An Application Engineering Workbench for Tailoring Ada Flight Components

Ross H. Wainwright
Air Force Research Laboratory
3550 Aberdeen Ave SE
Kirtland AFB, NM 87117-5776
1-505-853-4123

ross.wainwright@vs.afrl.af.mil

1. ABSTRACT

The Application Engineering Workbench is an interactive tool that automates the production of satellite flight software. The Workbench uses a hierarchy of graphical user interfaces (GUIs) to guide the satellite engineer through the process of entering the requirements of a flight software system. These requirements are captured in a relational database and are used to generate tailored Ada components. The decision model is the underlying mechanism that groups and orders the user's decisions. It enables the human to specify and the software to use mission requirements. The Workbench was developed by the recently completed Reusable Software Architecture for Spacecraft (RSAS) program.

1.1 Keywords

Satellite Flight Software, Software Reuse, Decision Model, Ada, Rapid Prototyping

2. INTRODUCTION

The development and support of software can be a complex activity that is subject to the wide variation in human behavior and individual assumptions. A technique for increasing software quality focuses on the concept of reusable software components and the process of using these components to build systems. An example of this technique is the Common Ada Missile packages program that successfully demonstrated the use of reusable Ada software for mission critical real-time embedded systems[1].

The RSAS Workbench is interactive software that allows a system engineer to enter requirements of a flight software system, select reusable components, and generate tailored versions of those components. Reusable components are not limited to source code, but also include requirements, design documents, test descriptions, test cases, associated test software, and past usage metrics.

The RSAS program has taken advantage of earlier work performed by Lockheed Martin in which domain analysis of satellite flight software was completed. The products of the domain analysis are complex. A process to navigate and tailor the domain model was essential to the viability of the Workbench. This process is called the decision model.

This paper begins with a description of the Workbench and an example. The example includes the views of the graphical user interfaces that guide the user through the decision process. The example is followed by a discussion of the underlying Decision Model and its implementation using a relational database.

3. WORKBENCH[4]

The Workbench is based on the Software Productivity Consortium's concept of leveraged Synthesis[2]. Synthesis constructs software systems as instances of a family of systems that have similar descriptions. Domain engineering and application engineering are the two aspects of the Synthesis process. The domain engineer's products are the domain model and the process to support the application engineer. The process allows the application engineer to extract and modify components from the domain to meet the requirements of a new application. The process can be used repeatedly to facilitate prototyping, and even makes possible a product-line approach to software development.

The Workbench, as diagrammed in Figure 1, has two main areas called Domain and Application. The domain area is more static, and contains the software components and tools which are used repeatedly to produce application products. The application area of the Workbench is much more dynamic. This is where the mission-specific data is loaded into value tables and eventually application components are generated.

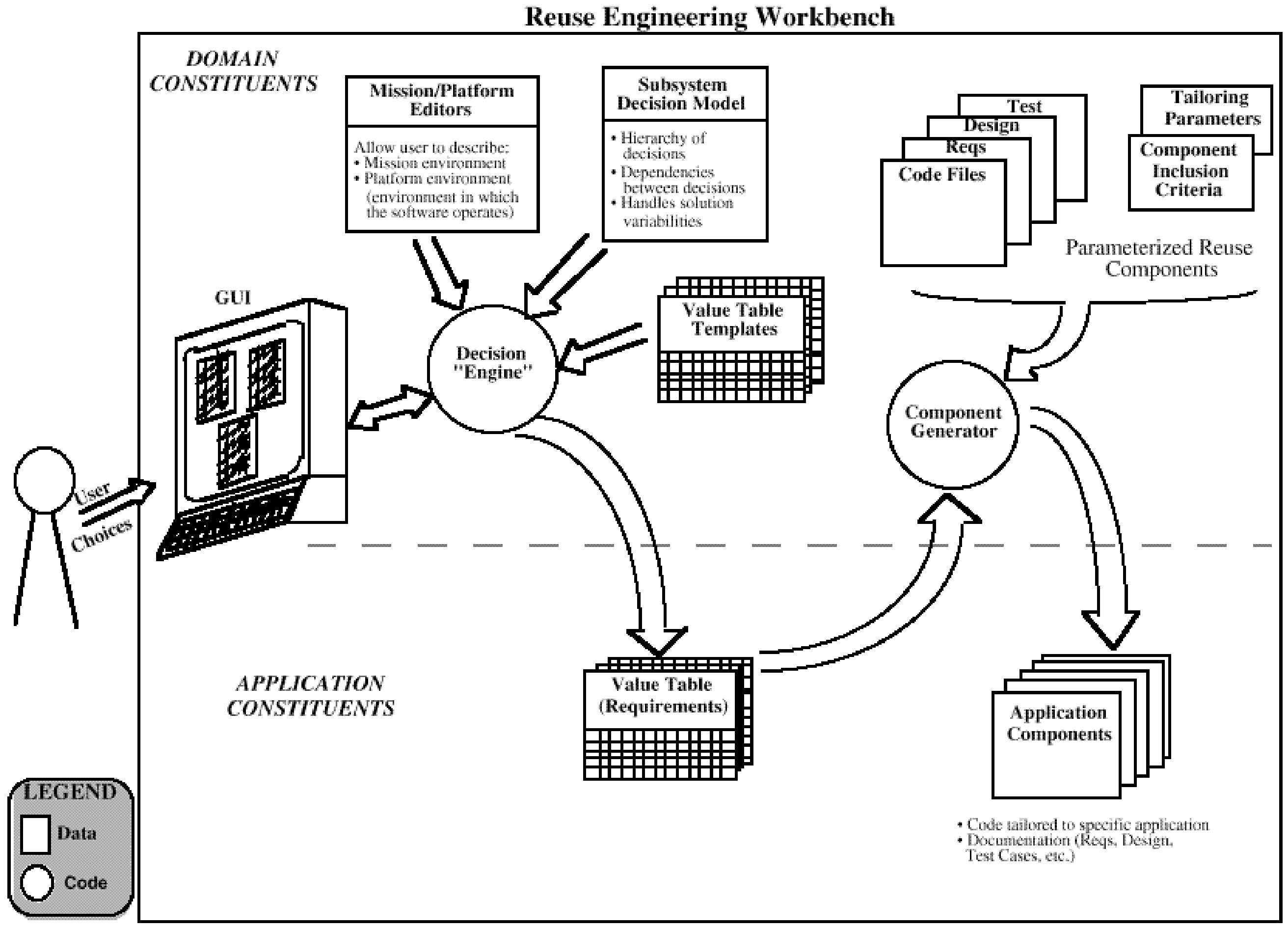

Reuse Engineering Workbench
DOMAIN CONSTITUENTS
Mission/Platform Editors
Allow user to describe:
• Mission environment
• Platform environment (environment in which the software operates)
Subsystem Decision Model
• Hierarchy of decisions
• Dependencies between decisions
• Handles solution variabilities
Test
Design
Reqs
Code Files
Tailoring Parameters
Component Inclusion Criteria
Parameterized Reuse Components
GUI
Decision "Engine"
Value Table Templates
Component Generator
User Choices
APPLICATION CONSTITUENTS
Value Table (Requirements)
Application Components
• Code tailored to specific application
• Documentation (Reqs, Design, Test Cases, etc.)
LEGEND
Data
Code

Figure 1[4]

The Workbench consists of many subsystems whose overall purpose is to aid a system engineer in the specification and development of flight software adapted to stated mission requirements.

