

Model-Based Guidelines for User-Centric Satellite Control Software Development

by
Somwang Thipphayathethana

M.S. Electrical Engineering
Chulalongkorn University, 2000

SUBMITTED TO THE SYSTEM DESIGN AND MANAGEMENT PROGRAM IN
PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

MASTER OF SCIENCE IN ENGINEERING AND MANAGEMENT
AT THE
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

JANUARY 2015

[February 2015]

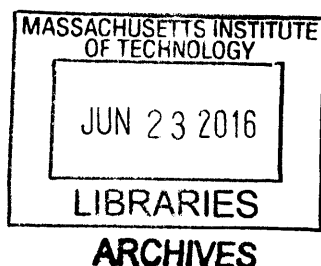
©2015 Somwang Thipphayathethana. All right reserved.

The author hereby grants to MIT permission to reproduce
and to distribute publicly paper and electronic
copies of this thesis document in whole or in part
in any medium now known or hereafter created.

Signature of Author: Signature redacted
Somwang Thipphayathethana
System Design and Management Program
January 5, 2015

Certified by: Signature redacted
Dov Dori
Visiting Professor at Engineering Systems Division
Thesis Supervisor

Accepted by: Signature redacted
Patrick Hale
Director
System Design and Management Program



This page intentionally left blank

Model-Based Guidelines for User-Centric Satellite Control Software Development

by
Somwang Thipphayathetthana

Submitted to the System Design and Management Program
on July 28, 2014 in Partial Fulfillment of the
Requirements for the Degree of Master of Science in
Engineering and Management

ABSTRACT

Three persistent common problems in satellite ground control software used by satellite controllers are obsolescence, lack of desired features and flexibilities, and endless software bug fixing. The obsolescence problem occurs when computer and ground equipment hardware become obsolete usually after only one third into the satellite mission lifetime. The satellite ground control software needs to be updated to accommodate changes on the hardware side, requiring significant work of satellite operators to test, verify, and validate these software updates. Software updates can also result from a new software version that offers new features or just fixes some bugs.

Trying to help solve these problems, an OPM model and guidelines for developing satellite ground control software have been proposed. The system makes use of a database-driven application and concepts of object-process orientation and modularity. In the new proposed framework, instead of coding each software function separately, the common base functions will be coded, and combining them in various ways will provide the different required functions. The formation and combination of these base functions will be governed by the main code, definitions, and database parameters. These design principles will make sure that the new software framework would provide satellite operators with the flexibility to create new features, and enable software developer to find bugs quicker and fix them more effectively.

Thesis Supervisor: Dov Dori

Title: Visiting Professor at Engineering Systems Division

Acknowledgements

Firstly, it would not be possible for me to attend the prestigious MIT's System Design and Management (SDM) program without monetary support from THAIKOM Plc. through the INTOUCH's scholarship program. Thank you THAIKOM's HR, Administration, Accounting, and Finance teams for the great support and coordination.

Secondly, I would like to thank you Dov Dori, Visiting Professor at Engineering Systems Division, MIT, for being my generous thesis advisor and for innovating a powerful Object Process Methodology (OPM) model that is the main tool used in this thesis.

Thirdly, I greatly appreciate the support from my cohort and some of SDM'13 fellows. You all make my life at MIT a marvelous experience.

Last but with the greatest importance, I am grateful to Patrick Hale, Director of SDM Program, for giving me the great opportunity to join in the wonderful SDM'14 cohort.

Table of Contents

ABSTRACT	3
Acknowledgements	4
Table of Figures	6
Introduction	7
Primary Research Objectives	8
Chapter 1. User Requirements.....	9
Basic Functional Requirements	9
Unmet Needs	10
Chapter 2. Requirements Analysis	12
Basic functional requirements	12
Unmet needs	13
Chapter 3. OPM Ground Control Software Model	15
The database-driven architecture solution concept.....	15
Modular interface or device driver	15
OPM System Diagram – The Top-Level System View	16
OPL of SD	16
OPL of SD1	17
OPL of SD1.1	18
OPL of SD1.2	19
OPL of SD2	20
OPL of SD2.1	22
OPL of SD2.1.1	23
OPL of SD2.2	24
OPL of SD2.2.1	25
OPL of SD2.2.2	26
OPL of SD2.3	27
OPL of SD2.3.1	28
OPL of SD2.4	29
OPL of SD2.5	30
OPL of SD2.6	31
OPL of SD2.6.1	32
OPL of SD2.7	33
OPL of SD3	35
OPL of SD4	37
Chapter 4. Guidelines for Satellite Ground Control Software	38
Proposed guidelines and their motivation	38
Chapter 5. Conclusion and Discussion	44
Future Research.....	44
Bibliography	45

Table of Figures

FIGURE 1. BASIC FUNCTIONS OF SATELLITE GROUND CONTROL SOFTWARE	9
FIGURE 2. SD DIAGRAM OF SATELLITE CONTROLLING SYSTEM	16
FIGURE 3. SD1 SATELLITE CONTROLLING IN-ZOOMED	17
FIGURE 4. SD1.1 INITIALIZING IN-ZOOMED	18
FIGURE 5. SD1.2 LOGGING IN IN-ZOOMED	19
FIGURE 6. SD2 - SATELLITE CONTROLLING UNFOLDED	20
FIGURE 7. SD2.1 COMMANDING IN-ZOOMED	22
FIGURE 8. SD2.1.1 TELECOMMAND SENDING IN-ZOOMED	23
FIGURE 9. SD2.2 DATA DISPLAYING IN-ZOOMED	24
FIGURE 10. SD2.2.1 SYSTEM REQUESTED DISPLAYING IN-ZOOMED	25
FIGURE 11. SD2.2.2 USER REQUESTED DISPLAYING IN-ZOOMED	26
FIGURE 12. SD2.3 DATA COLLECTING & PROCESSING UNFOLDED	27
FIGURE 13. SD2.3.1 TELEMETRY PROCESSING IN-ZOOMED	28
FIGURE 14. SD2.4 DATA SET UNFOLDED	29
FIGURE 15. SD2.5 DATABASE MANAGING IN-ZOOMED	30
FIGURE 16. SD2.6 DATABASE SET UNFOLDED	31
FIGURE 17. SD2.6.1 COMMAND DEFINITION SET UNFOLDED	32
FIGURE 18. SD2.7 GROUND EQUIPMENT SET UNFOLDED	33
FIGURE 19. SD3 SATELLITE UNFOLDED	35
FIGURE 20. SD4 SATELLITE CONTROLLING SYSTEM UNFOLDED.....	37

Introduction

As satellite operators' day-to-day operations rely heavily on satellite ground control software, a decision of great significance for satellite owners is how to select a satellite ground control system for controlling their fleet. Over many years, several commercial off-the-shelf (COTS) satellite control software products have been introduced into the market. These COTS products offer both standard functions [1] and specific features that help make the job of satellite controllers easier. While the satellite owner or operators must choose one product, different COTS software products have different pros and cons. One product might have very good user interface, but limited non-standard features or lack of user-customized functions. Another product that might have poor user interface and slow learning curve offers a more complete range of features and customized functions.

Main metrics for selecting COTS products are the maturity of the product, the expertise and experience of the product developer, standard functions and special features of the product, product price, and quality of after-sale service. Unfortunately, these metrics provide satellite owners with little help as they are faced with the need to select the best COTS product for their needs. Satellite operators using COTS satellite ground control software have repeatedly experienced a variety of problems [1, 2]. Three main problems are the following.

- **Obsolescence:** A communication satellite is nominally designed for 12-20 years of service, but many pieces of ground equipment, such as the antenna control unit, cortex, and server/workstation computers, have shorter lifespan of 6-10 years. This ground equipment will become obsolete and be discontinued, making the control software at all levels unusable. The alternatives are to look for expensive rare equipment in the second-hand market, or to invest in developing a new version of the missing control software. Clearly, none of these options is attractive.
- **Lack of desired features:** One piece of software can do some useful things that others cannot, but users have to choose only one and work around to overcome the lack of desired but missing features. Moreover, sometimes the desired and promised features do not function as users expected, so they have to develop their own tools to achieve these defunct features.
- **Endless bug fixing:** Everyone knows and accepts the fact that no software is bug-free. However, it is painful to find bugs, collect data for problem reproduction, and then send a report to the software developers. If and when software developers fix a reported problem, they often deliver a new software version that is plagued with new problems, leaving users stuck in an endless loop of bug fixing.

