables during the packet-building process. The variable settings may be edited. The user is prohibited from changing any packet data that were set as constants. Relative time offset data tied to packets and the event's T-0 are used by analysis and command building software to determine the sequence and absolute execution time of primitives over the entire event.

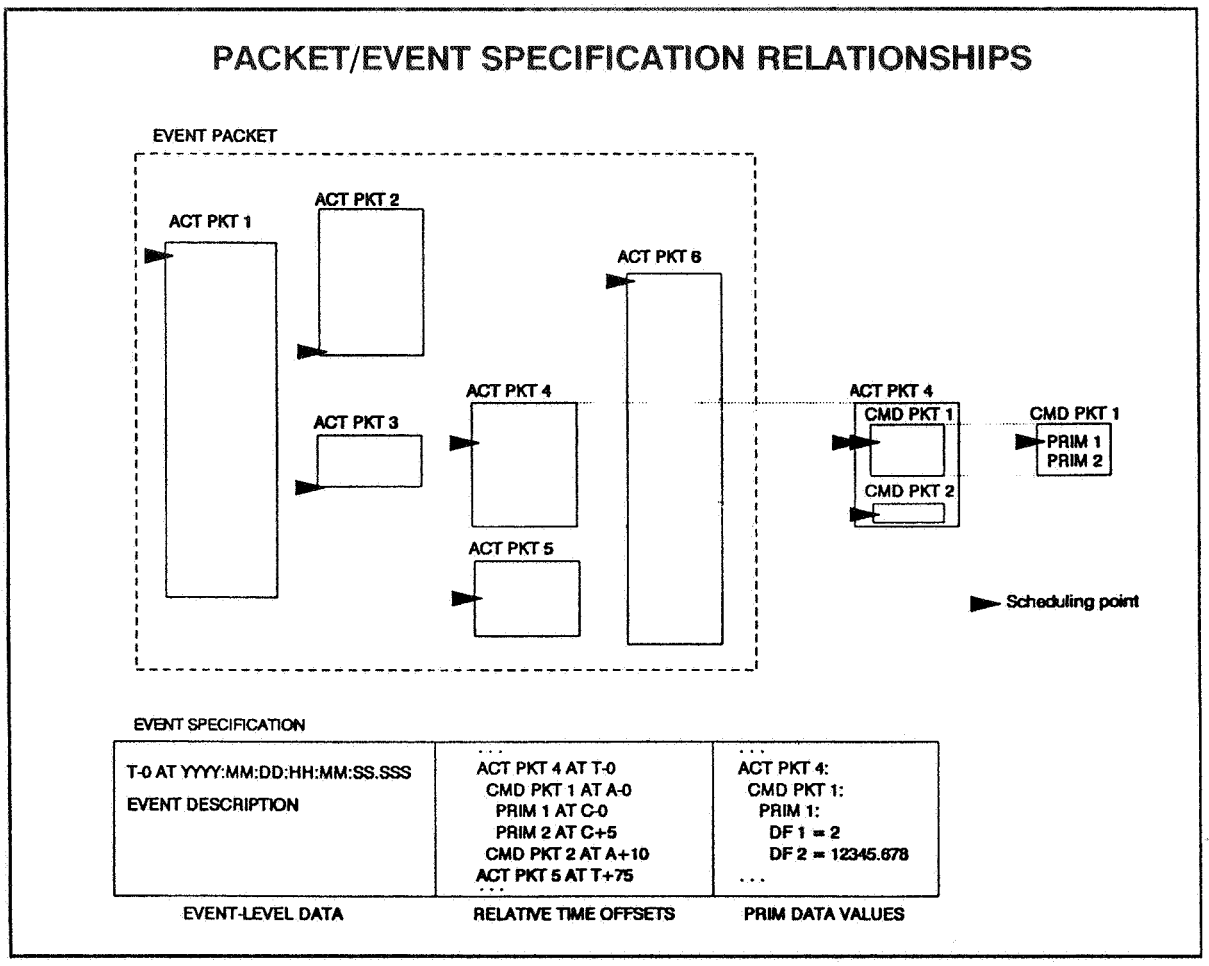

Figure 3

5. IMPLEMENTATION

The event specification is used to drive event analysis software. This software evaluates the predicted geometry, attitude, power loads, and thermal conditions. Event analysis software also uses the event specification to assess compliance of the event's sequence of activities with flight rules, and verifies that all elements of the specification are consistent.

The command formulation software builds commands based on the data in the event specification and on the APLCOM format information stored for each message template.

All packet and event specification data are stored in a relational database. Point and click windows function as user interfaces for building primitives and packets, editing event specifications, executing analysis software, and formulating commands. Software is designed to check a user's input against data value domain definitions stored in the database. The system is implemented on a network of RISC workstations operating under UNIX.

6. CONCEPT PROS/CONS

6.1 Pros

The packet concept enables operations planners to catalog knowledge about how to use the spacecraft and to reuse that information for a similar purpose. Packets can be built in advance and checked once. Packets can be set up to mirror the physical structure of the spacecraft or to match the logical nature of a function spread across

several subsystems.

Software is used to verify the activities in an event sequence prior to building commands. Software is also used to build commands. By removing humans from the command formulation step, translation and transcription errors are eliminated.

The packet-based planning concept and system design are transferrable to other spacecraft. For a different satellite, a new database of message templates and libraries for primitives and packets would be needed.

6.2 Cons

Using packets to plan is an effective solution if the spacecraft performs similar functions repeatedly. The packet-based system is cumbersome when building small command sets for one-time use.

To build packets that are reusable, the builder must have a broad understanding about how different experimenters plan to use the spacecraft. APL addresses this issue by assigning responsibility for the smallest building blocks to spacecraft specialists who are very familiar with the spacecraft. Operations coordinators who understand all experiment plans within one or more experiment classes are responsible for higher-level planning packets. Planning team members communicate often so that multiple packets are not built for the same purpose. One team member is responsible for maintaining control of the packet libraries.

6.3 Summary

The packet-based concept for command planning provides flexibility and reusability to the planning process. It allows the analysis and command-building processes to operate from a common set of information. Using this concept, planners think in terms of how to command the spacecraft in every planning phase. The on-orbit planning team size can be minimized through the use of pre-built planning packets.

ACKNOWLEDGMENT: This concept was developed to support the operations planning function for the Midcourse Space Experiment under Navy contract N00039-91-0001. Turning the packet concept into reality would have been impossible without the brainpower and dedication of my colleagues. Special thanks go to our database guru, Ron Glaser, whose clarity of thought, insistence on consistency, and sense of humor were vital.