Mission/Platform Graphical Editors. The graphical editors are typically used soon after the user has created a new mission. They allow the user to communicate to the Workbench, via graphical representations, the mission and platform requirements that have flowed down from work done by mission analysts and spacecraft systems engineers. There are two graphical editors. The Mission Editor allows the user to specify mission phases, constellation spacecraft, celestial bodies, ground stations, relays, etc. In the Platform Editor, the user lays out the environment within which the software must operate: sensors, actuators, antennae, computers, busses, etc.

Subsystem Decision Model. This is a database of interrelated decisions that the user must make in order to specify a software subsystem for a particular computer. Embedded in the decisions are the dependencies between them. For example, if the user has indicated the existence of a Global Positioning Satellite (GPS) receiver, then later in the decision process s/he is asked to specify the details of the software, which handles the GPS receiver.

Value Table Templates and Requirements. The templates expedite runtime processing, and are instantiated as requirements when a user creates a Workbench mission. They contain a consistent set of default values for decision model decisions. That is, together they already represent one of the more common RSAS solutions, and as the user makes decisions about the software, this consistency is maintained through the use of rules.

Decision Engine. This software drives the traversal of the decision model, presenting the decision screens to the user and populating the mission-specific value tables.

Component Generator. This software generates and tailors the components according to values produced during the decision-making process.

3.1 Workbench Example

Figure 2 illustrates the process flow.

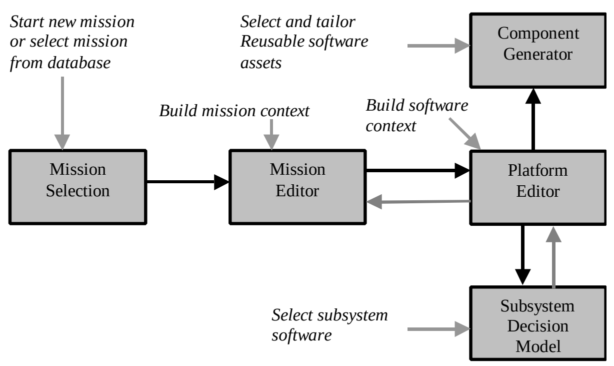
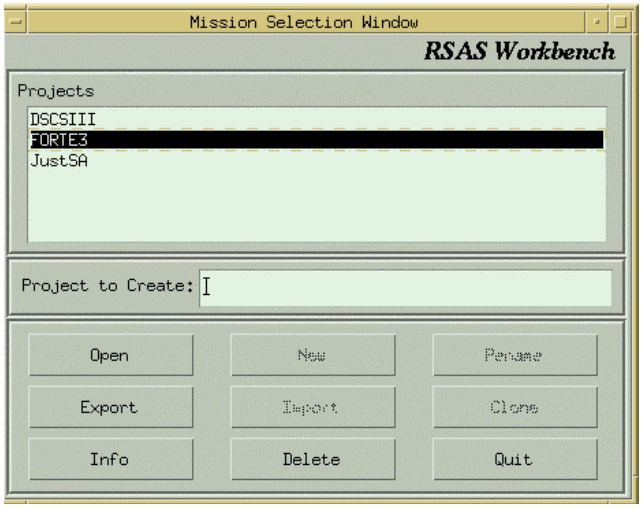


Figure 2[6]

A typical operations scenario for the Workbench is now described.

The user begins by creating a new project or selecting a project from the list from the Workbench's startup window. Here the mission, FORTE3, is selected in the Mission Selection Window (Figure 3). (FORTE is a Los Alamos National Laboratory and Sandia National Laboratory experimental satellite[3]. This work resulted from another AFRL/VSSS project. The Workbench was used to gain insight into the FORTE attitude control system. Software developed by the Workbench was **not** flown on FORTE.)



Projects
DSCSIII
FORTE3
JustSA

Project to Create:

Figure 3

The Mission Editor (Figure 4) is opened for FORTE3. This editor allows the user selects various items from the tool palette to graphically describe the mission environment.

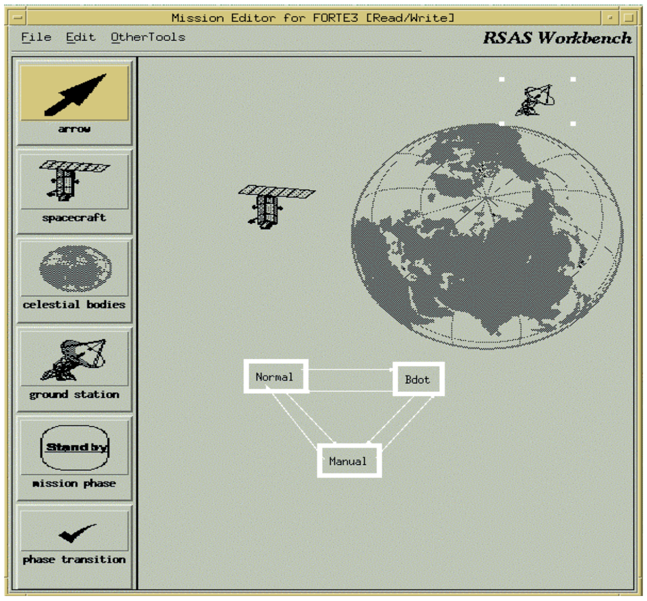
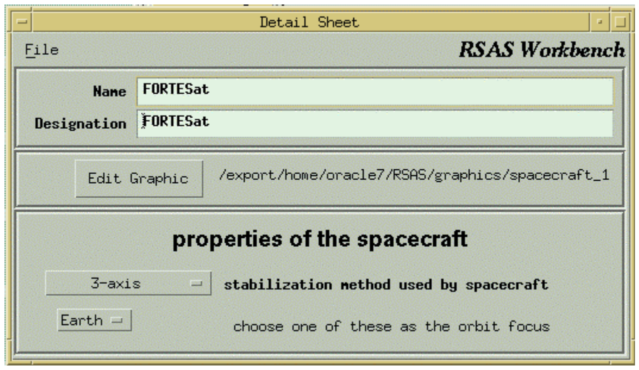


Figure 4

The mission environment is the external environment in which a spacecraft flies and includes influences on the host's orbit or attitude, entities with which it communicate, entities for which it maintains orbit knowledge, and perhaps entities it remotely senses. Some possible entities are mission phases, constellation spacecraft, celestial bodies, ground stations, and relay satellites.