Primary Research Objectives

This research was aimed at helping satellite owners in either selecting COTS software or developing their in-house software by providing an OPM (Object-Process Methodology [3]) model-based framework and guidelines for software selection or development. The research focuses on users' requirements collected by reviewing the literature and looking for common problems of satellite ground control software and users' complaints about satellite ground control software.

Taking a model-based systems engineering (MBSE) approach, analysis of user requirements and common problems reveal their unmet needs. These, in turn, are used to define and conceptually model a set of requirements for developing satellite ground control software or selecting some vendor-specific satellite control software package.

Based on users' feedback and derived requirements, a conceptual OPM model of satellite ground control software has been developed and tested. The OPM model-based framework provides practical guidelines for developing satellite control software or selecting a vendor-developed satellite control software package.

Chapter 1. User Requirements

In this thesis, user requirements are divided into two groups. The first group includes basic functional requirements that come directly from standard, general functions of satellite control software. The second requirements group are those related to specific features and users' common problems and complaints; it is the collection of users' unmet needs which reflect the satellite ground control software quality and vendors' after sale or maintenance services.

Basic Functional Requirements

Users' basic functional requirements for ground control software are mainly derived from the functions and components of the satellite. Thus, the complexity of users' requirements is in line with the complexity of the satellite and its components. When users require high-level automatic functions, high availability, or ground control intelligent features, the complexity of the ground control software can be high making it expensive [4-7]. In this research, the basic functions of ground control satellite are partially based on a simple model discussed by Straka [1].

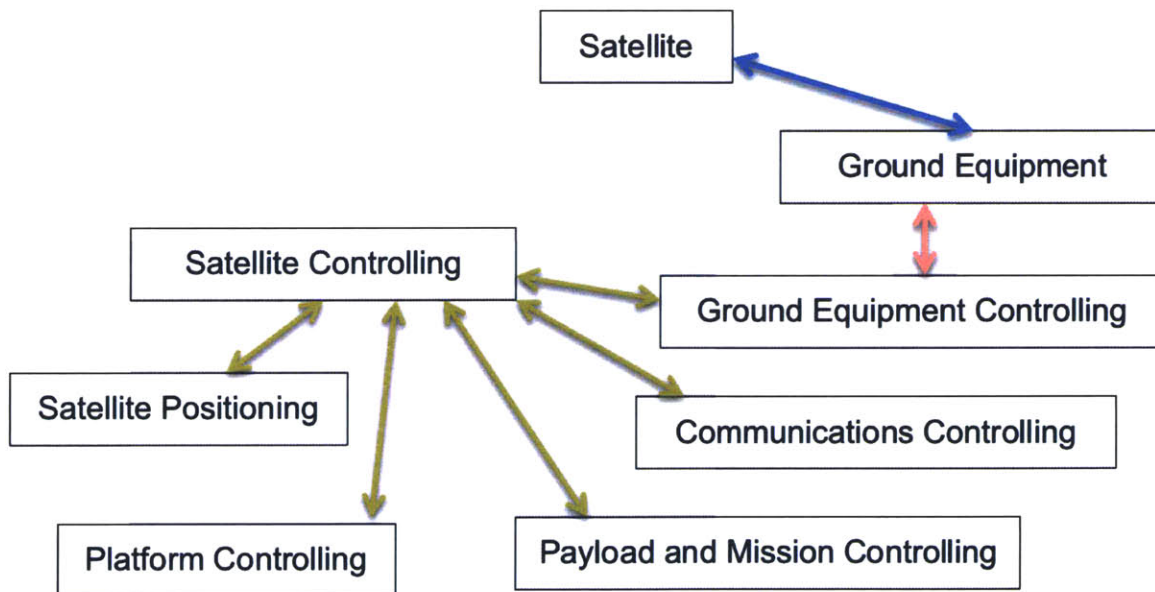


Figure 1. Basic functions of satellite ground control software

Figure 1 is a simple block diagram that shows the main function of ground control software called **Satellite Controlling**. Under this main function are five sub-functions: **Ground Equipment Controlling**, **Satellite Positioning**, **Communications Controlling**, **Platform Controlling**, and **Payload and Mission Controlling**. **Satellite Controlling** is an overarching system function that oversees all the services and interfaces to both its subsystems and the users. **Ground Equipment Controlling** provides all ground equipment related

services to the **Satellite Controlling** main function. **Satellite Positioning** processes the satellite's range and position provided by the main function and returns the attitude and orbit of satellite back to the main function. **Communications Controlling** takes care of the status and parameters of the satellite's communication system. **Platform Controlling** monitors and controls all the satellite's platform units. Finally, **Payload and Mission Controlling** manages the satellite's payload configuration and monitors the performance of the satellite's missions or services.

Each sub-function monitors, determines, and reports its current status and trend of the units or processes it is responsible for back to main function by using telemetry, radar, and other observations and measurements. The main function propagates this detailed information, including warnings and alarm messages, to users and subsystems. The main function can automatically perform certain predefined actions, including sending commands to ground equipment or to satellite, or setting off warning or alarm status and messages in response to a reported status or trend from its subsystems. Users or system analysts analyze the information received from the main function and take immediate actions or plan scheduled operations, such as sending out command to ground control units or satellite. The main function generates, checks, sends, and verifies the command requested by both its internal automatic functions and users' inputs.

Unmet Needs

When it is time to upgrade the satellite computer hardware, or when ground equipment is broken, satellite operators often get frustrated to find out that the model of that equipment is discontinued and there are no spare parts available in the market. They then learn from the ground control software vendor that if they upgrade their satellite computer hardware or operating system, or change the model of ground equipment, they have to upgrade the ground control software as well.

Even in the most mature organizations, those that have achieved CMM level 5 [8], upgrading the COTS products is still a challenge. Anderson and McAuley [2] shared their experience of a hardware upgrade event that mandated complementary upgrade of many COTS products. They describe the many obstacles they faced, ranging from changing the primary contractor to COTS product vendors or developers, going through acquisition and merging process, to the difficulty of getting full attention from COTS software developers in order to solve the issues reported during software upgrading. These hardships caused schedule delays and over-budgeted man-years effort.

Straka [1] described eight concerns associated with the COTS software, which included the software not being fully compliant with user's requirements, vendor or developer going out of business, obsolescence, software interface issues, standardization of COTS software, responsiveness of COTS vendor or developer, software features flexibility, and hidden cost of COTS software, such

as training and maintenance cost. By and large, these complaints and concerns can be grouped into three major issues: (1) missing desired features of COTS software, (2) obsolescence, and (3) poor quality of vendor services.

Three main unmet needs from satellite software, in addition to serving the basic functions, arise from the literature review and are phrased below as fundamentally non-functional requirements:

1. The ground control software shall provide flexibility and customizability (customizing capabilities) for the satellite system.
2. The ground control software shall support upgrades or changes in the hardware or operating system.
3. The ground control software vendor or developer shall provide responsive high-quality technical support, bug correction, and troubleshooting training.

Chapter 2. Requirements Analysis

As this research focuses on users' unmet needs, the basic functional requirements from the usual standard COTS software system are stated below in a short, simplified version.

Basic functional requirements

1. **Satellite Controlling** shall provide overall services and interfaces to its subprocesses and users. Satellite Controlling shall:
 - Collect and process data from the satellite, the ground equipment, and its own subprocesses listed in items 2-6 below,
 - Display and archive the raw and processed data, and
 - Generate, validate, execute, and verify commands to change satellite status or parameters, ground equipment, and its own subprocesses.
2. **Ground Equipment Controlling** shall provide all ground equipment related services. Ground Equipment Controlling shall:
 - Collect and process data from ground equipment, and
 - Help Satellite Controlling generate, validate, execute, and verify commands to change ground equipment status or parameters.
3. **Satellite Positioning** shall handle position-related computation:
 - Collect and process satellite range and position data, and
 - Calculate the attitude and orbit of satellite.
4. **Communications Controlling** shall command and control the communication system of the satellite:
 - Collect and process status and parameter of the satellite communication system, and
 - Help Satellite Controlling generate, validate, execute, and verify commands to change status or parameters of the satellite communication system.
5. **Platform Controlling** shall monitor and control all satellite platform units:
 - Collect and process status and parameter of all satellite platform units, and
 - Help Satellite Controlling generate, validate, execute, and verify commands to change status or parameter of all platform units.
6. **Payload and Mission Controlling** shall manage the satellite's payload configuration and maintain the satellite mission or service performance:
 - Collect and process status and parameters related to payload configuration and mission, and
 - Help Satellite Controlling generate, validate, execute, and verify commands to change status or parameter related to the satellite payload configuration and mission.