The user may ask for a Detail Sheet for any item in the display. The Properties of the Spacecraft Detail Sheet (Figure 5) allows the user to enter relevant details, such as spacecraft names and orbital characteristics.



Name FORTESat
Designation FORTESat
/export/home/oracle7/RSAS/graphics/spacecraft_1

properties of the spacecraft

3-axis stabilization method used by spacecraft
Earth choose one of these as the orbit focus

Figure 5

Each spacecraft drawn in the Mission Editor can be expanded to reveal its own platform characteristics. The user expands (double clicks on) the spacecraft in the mission drawing. This causes the Platform Editor (Figure 6) for that spacecraft to be displayed.

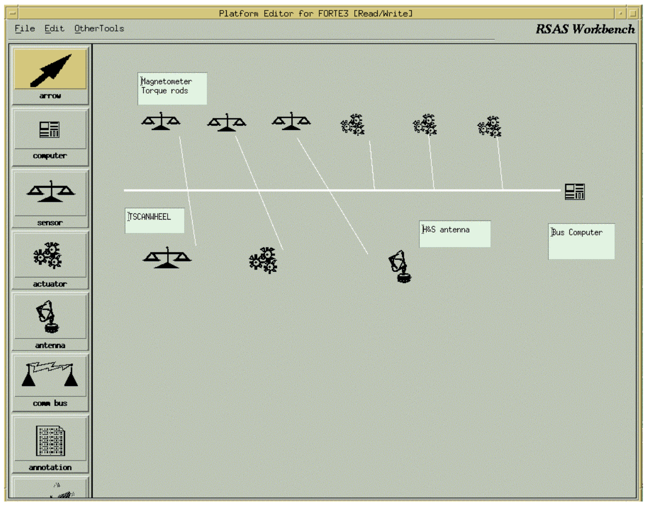
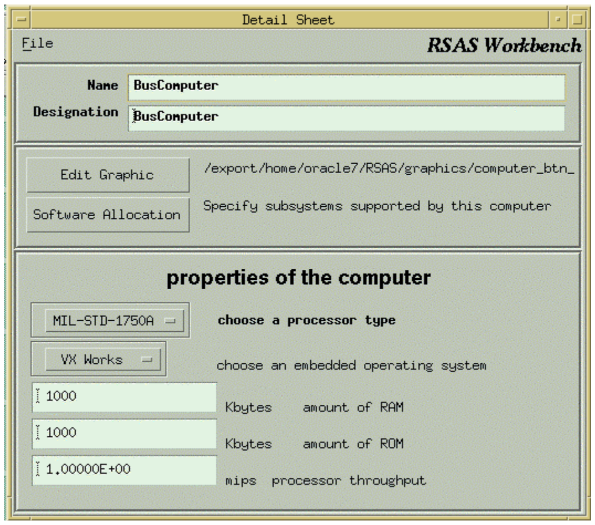


Figure 6

The Platform Editor captures the execution environment in which the flight computer's software operates. The environment may include actuators, sensors, antennae, computers, and buses. The Platform Editor also allows the user to allocate particular software subsystems to separate computers. The FORTE3 drawing indicates: a computer on the right; a bus drawn with white lines; three 2-axis magnetometers shown as scales above the bus; three torque rods indicated by gears above the bus; a scanwheel comprised of the scale and gears below the bus; and an antenna. As with the Mission Editor, the user may request any item's detail sheet.

The Properties of the Computer Detail Sheet (Figure 7) allows the user to specify the type of computer, its memory, and speed.



Name BusComputer
Designation BusComputer

/export/home/oracle7/RSAS/graphics/computer_btn_
Specify subsystems supported by this computer

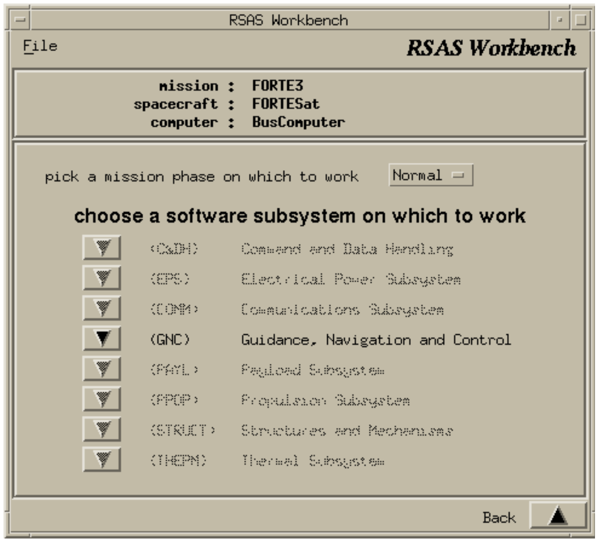
properties of the computer

MIL-STD-1750A choose a processor type
VX Works choose an embedded operating system
1000 Kbytes amount of RAM
1000 Kbytes amount of ROM
1.00000E+00 mips processor throughput

Figure 7

From the Platform Editor, the user may enter the Decision Model. The Decision Model captures a flight domain specialist's expertise by displaying a hierarchy of screens, and prompting the user for relevant information. The decisions made by the user are captured as values in a database, and are consulted when reusable components are selected and tailored.

The user expands the computer in the Platform Editor, causing the top-level Decision Model screen (Figure 8) for that computer to be displayed.



mission : FORTE3
spacecraft : FORTESat
computer : BusComputer

pick a mission phase on which to work Normal

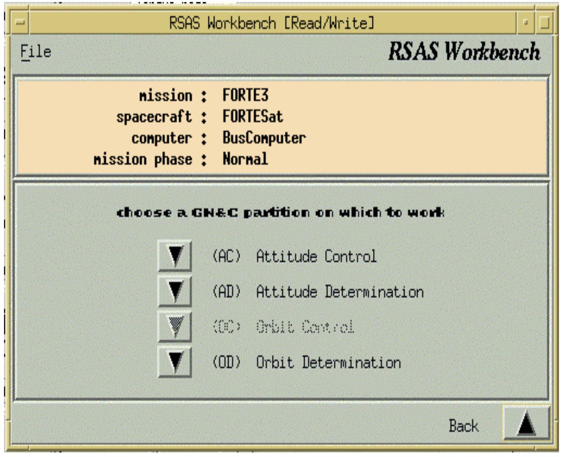
choose a software subsystem on which to work

(C&DH) Command and Data Handling
(EPS) Electrical Power Subsystem
(COMM) Communications Subsystem
(GNC) Guidance, Navigation and Control
(PAYL) Payload Subsystem
(PROP) Propulsion Subsystem
(STRUCT) Structures and Mechanisms
(THERM) Thermal Subsystem

Figure 8

The user may select from a menu of software subsystems. Subsystems are restricted to the ones that were allocated in the Platform Editor. The user also identifies the mission phase for which the software is to be specified. Mission phases were defined in the Mission Editor.