Unmet needs

1. The ground control software shall be flexible and customizable. It shall provide customizing capabilities for the specific system.
 - The ground control software shall be divided into separate modules, each coded as a database-driven function, such that in order to change or customize a function, the user will be able to just modify the related function in the database, principally without affecting other modules.
 - The ground control software shall provide a tool for user to create and modify database.
 - The ground control software shall provide database translator that allow user to convert manufacturers' database to the database that is used by the software.
 - The ground control software shall provide a tool for user to create and customize the software's functions.
 - The ground control software shall allow user to define names and locations of the data recorded files generated by the software.
 - The ground control software shall allow users to create and modify or display format of the displayed data by modifying and creating the database without any change in the software code
 - The ground control software shall allow users to create new commands and modify existing commands by creating and modifying the database without any change in the software code.
 - The inter-module interfaces of each module shall be clearly defined and include definitions and examples of input and output.
2. The ground control software shall support upgrades and changes in the hardware and/or in the operating system.
 - Using this modular and database-driven function model, upgrading or changing in software code shall be easy to manage and limited in scope.
 - The ground control software shall be coded with platform- and operating-system-independent programming languages.
 - The re-coding of ground control software due to any hardware change shall affect only the changed hardware related modules, and the required re-coding shall not propagate and be invisible to other modules and main processes.
 - The ground control software shall be able to use and display all data recorded before the software upgraded, or shall provide a tool to convert all data generated by the software in older versions to the data that can be used and displayed by the upgraded software.
3. The ground control software vendor or developer shall provide high-quality technical support, bug correction, and troubleshooting training.
 - The ground control software system shall keep all system information and error exceptions and provide error records in a human understandable format and intelligible language.

- The ground control software shall record and provide report of software events, for example the successful and failure of processes' initialization.
- The ground control software shall record and provide report of users' activities, such as user's logging in to the system, user's request to send a command, and database modification by user.

Chapter 3. OPM Ground Control Software Model

In this chapter we discuss the concept of the proposed software system and the ways it solves the problems specified in the previous chapters. We then present the OPM model of the ground control software and its merits.

The database-driven architecture solution concept

Based on the functional requirements, the ground control software will be nominally divided by the different functions matched with the satellite subsystem functions. Adopting a database-driven application architecture concept [9,10], each functional requirement is assigned to a generic module, governed by different set of database elements. Thus, instead of having a different software code for each module, which is lacking functionality, the database will comprise a small set of generic modules, each providing a basic common functionality.

Rather than having six different functional models—**Satellite Controlling, Ground Equipment Controlling, Satellite Positioning, Communications Controlling, Platform Controlling, and Payload and Mission Controlling**—the Database Set shall comprise three common function modules: **Data Displaying, Data Collecting & Processing, and Commanding**. This database-driven architecture will accommodate the flexible and new features requirement.

Modular interface or device driver

To address the hardware obsolescence problem, the concept of modular interface or device driver is added as a software development requirement. With this modular interface or device driver, in case the hardware becomes obsolete or needs to be changed, only the device driver or interface module related to that specific hardware would need to be recoded.

To cope with a change of the operating system, a requirement on the choice of programming language shall be imposed. The programming language being used shall be readily portable to the new OS version or even across different OSs. **System Data Recording** and **Activity Recording** that will record all activities and system events will facilitate system troubleshooting and debugging.

OPM System Diagram – The Top-Level System View

The OPM model of **Satellite Controlling System** consists of several levels of Object-Process Diagrams (OPDs). The top level, called the System Diagram (SD), shows the overview of **Satellite Controlling System**. In Figure 2, the **Operator Group** is an agent (or actor in SysML terms) who is the group of professionals who handles the **Satellite Controlling** process. The **Satellite Owner**, an environmental object representing the owner of the **Satellite**. The **Satellite Controlling** process affects both **Satellite**, which is an environmental object, and **Satellite Controlling System**.

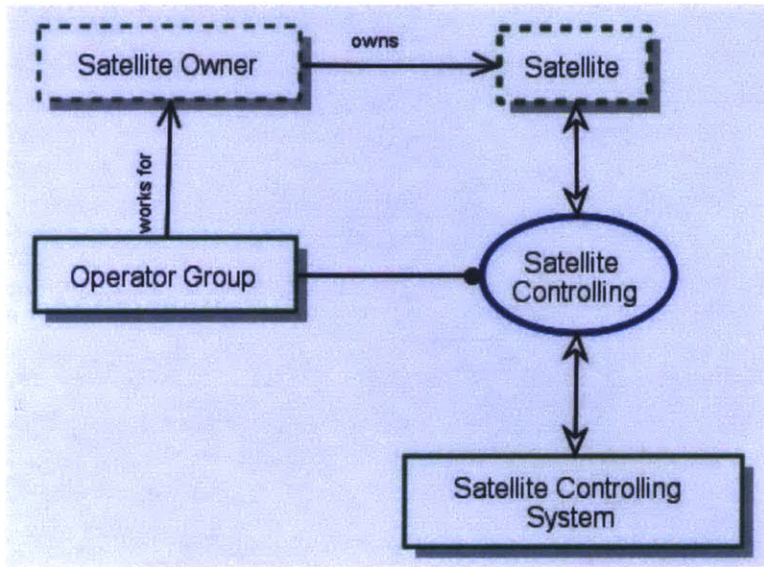


Figure 2. SD diagram of Satellite Controlling System

OPL of SD

Satellite is environmental and physical.

Satellite Owner is environmental and physical.

Satellite Owner owns Satellite.

Operator Group is physical.

Operator Group works for Satellite Owner.

Operator Group handles Satellite Controlling.

Satellite Controlling System is physical.

Satellite Controlling affects Satellite and Satellite Controlling System.

The following OPDs elaborate on details of **Satellite**, **Satellite Controlling** process, and **Satellite Controlling System**, and also demonstrate the effects and interactions between **Satellite Controlling** subprocesses and the components and processes of **Satellite** and **Satellite Controlling System**.

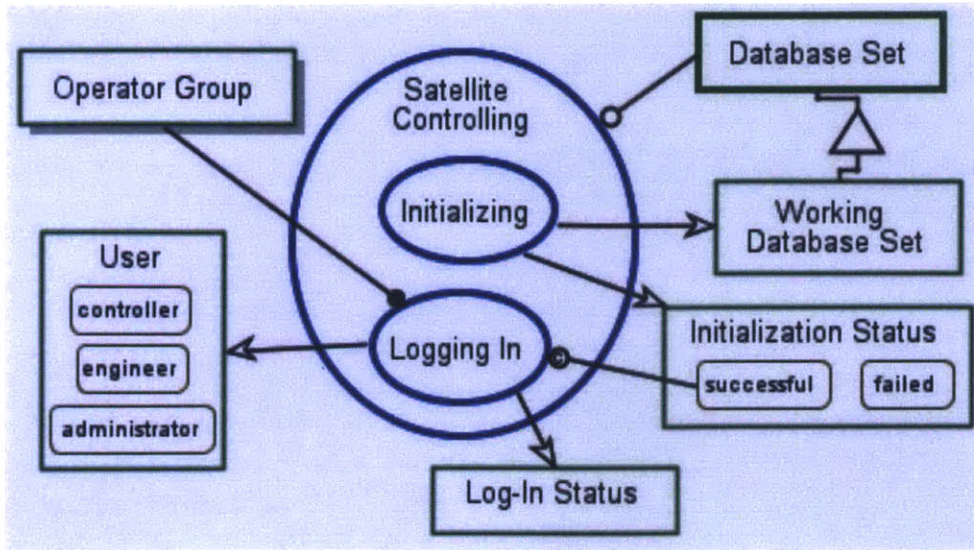


Figure 3. SD1 Satellite Controlling in-zoomed

OPL of SD1

Operator Group is physical.

Operator Group handles Logging In.

User can be controller by default, engineer, or administrator.

Initialization Status can be successful or failed.

Working Database Set is a Database Set.

Satellite Controlling requires Database Set.

Satellite Controlling zooms into Initializing and Logging In.

Initializing yields Working Database Set and Initialization Status.

Logging In occurs if Initialization Status is successful.

Logging In yields User and Log-In Status.

The in-zoomed of **Satellite Controlling** process is shown in Figure 3. **Initializing** and **Logging In** are subprocesses of **Satellite Controlling**. The **Initializing** subprocess is the first process to be executed, and as a result, **Working Database Set**, used by other processes, is created. Then, if the system is successfully initialized, **Operator Group** is allowed to log into the system using the second subprocess, **Logging In**. If the Logging In process is successful, a system's **User** is created. The **Initialization Status** and **Log-In Status** are also generated and recorded.

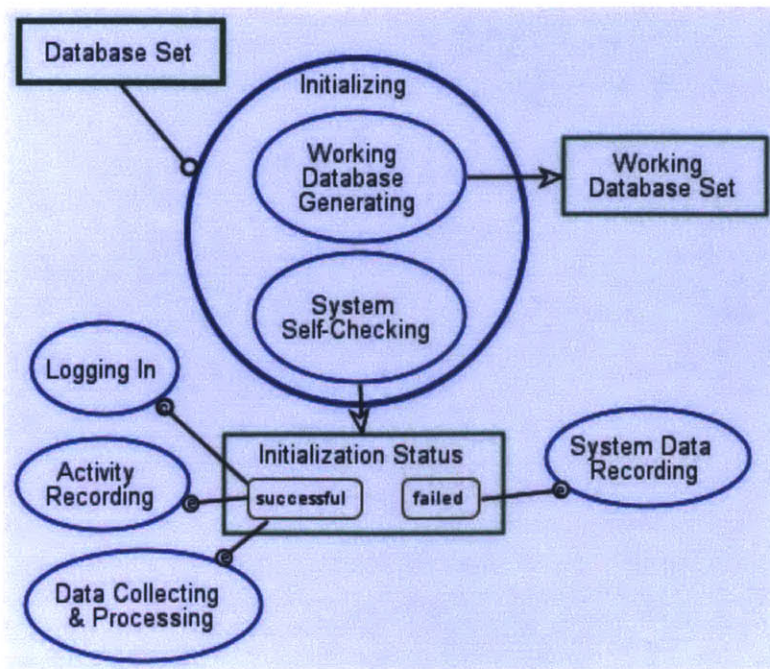


Figure 4. SD1.1 Initializing in-zoomed

OPL of SD1.1

Initialization Status can be successful or failed.

Initialization Status triggers Data Collecting & Processing and Activity Recording when it enters successful.

Initialization Status triggers System Data Recording when it enters failed.

Data Collecting & Processing requires successful Initialization Status.

System Data Recording requires failed Initialization Status.

Logging In occurs if Initialization Status is successful.

Activity Recording requires successful Initialization Status.

Initializing requires Database Set.

Initializing zooms into Working Database Generating and System Self-Checking.

Working Database Generating yields Working Database Set.

System Self-Checking yields Initialization Status.

The in-zoomed view of the **Initializing** process (see Figure 4) shows two subprocesses: **Working Database Generating** and **System Self-Checking**. When executed, **Working Database Generating** creates a **Working Database Set**. Then, **System Self-Checking** checks the readiness of the overall system and reports the **Initialization Status** to the **Activity Recording** process if system initialization is successful, or to the **System Data Recording** process if the Initializing process fails.

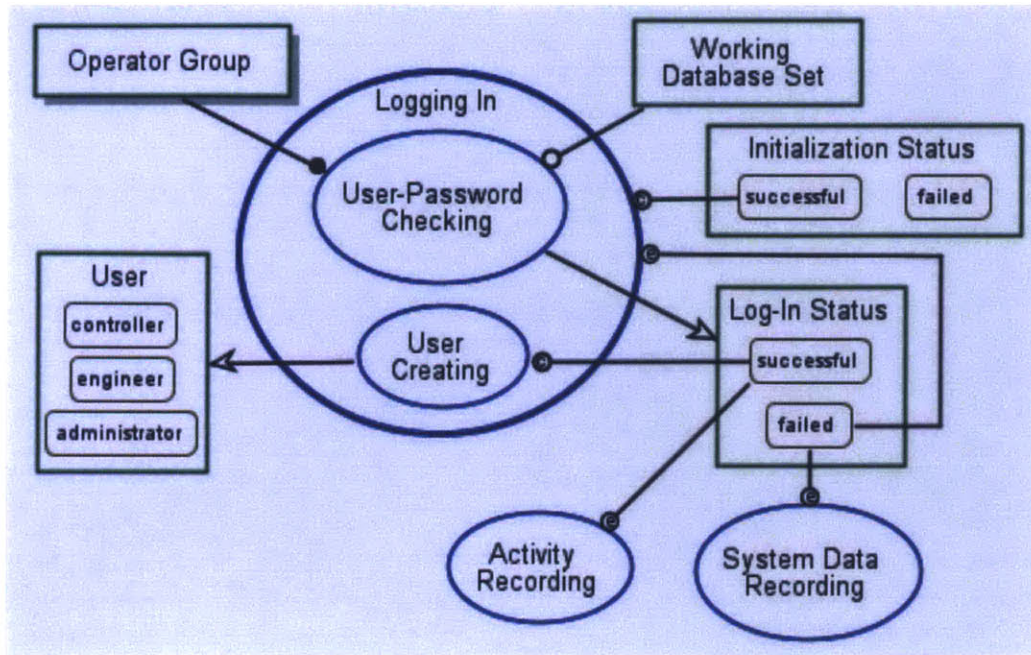


Figure 5. SD1.2 Logging In in-zoomed

OPL of SD1.2

Operator Group is physical.

Operator Group handles User-Password Checking.

User can be controller by default, engineer, or administrator.

Initialization Status can be successful by default or failed.

Log-In Status can be successful or failed.

Log-In Status triggers Activity Recording when it enters successful.

Log-In Status triggers Logging In and System Data Recording when it enters failed.

Activity Recording requires successful Log-In Status.

System Data Recording requires failed Log-In Status.

Logging In occurs if Initialization Status is successful.

Logging In requires failed Log-In Status.

Logging In zooms into User-Password Checking and User Creating.

User-Password Checking requires Working Database Set.

User-Password Checking yields Log-In Status.

User Creating occurs if Log-In Status is successful.

User Creating yields User.

The in-zoomed view of **Logging In** shows two subprocesses: **User-Password Checking** and **User Creating** (see Figure 5). **Operator Group** interacts with **User-Password Checking**, which authenticates **Operator Group** by checking the user name and password stored in the **Working Database Set**. **User-Password Checking** provides **Log-In Status** that signifies the success or failure of the **Logging In** process. If **Logging In** succeeds, **User Creating** will be

executed and create a **User**, and then **Activity Recording** will be activated to record the logging in activity. If **Logging In** fails, the **Logging In** process will be triggered to allow **Operator Group** to try logging in again, and **System Data Recording** will be triggered to record the attempt of user logging in.