In this FORTE example, subsystems were limited to Orbit Determination (OD) and Attitude Determination (AD) components of Guidance, Navigation, and Control (GNC). Thus, only GNC is available and is selected (Figure 9).



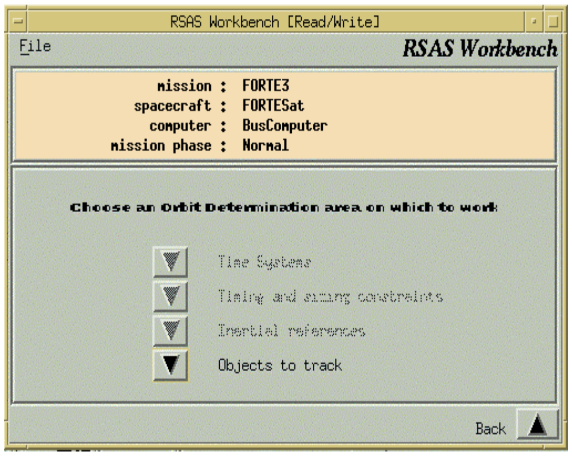
mission : FORTE3
spacecraft : FORTESat
computer : BusComputer
mission phase : Normal

choose a GN&C partition on which to work

(AC) Attitude Control
(AD) Attitude Determination
(OC) Orbit Control
(OD) Orbit Determination

Figure 9

In this example, the user selects Orbit Determination (Figure 10), and then Objects to track (Figure 11).


RSAS Workbench [Read/Write]

File

RSAS Workbench

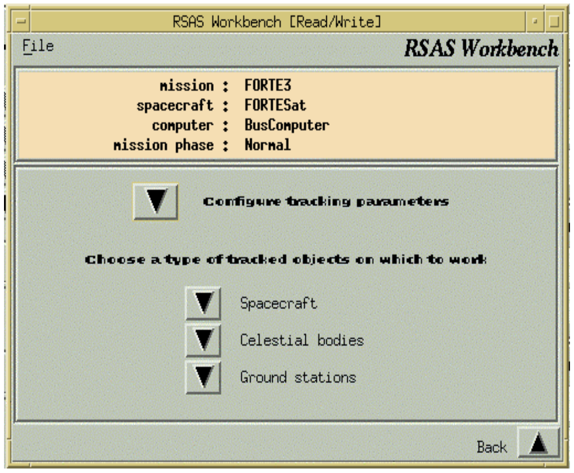
mission : FORTE3
spacecraft : FORTESat
computer : BusComputer
mission phase : Normal

Choose an Orbit Determination area on which to work

Time Systems
Timing and sizing constraints
Inertial references
Objects to track

Back

Figure 10


RSAS Workbench [Read/Write]

File

RSAS Workbench

mission : FORTE3
spacecraft : FORTESat
computer : BusComputer
mission phase : Normal

Configure tracking parameters

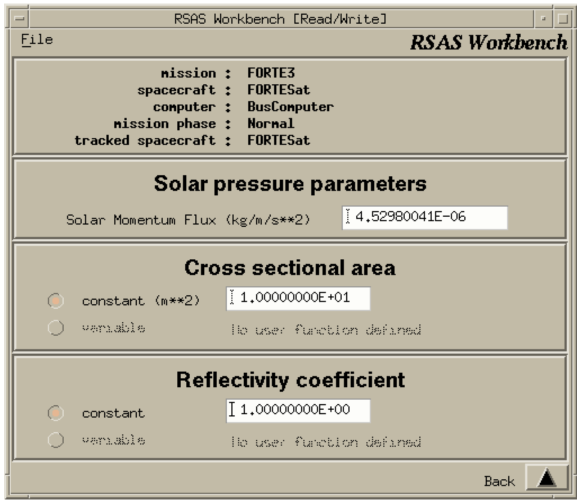
Choose a type of tracked objects on which to work

Spacecraft
Celestial bodies
Ground stations

Back

Figure 11

Eventually the user will be presented with the Solar Pressure user interface (Figure 12). This is an example of a bottom level node.


RSAS Workbench [Read/Write]

File

RSAS Workbench

mission : FORTE3
spacecraft : FORTESat
computer : BusComputer
mission phase : Normal
tracked spacecraft : FORTESat

Solar pressure parameters

Solar Momentum Flux (kg/m/s**2) 4.52980041E-06

Cross sectional area

constant (m**2) 1.00000000E+01
variable No user function defined

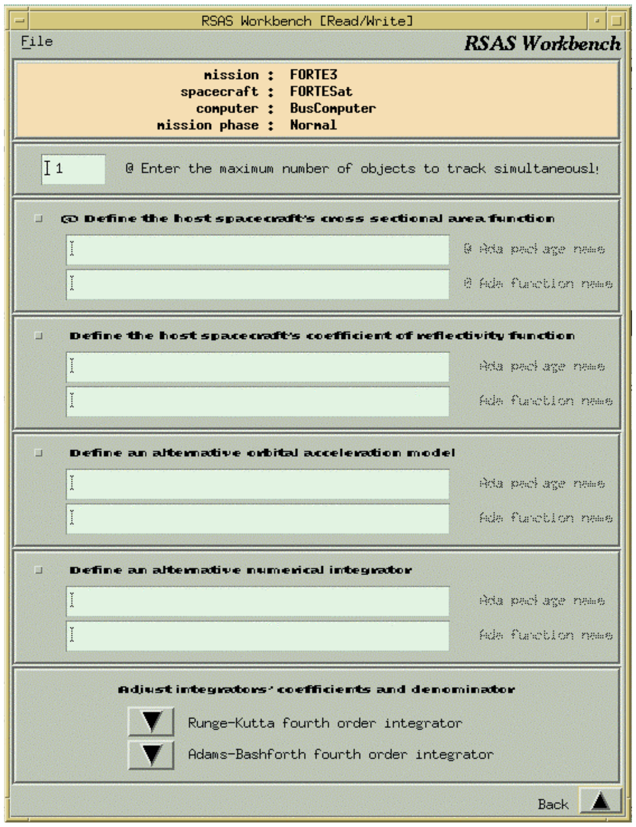
Reflectivity coefficient

constant 1.00000000E+00
variable No user function defined

Back

Figure 12

The engineer has the option to include user developed Ada packages. For example, the user may specify alternative Ada functions for the spacecraft's cross sectional area, reflectivity, and orbit. An alternate numerical integrator may be also be specified (Figure 13).


RSAS Workbench [Read/Write]

File

RSAS Workbench

mission : FORTE3
spacecraft : FORTESat
computer : BusComputer
mission phase : Normal

1 @ Enter the maximum number of objects to track simultaneousl

@ Define the host spacecraft's cross sectional area function
@ Ada package name
@ Ada function name

Define the host spacecraft's coefficient of reflectivity function
Ada package name
Ada function name

Define an alternative orbital acceleration model
Ada package name
Ada function name

Define an alternative numerical integrator
Ada package name
Ada function name

Adjust integrators' coefficients and denominator

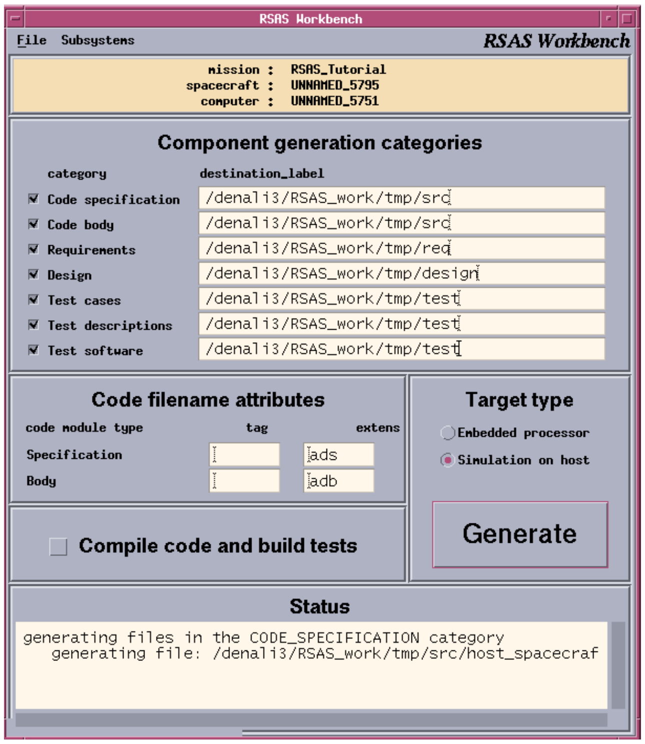
Runge-Kutta fourth order integrator
Adams-Bashforth fourth order integrator

Back

Figure 13

When the user is satisfied with the specification of the software for a particular computer, s/he may choose to generate components using the Component Generator. Ada specifications and bodies, requirements, design, and test cases are automatically generated upon user selection by the TRF2 tailoring tool. Note that the Component Generator has an option to automatically compile and build test cases. This option is intended to demonstrate that flight code will successfully compile, link and execute. The source code output of this example was 56 Ada body and specification files. Also generated were files containing Ada test codes, test results, design files, and requirement files.

The user may request assistance at any time from an On-Line Help capability using an Internet browser such as Netscape (Figure 14). Help can be either context sensitive help or a top-level index of help topics. Both modes of entry allow the user to completely navigate the help database, or set it aside for later use.



mission : RSAS_Tutorial
spacecraft : UNNAMED_5795
computer : UNNAMED_5751

Component generation categories

category	destination_label
✓ Code specification	/denali3/RSAS_work/tmp/src
✓ Code body	/denali3/RSAS_work/tmp/src
✓ Requirements	/denali3/RSAS_work/tmp/req
✓ Design	/denali3/RSAS_work/tmp/design
✓ Test cases	/denali3/RSAS_work/tmp/test
✓ Test descriptions	/denali3/RSAS_work/tmp/test
✓ Test software	/denali3/RSAS_work/tmp/test

Code filename attributes

code module type	tag	extens
Specification		ads
Body		adb

Target type: Embedded processor [✓ Simulation on host]

Compile code and build tests

Status

```
generating files in the CODE_SPECIFICATION category
   generating file: /denali3/RSAS_work/tmp/src/host_spacecraf
```

Figure 14

The Workbench operates on a Sun SPARC running Common Desktop Environment (CDE). The Workbench has a Motif user interface that is intuitive and familiar to most engineers. The lists in Figure 15 provide the Workbench's developmental and operational environment[8].

<u>Developmental Software</u>

? GNAT Ada compiler version 3.09

? PRO*C pre-compiler (ORACLE product)

? UIM/X GUI builder version 2.9

? Motif developer kit

? FileMaker Pro 3.0 (optional)

<u>Operational Hardware and Software</u>

? Sun SPARC station 2

? 32 Mbytes RAM

? 100 Mbytes disk space

? Color Terminal

? Solaris 2.5 or 2.5.1

? CDE (Common Desktop Environment)

? Netscape Navigator

? Shared Libraries (suggested)

? libMrm.so.3, libXm.so.3, libXt.so.5.0

? libX11.so.5.0, libm.so.1, libc.so.1

? TRF-2

? Access To Oracle Database (via SQL*NET)

? Oracle Product Version at least 7.0.16

Figure 15[7]

4. DOMAIN ANALYSIS

The RSAS Workbench development effort benefited from data developed in a concurrent Independent Research and Development (IR&D) project that Lockheed Martin Astronautics was conducting at the time. The IR&D focused on the development of reusable flight software components. Domain engineering was performed for three-axis stabilized spacecraft. This activity resulted in the generic spacecraft system architecture, the reusable software components, and the associated decision model that were subsequently used in the Workbench[7].

The Shlaer-Mellor Object-Oriented Analysis methodology as implemented by Cadre Teamwork was used to guide the domain engineering and the development of domain models. The domain analysis included the seven satellites shown in Figure 16 whose diversity provided a comprehensive view of satellite flight systems. These satellites ranged from low earth orbit satellites to interplanetary missions and include both civil and defense missions.

Program	Customer	Market Sector	Orbit type	Mission
GE-1 (A2100)	GE American	Business	Geo-stationary	Communication
Brilliant Pebbles	SDIO	Defense	Classified	Missile Defense
P-81	Classified	Defense	Classified	Classified
Magellan	NASA-JPL	Civil	Inter-planetary/ Low Altitude	Scientific
Lunar Resource Mapper	NASA-JPL	Civil	Inter-planetary/ Low Altitude	Scientific
Earth Observation System	NASA-Goddard	Civil	Low Earth Orbit	Remote Sensing/Earth Science
Brilliant Eyes	SDIO	Defense	Classified	Remote Sensing

Figure 16[8]

A representative spacecraft's flight software consists of seven bus subsystems and the payload. The bus subsystems are Guidance, Navigation & Control (GNC);

Communications (COMM); Command & Data Handling (C&DH); Electrical Power (EPS); Thermal (THERM); Structures & Mechanisms (STRUCT); and Propulsion (PROP). Bus subsystems support the payload in several ways such as pointing, controlling temperature, supplying electrical power, and commanding[5]. Subsystem plus payload high level relationships can be seem from the Subsystem Relationship model shown in Figure 17.