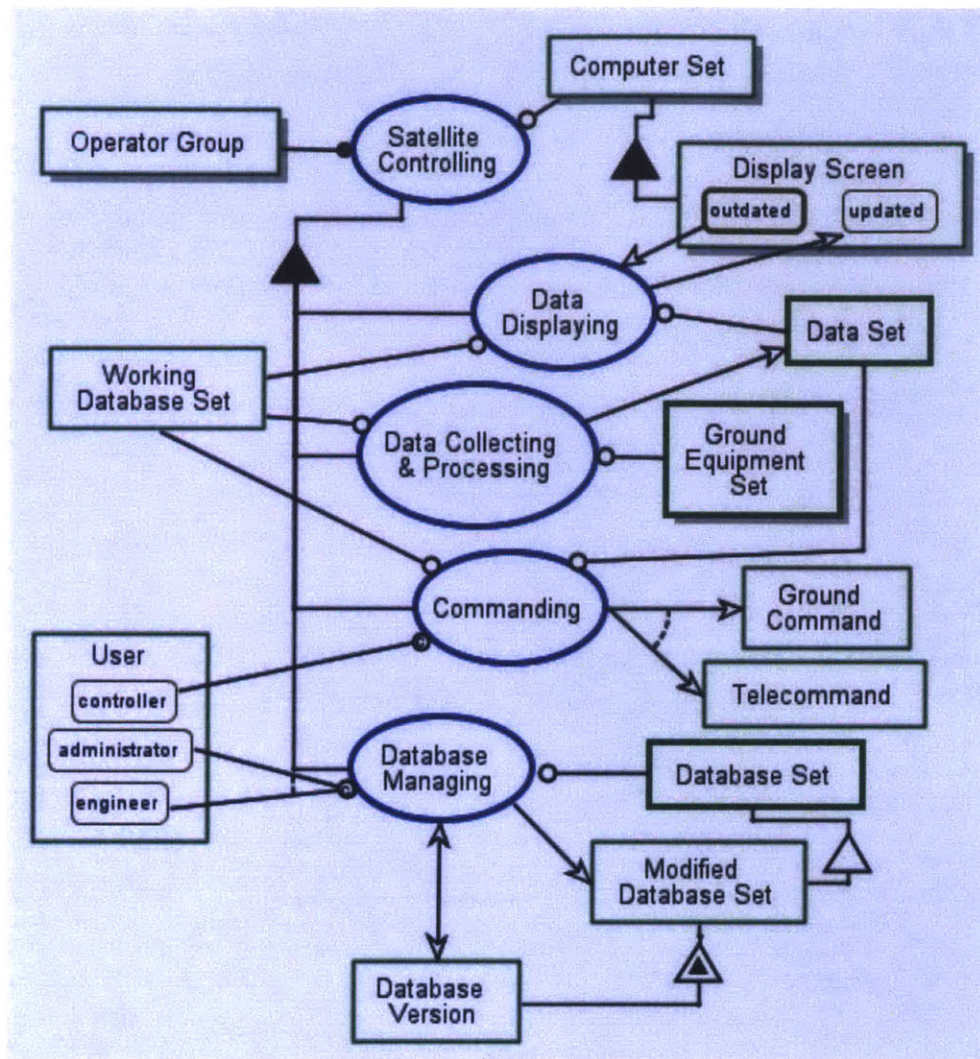


Figure 6. SD2 - Satellite Controlling unfolded

OPL of SD2

Operator Group is physical.

Operator Group handles Satellite Controlling.

Computer Set is physical.

Computer Set consists of Display Screen.

Display Screen is physical.

Display Screen can be outdated or updated.

outdated is initial.

Ground Equipment Set is physical.
 User can be controller by default, engineer, or administrator.
 Modified Database Set is a Database Set.
 Modified Database Set exhibits Database Version.
 Satellite Controlling consists of Data Collecting & Processing, Database Managing, Commanding, and Data Displaying.
 Data Collecting & Processing requires Working Database Set and Ground Equipment Set.
 Data Collecting & Processing yields Data Set.
 Database Managing occurs if either User is administrator or User is engineer.
 Database Managing requires Database Set.
 Database Managing affects Database Version.
 Database Managing yields Modified Database Set.
 Commanding occurs if User is controller.
 Commanding requires Working Database Set and Data Set.
 Commanding yields either Ground Command or Telecommand.
 Data Displaying requires Data Set and Working Database Set.
 Data Displaying changes Display Screen from outdated to updated.
 Satellite Controlling requires Computer Set.
 Satellite Controlling zooms into Initializing and Logging In.

Figure 6 illustrates four asynchronous subprocesses of **Satellite Controlling**: **Data Displaying**, **Data Collecting & Processing**, **Commanding**, and **Database Managing**. When requested by **Operator Group**, **Data Displaying** process reads, formats, and displays data from **Data Set**. **Data Collecting & Processing** can be triggered by different system events to collect and process data related to those events. **Operator Group** who is logged in as controller is allowed to send commands to both **Ground Equipment Set** and **Satellite** via the **Commanding process**. **Operator Group** who is logged in as administrator or engineer is allowed to make changes to the **Database Set** by calling the **Database Managing** process.

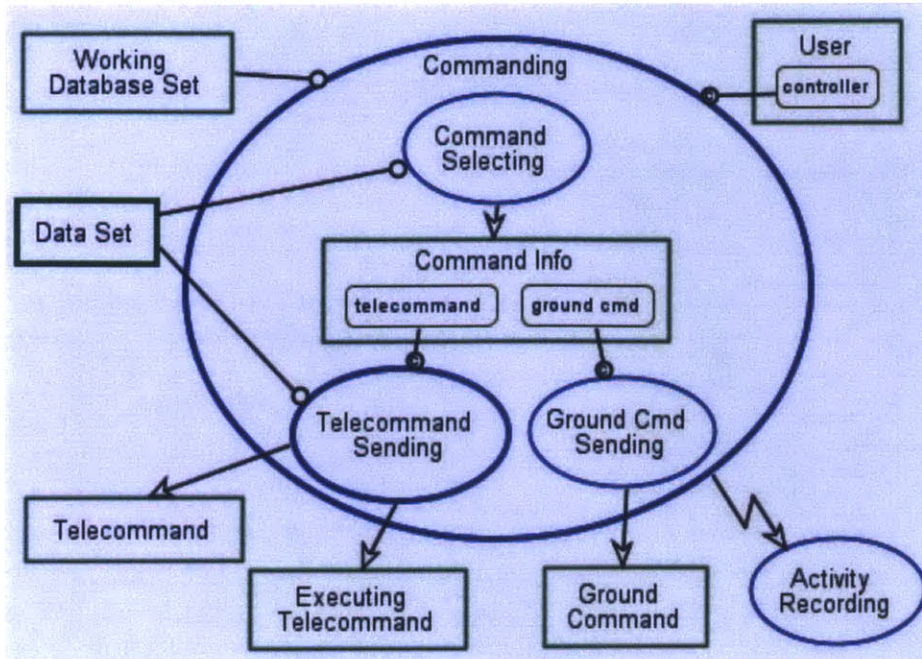


Figure 7. SD2.1 Commanding in-zoomed

OPL of SD2.1

User is controller by default.

Commanding occurs if User is controller.

Commanding requires Working Database Set.

Commanding invokes Activity Recording.

Commanding zooms into Command Selecting, Telecommand Sending, and Ground Cmd Sending, as well as Command Info.

Command Info can be ground cmd or telecommand.

Command Selecting requires Data Set.

Command Selecting yields Command Info.

Telecommand Sending occurs if Command Info is telecommand.

Telecommand Sending requires Data Set.

Telecommand Sending yields Telecommand and Executing Telecommand.

Ground Cmd Sending occurs if Command Info is ground cmd.

Ground Cmd Sending yields Ground Command.

The in-zoomed view of **Commanding** process in Figure 7 shows three subprocesses: **Command Selecting**, **Telecommand Sending**, and **Ground Cmd Sending**. **Command Selecting** uses information from **User**, **Working Database Set**, and **Data Set** to generate a **Command Info** object. If the **Command** is related to **Ground Equipment Set**, the **Ground Cmd Sending** process generates a **Ground Command** object. If the **Command Info** is a telecommand, the **Telecommand Sending** process generates a **Telecommand**

and an **Executing Telecommand** objects. Upon completion of the **Commanding** process, the **Activity Recording** process records the details of **Command** and **User** information.