In this model, GNC is further broken into the Attitude Control (AC), Attitude Determination (AD), Orbit Control (OC), and Orbit Determination (OD) partitions.

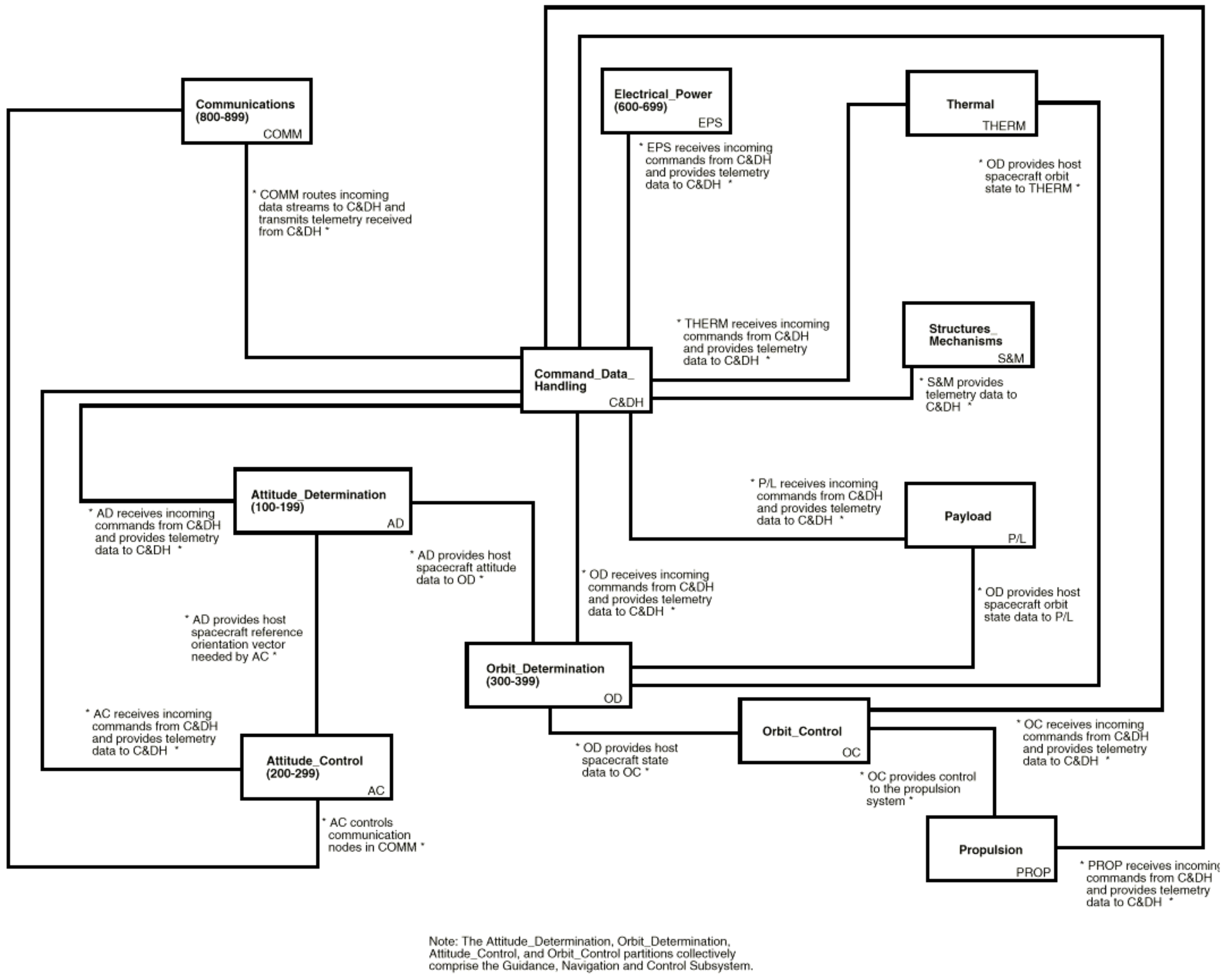

Communications (800-899) COMM
* COMM routes incoming data streams to C&DH and transmits telemetry received from C&DH *
Electrical_Power (600-699) EPS
* EPS receives incoming commands from C&DH and provides telemetry data to C&DH *
Thermal THERM
* OD provides host spacecraft orbit state to THERM *
* THERM receives incoming commands from C&DH and provides telemetry data to C&DH *
Structures_ Mechanisms S&M
* S&M provides telemetry data to C&DH *
Command_Data_ Handling C&DH
* P/L receives incoming commands from C&DH and provides telemetry data to C&DH *
Payload P/L
Attitude_Determination (100-199) AD
* AD receives incoming commands from C&DH and provides telemetry data to C&DH *
* AD provides host spacecraft attitude data to OD *
* OD receives incoming commands from C&DH and provides telemetry data to C&DH *
* OD provides host spacecraft orbit state data to P/L
* AD provides host spacecraft reference orientation vector needed by AC *
Orbit_Determination (300-399) OD
* AC receives incoming commands from C&DH and provides telemetry data to C&DH *
Orbit_Control OC
* OC receives incoming commands from C&DH and provides telemetry data to C&DH *
* OD provides host spacecraft state data to OC *
Attitude_Control (200-299) AC
* OC provides control to the propulsion system *
* AC controls communication nodes in COMM *
Propulsion PROP
* PROP receives incoming commands from C&DH and provides telemetry data to C&DH *
Note: The Attitude_Determination, Orbit_Determination, Attitude_Control, and Orbit_Control partitions collectively comprise the Guidance, Navigation and Control Subsystem.

Figure 17[8]

Besides the Subsystem Relationship Model there are Information Models, Object Communication Models, and other charts that are far too detailed to fit in this paper. A mechanism was required to aid the application engineer in tailoring and specifying the generic architecture to meet the requirements of a specific project. The mechanism is the Decision Model.

5. DECISION MODEL[9]

The Decision Model is a mechanism that groups and orders decisions giving every decision context (expressed as ancestors, siblings, and descendant decisions). It enables the human user to specify and software program to capture mission requirements necessary to select and tailor components for a well-suited solution to the mission. Every decision also has specific relevance, a decision can either be important or rendered unimportant (or irrelevant) by a previously made decision or decisions. The satellite engineer is guided through the decision tree by graphical user interfaces. The engineer's decisions and inputs are captured in a relational database (Oracle). These decisions and inputs are eventually used to tailor components to meet specific requirements.

The Decision Model or more visually an upside down tree (Figure 18) is constructed of nodes.

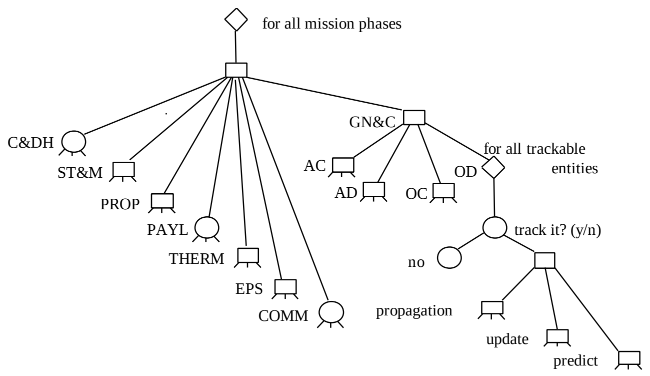


Figure 18

There are only three different types of nodes in this tree (Figure 19). These are decision nodes, decision group nodes, and iterated decision group nodes. One advantage of this structure is that it can be drawn and viewed in its entirety where relationships and decision sequences become evident. Decisions are represented graphically as ovals, decision groups as rectangles, and iterated groups as diamonds.

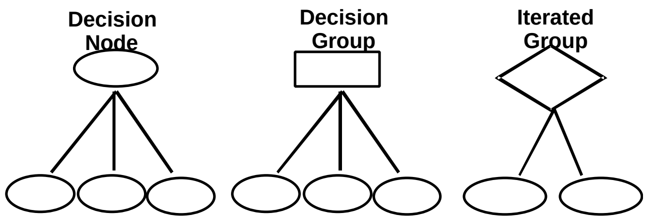


Figure 19

Decision nodes, i.e., the ovals, are the only places were actual user choices are recorded (persisting in a stored database). The value for solar momentum flux is a user choice (Figure 20).

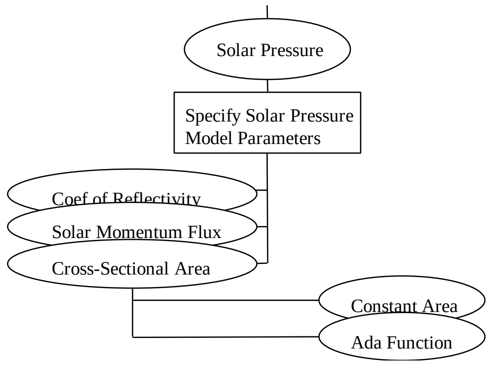


Figure 20

Decision groups, i.e., the rectangles, collect closely related nodes thereby reinforcing their relationship, decision groups. Gravity, solar pressure, and atmospheric drag are all in the perturbing acceleration group (Figure 21).

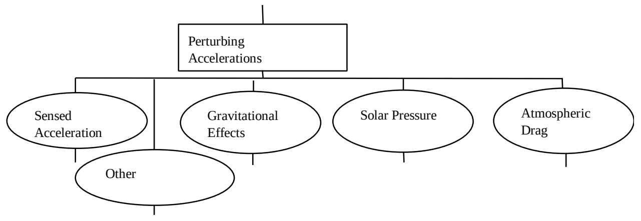


Figure 21

Iterated groups, i.e., the diamonds, are modified versions of decision groups. They support the situations where the same decisions must be made repeatedly dependent upon some previous decision. For example, variability may state that a spacecraft may have any number, N, of actuators (Figure 22). Each actuator's position must be expressed in body-fixed coordinates so that its measurements can be correctly interpreted. The Decision Model designer can't force all spacecraft to have the same number of actuators but once the user decides there are five then the location decision must be repeated precisely five times.

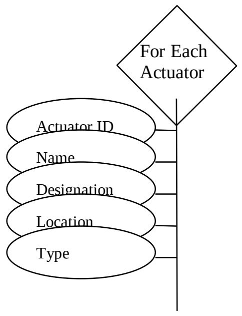


Figure 22

The hierarchical nature of the Decision Model is evident from a segment of the orbit determination model (Figure 23). The root, "Orbit Determination", is a decision group. Putting into words what the root node implies, "...which of the following do you want to tell me about now: time system, inertial references, object tracking, or timing and sizing constraints?" More generally, "You will eventually have to investigate all of my (relevant) children but which one do you want to investigate now?" By choosing "Objects to Track", the engineer can traverse the link to the chosen child and examine it.

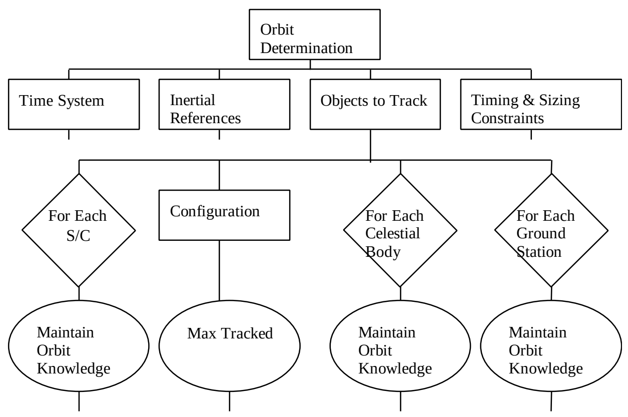


Figure 23

The Decision Model is, at this point, a rather large drawing. The next task was to construct a relational database that represents the Decision Model and also captures the satellite engineer's decisions and data.

6. RELATIONAL DATABASE[10]

The Decision Model maintains the engineer's choices in dynamic value tables. Logically, the Decision Model value tables are split into many different sets. Figure 24 provides a rather simple example of a Decision Model. The actual Decision Model is very deep and broad. Again, rectangles, ovals and diamonds represent decision groups, decisions, and iterated groups, respectively.

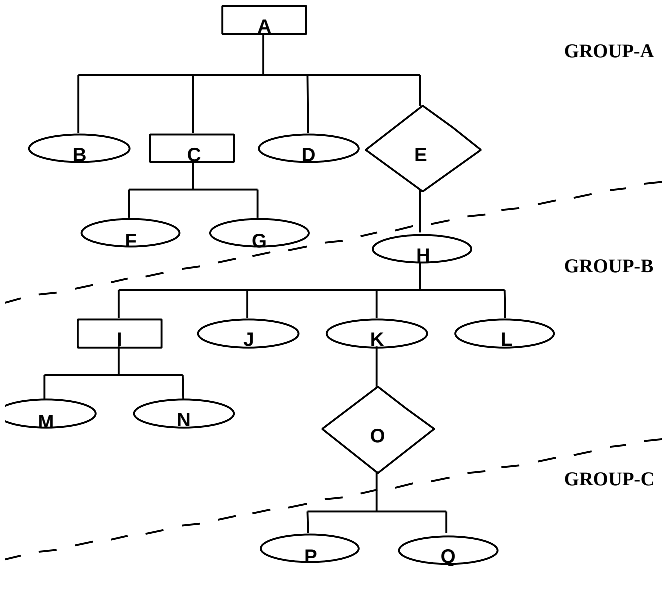


Figure 24

The logical table structure in the relational database represents of the decision hierarchy (Figure 25).