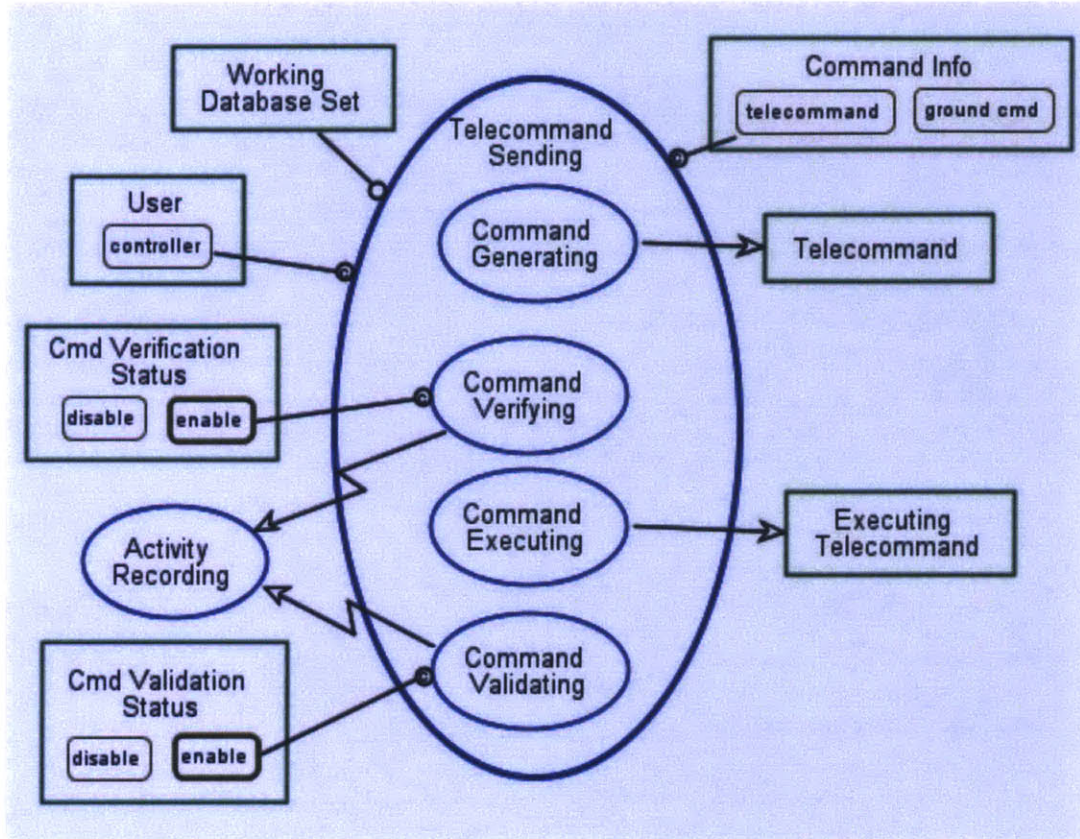


Figure 8. SD2.1.1 Telecommand Sending in-zoomed

OPL of SD2.1.1

User is controller by default.

Cmd Verification Status can be enable or disable.
enable is initial.

Cmd Validation Status can be enable or disable.
enable is initial.

Command Info can be ground cmd or telecommand.

Telecommand Sending occurs if User is controller and Command Info is telecommand.

Telecommand Sending requires Working Database Set.

Telecommand Sending zooms into Command Generating, Command Verifying, Command Executing, and Command Validating.

Command Generating yields Telecommand.

Command Verifying occurs if Cmd Verification Status is enable.

Command Verifying invokes Activity Recording.

Command Executing yields Executing Telecommand.
Command Validating occurs if Cmd Validation Status is enable.
Command Validating invokes Activity Recording.

Figure 8 shows several sequential subprocesses of the **Telecommand Sending** process. **Command Generating** generates a **Telecommand** object that is sent to **Satellite** via **Ground Equipment Set**. If **Cmd Verification Status** is **enable**, the **Command Verifying** process will be executed to verify that the command is correctly sent to **Satellite** by comparing the sent command with the **Data Set** collected from **Satellite** telemetry. If the **Cmd Verification Status** is **disable** or **Command Verifying** process is successfully executed, **Command Executing** process generates **Executing Telecommand** to execute the stored command on **Satellite**. Next, if **Cmd Validation Status** is **enable**, the **Command Validating** process is executed to check whether the command is successfully executed by comparing the expected effect defined by the related database in **Database Set** with the **Data Set** collected from **Satellite** telemetry. Upon the completion of the **Command Verifying** process or **Command Validating** process, the **Activity Recording** process is executed to keep record of the command verification or validation result.

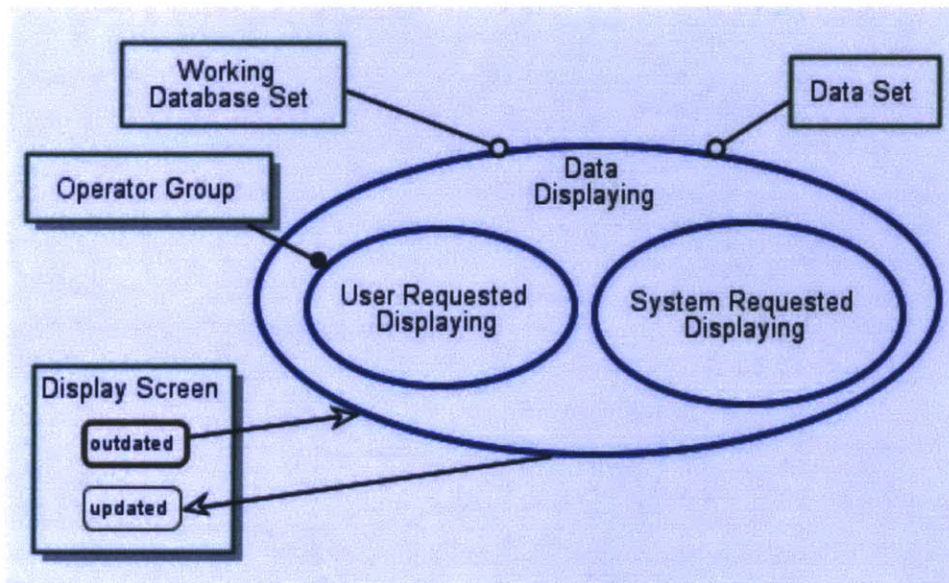


Figure 9. SD2.2 Data Displaying in-zoomed

OPL of SD2.2

Operator Group is physical.

Operator Group handles User Requested Displaying.

Display Screen is physical.

Display Screen can be outdated or updated.

outdated is initial.

Data Displaying requires Data Set and Working Database Set.
 Data Displaying changes Display Screen from outdated to updated.
 Data Displaying zooms into System Requested Displaying and User Requested Displaying.

The in-zoomed view of **Data Displaying** (Figure 9) consists of two independent subprocesses: **System Requested Displaying** and **User Requested Displaying**. **System Requested Displaying** gets data from **Database Set** and **Data Set**, and then formats and displays the data on **Display Screen**. The **User Requested Displaying** listens to the **Operator Group's** request and acquires the related data from **Database Set** and **Data Set**, and then formats and displays it on **Display Screen**.

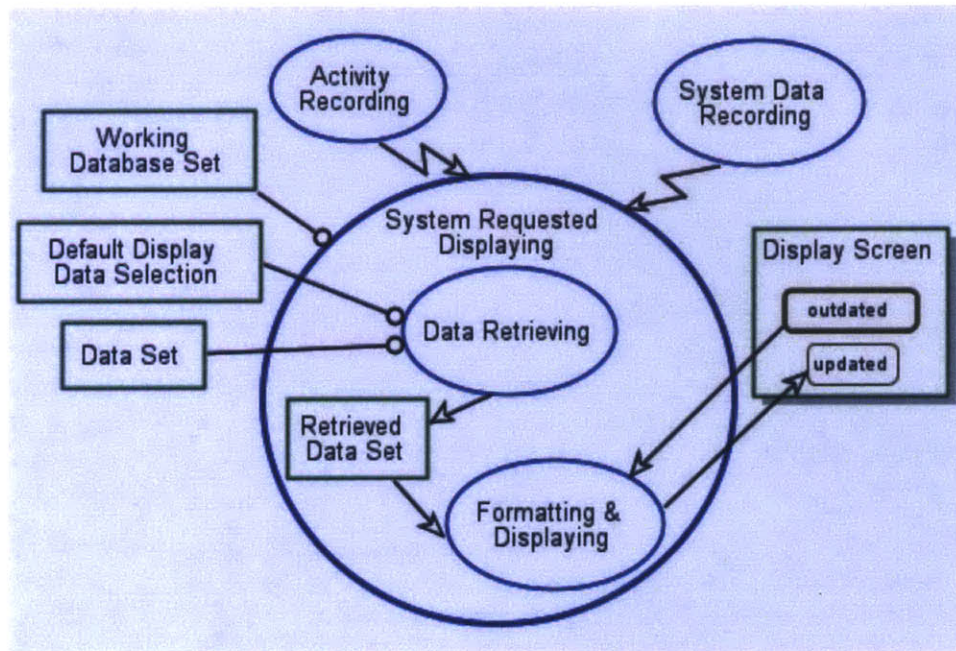


Figure 10. SD2.2.1 System Requested Displaying in-zoomed

OPL of SD2.2.1

Display Screen is physical.

Display Screen can be outdated or updated.

outdated is initial.

System Data Recording invokes System Requested Displaying.

Activity Recording invokes System Requested Displaying.

System Requested Displaying requires Working Database Set.

System Requested Displaying zooms into Data Retrieving and Formatting & Displaying, as well as Retrieved Data Set.

Data Retrieving requires Default Display Data Selection and Data Set.

Data Retrieving yields Retrieved Data Set.

Formatting & Displaying changes Display Screen from outdated to updated.
Formatting & Displaying consumes Retrieved Data Set.

Figure 10 illustrates the steps of the **System Requested Displaying** process. When activated by **Satellite Controlling** process, **Activity Recording** and **System Recording** for example, the **System Requested Displaying** process looks at the pre-defined **Default Display Data Selection**, retrieves the related data and information from **Database Set** and **Data Set**, and then formats and displays data on **Display Screen**.

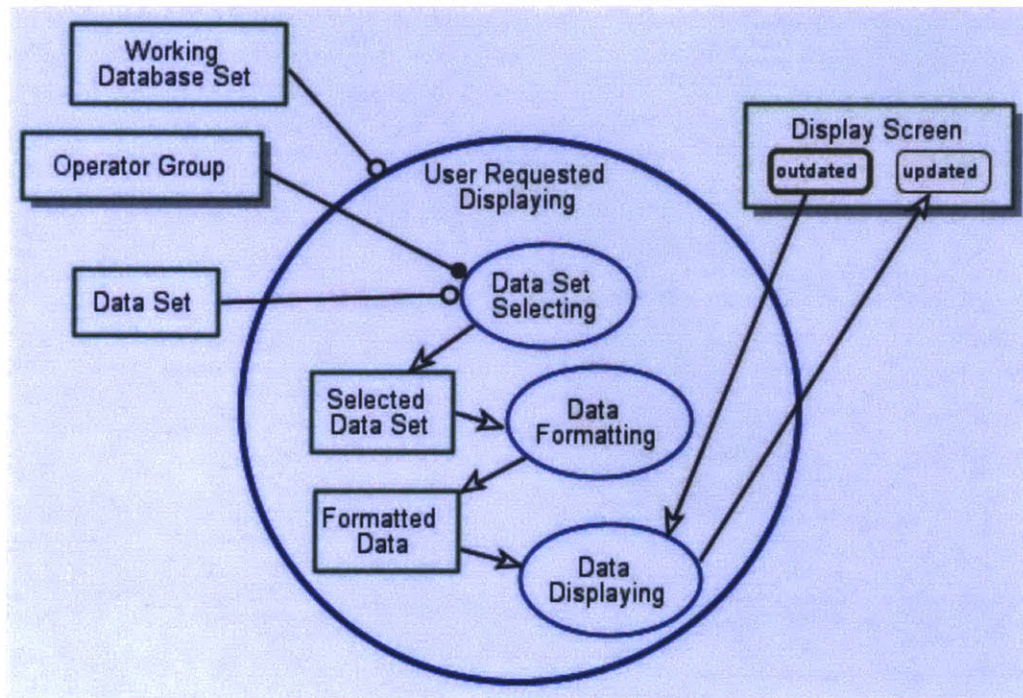


Figure 11. SD2.2.2 User Requested Displaying in-zoomed

OPL of SD2.2.2

Operator Group is physical.

Operator Group handles Data Set Selecting.

Display Screen is physical.

Display Screen can be outdated or updated.

outdated is initial.

User Requested Displaying requires Working Database Set.

User Requested Displaying zooms into Data Set Selecting, Data Formatting, and Data Displaying, as well as Formatted Data and Selected Data Set.

Data Set Selecting requires Data Set.

Data Set Selecting yields Selected Data Set.

Data Formatting consumes Selected Data Set.

Data Formatting yields Formatted Data.

Data Displaying changes Display Screen from outdated to updated.
 Data Displaying consumes Formatted Data.

The subprocesses of the **User Requested Displaying** process is shown in Figure 11. Per **Operator Group** request, the **Data Set Selecting** process allows **Operator Group** to select data set of his/her interest. **Data Set Selecting** acquires the related data from **Working Database Set** and **Data Set**, and creates the **Selected Data Set** object. Based on the **Selected Data Set** and information from **Database Set**, the data is formatted by **Data Formatting** process and displayed on **Display Screen** by **Data Displaying** process.

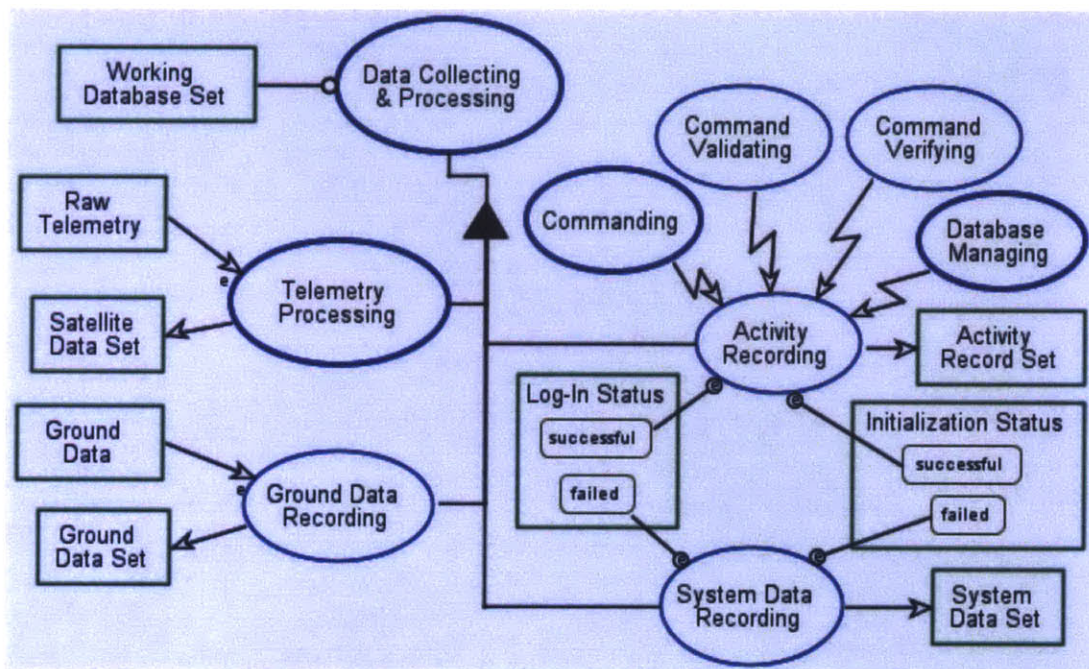


Figure 12. SD2.3 Data Collecting & Processing unfolded

OPL of SD2.3

Raw Telemetry triggers Telemetry Processing.

Ground Data triggers Ground Data Recording.

Initialization Status can be successful or failed.

Initialization Status triggers Activity Recording when it enters successful.

Initialization Status triggers System Data Recording when it enters failed.

Log-In Status can be successful or failed.

Log-In Status triggers Activity Recording when it enters successful.

Log-In Status triggers System Data Recording when it enters failed.

Database Managing invokes Activity Recording.

Commanding invokes Activity Recording.

Command Verifying invokes Activity Recording.

Command Validating invokes Activity Recording.

Data Collecting & Processing consists of Telemetry Processing, Ground Data Recording, System Data Recording, and Activity Recording.

Telemetry Processing consumes Raw Telemetry.

Telemetry Processing yields Satellite Data Set.

Ground Data Recording consumes Ground Data.

Ground Data Recording yields Ground Data Set.

System Data Recording requires failed Initialization Status and failed Log-In Status.

System Data Recording yields System Data Set.

Activity Recording requires successful Log-In Status and successful Initialization Status.

Activity Recording yields Activity Record Set.

Data Collecting & Processing requires Working Database Set.

Data Collecting & Processing process, consisting of four asynchronous subprocesses, is shown in Figure 12. Each subprocess is triggered by related events. For example, **Telemetry Processing** is triggered by the creation of **Raw Telemetry** object and a new **Ground Data** triggers **Ground Data Recording**. The outputs of these processes are specific types of **Data Set**, **Satellite Data Set** and **Ground Data Set** for example.