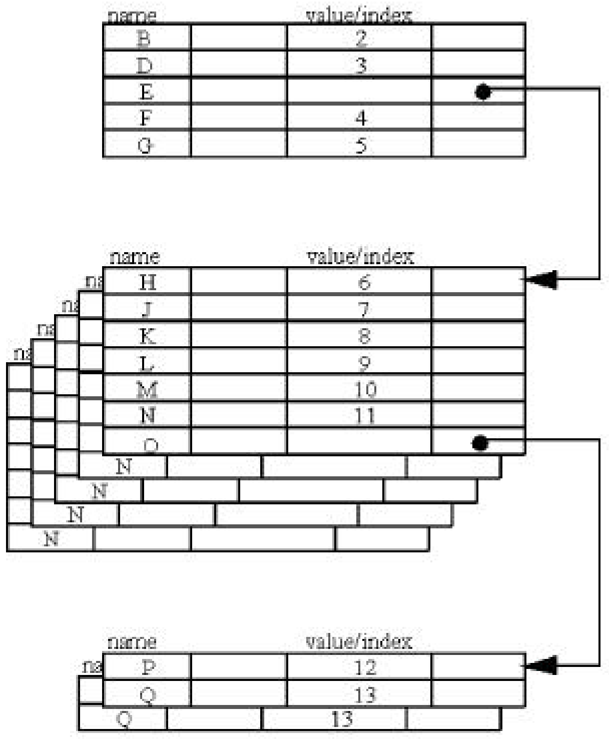


Figure 25

Iterated decision nodes, "E" and "O", require an upper bound to their iteration, and that the bound is in reality a value stored by a decision in a value table.

The three decision group nodes, "A", "C" and "I" don't have entries in the value tables. Decision group nodes are only for grouping. The limit for the iterated node "E" in the Group A table is the value of the decision "G". The limit for iterated node "O" in Group B is decision node "B". Note the number of copies of tables for Group B and C, The iterated groups force new tables distinct from the table in which the decision group node resides. An iteration limit value for an iterated group precisely defines the number of table copies of the repeated sub-tree. The Group A table is referred to as a top-level table. There is always a single copy of a top-level table and it can be allocated whenever a new mission is created. The DM dynamically allocates the tables of Groups B and C after the iteration value is set by the engineer.

7. WORKBENCH USAGE

The Workbench is installed in the Lockheed Martin Astronautics (LMA) Spacecraft Technology Center II (Denver, CO), the Lockheed Martin Missiles and Space (LMMS) Spacecraft Product Center (Sunnyvale, CA), and the Air Force Research Laboratory's Phillips Research Site.

In December 1997, the Workbench was used at Sunnyvale to support a demonstration of an application engineering process for the SBIRS (Space-Based Infrared System) Program. LMA and LMMS installations of the Workbench are configured to support application engineering in spacecraft flight-software development environments. These installations are essentially identical to the Workbench that was developed for the AFRL Space Vehicles Directorate.

The AFRL's Intelligent Satellite Systems Group (part of AFRL/VSSS) plans to integrate the Workbench into its Autonomous Payload Testbed that is currently under development.

8. CONCLUSION

This paper has presented the application engineering Workbench that was developed by the RSAS program.

The Workbench was developed to simplify and automate the reuse of spacecraft flight software. When we initiated this program, we strove to move past the stove-pipe-one-of-a-kind method of generating satellite flight software. Using the Workbench, a user is removed from much of the coding task and can concentrate on meeting mission requirements. A user's task is simplified because the production of nonsense or incorrect solutions is prevented by the enforcement of strict rules in the decision-making process. The value of the "Compile and Build Test Code" feature should not be underestimated. It demonstrates that generated code actually compiles and meets requirement specifications. Finally, the Workbench enables rapid prototyping and even product-line software development.

The Workbench incorporates a domain Decision Model that is used to determine the selection and tailoring of reusable components for a specific spacecraft application. Workbench technology does not have to be restricted to the spacecraft flight software domain. The decision model concept can be adapted for other domains in which domain-engineering tasks have been completed.

The Workbench supports Attitude Determination (AD), Attitude Control (AC), Orbit Determination (OD), communications antenna articulation, and solar array articulation components.

The RSAS program was a four-year research and development contract initiated and managed by the Air Force Research Laboratory's Space Sensing and Vehicle Control Branch. It was awarded after a competitive bid to Lockheed Martin Astronautics (then Martin Marietta), Denver, Colorado, in April 1994. The Workbench is currently installed at AFRL, Kirtland AFB, NM, and Lockheed Martin locations.

9. ACKNOWLEDGMENTS

The RSAS project was funded by the Producibility, Reliability, Accountability, and Maintainability (PRAM) Program Office (AFMC TTO/TTP), Wright-Patterson AFB, Ohio.

The development of the Workbench, and especially the Decision Model, was no small task. The Lockheed Martin program manager, Dr Ben Moultrie, the technical lead, Monte Ratajczyk, and Eric Chen and Karen Hover are all to be congratulated.

10. REFERENCES

[1] Common Ada Missile Packages (CAMP) Program, Air Force Armaments Laboratory, F08635-88-C-0002

[2] Domain Engineering Guidebook, SPC-92019.CMC Version 01.00.03, Dec 92, Software Productivity Consortium, N. Herndon, VA 90019

[3] FORTE home page, http://nis-www.lanl.gov/nis-projects/FORTE

[4] Hover, K. electronic correspondence

[5] Larson, W.J. and Wertz, J,R. (eds) Space Mission Analysis and Design, Second Edition, Microcosm Inc, Torrance, CA, and Kluwer Academic Publishers, London, The Netherlands, 1993

[6] Moultrie, B. Reusable Software Architecture for Spacecraft Tutorial, Lockheed Martin Astronautics, P.O. Box 179, Denver, CO 80201-0179, Contact No. F29601-94-C-0114

[7] Moultrie, B. Reusable Software Architecture for Spacecraft Final Project Report, Lockheed Martin Astronautics, P.O. Box 179, Denver, CO 80201-0179, Contact No. F29601-94-C-0114

[8] Satellite Domain Analysis Report for Reusable Software Architecture for Spacecraft, Martin Marietta Technologies, Inc., Astronautics, P.O. Box 179, Denver, CO 80201-0179, Contact No. F29601-94-C-0114.

[9] Ratajczyk, M. Reusable Software Architecture for Spacecraft, An Informal Explanation of the Decision, Lockheed Martin Astronautics, P.O. Box 179, Denver, CO 80201-0179, Contact No. F29601-94-C-0114.

[10] Ratajczyk, M., and Hover, K. Reusable Software Architecture for Spacecraft, Naming Database Values, Lockheed Martin Astronautics, P.O. Box 179, Denver, CO 80201-0179, Contact No. F29601-94-C-0114.