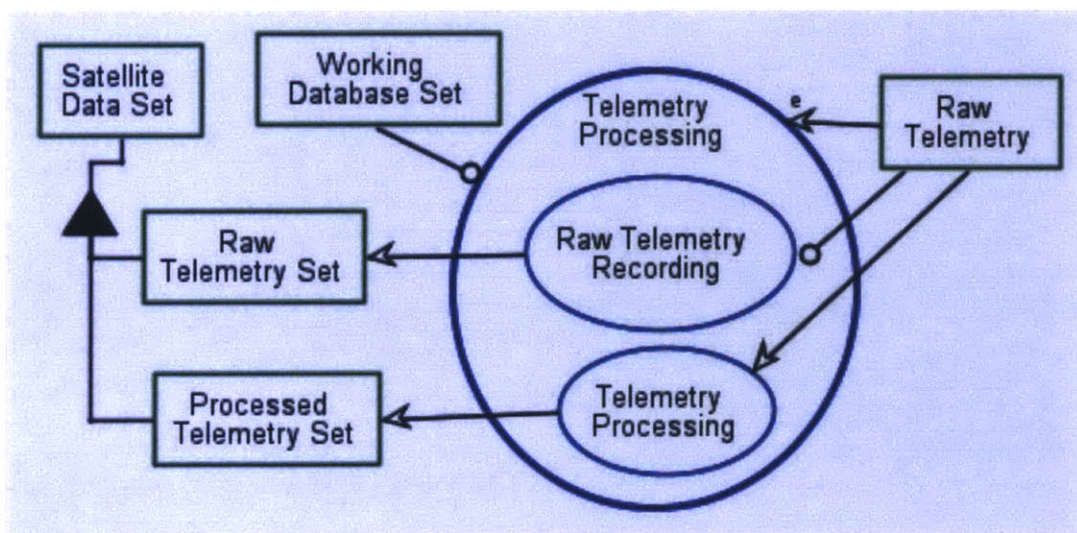


Figure 13. SD2.3.1 Telemetry Processing in-zoomed

OPL of SD2.3.1

Raw Telemetry triggers Telemetry Processing.

Satellite Data Set consists of Raw Telemetry Set and Processed Telemetry Set.

Telemetry Processing requires Working Database Set.

Telemetry Processing consumes Raw Telemetry.
Telemetry Processing zooms into Raw Telemetry Recording and Telemetry Processing.

Raw Telemetry Recording requires Raw Telemetry.
Raw Telemetry Recording yields Raw Telemetry Set.
Telemetry Processing consumes Raw Telemetry.
Telemetry Processing yields Processed Telemetry Set.

Telemetry Processing, depicted in Figure 13, has two sequential subprocesses. **Raw Telemetry Recording** uses data definition provided by **Working Database Set** to de-commutate and records **Raw Telemetry** in **Satellite Data Set**. Then, **Telemetry Processing** uses various definitions in **Working Database Set** such as parameter and function definitions to process the **Raw Telemetry** into the **Processed Telemetry Set**.

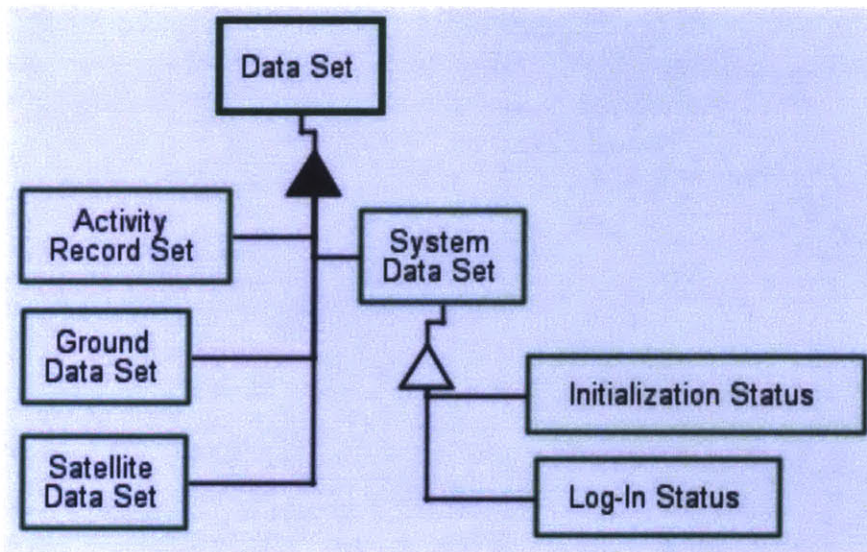


Figure 14. SD2.4 Data Set unfolded

OPL of SD2.4

Data Set consists of Activity Record Set, Satellite Data Set, Ground Data Set, and System Data Set.

Initialization Status is a System Data Set.

Log-In Status is a System Data Set.

Data Set, illustrated in Figure 14, represents all information that is recorded by **Satellite Controlling** process. **Data Set** can be categorized into several sets of data such as **Activity Record Set**, **Ground Data Set**, **Satellite Data Set**, and **System Data Set**.

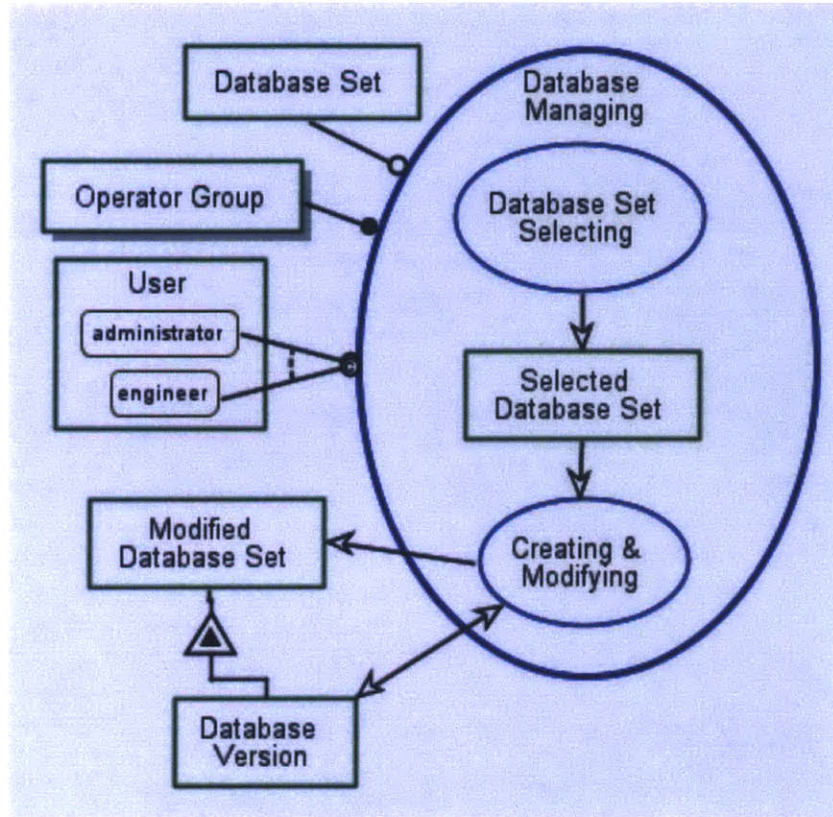


Figure 15. SD2.5 Database Managing in-zoomed

OPL of SD2.5

Operator Group is physical.

Operator Group handles Database Managing.

User can be engineer or administrator.

Modified Database Set exhibits Database Version.

Database Managing occurs if either User is administrator or User is engineer.

Database Managing requires Database Set.

Database Managing zooms into Database Set Selecting and Creating & Modifying, as well as Selected Database Set.

Database Set Selecting yields Selected Database Set.

Creating & Modifying affects Database Version.

Creating & Modifying consumes Selected Database Set.

Creating & Modifying yields Modified Database Set.

The simplified process of **Database Managing** is illustrated in Figure 15. When **Operator Group**, logged in as **administrator** or **engineer**, triggers the **Database Managing** process. **Database Set Selecting** allows **Operator Group** to select a **Database Set** to be modified or created, and based on input from **Operator Group** and information from **Database Set**, a **Selected Database Set** object is created. Working on the **Selected Database Set**, **Creating &**

Modifying process allows **Operator Group** to modify or create a new **Database Set**. The new or modified database set is generated together with its controlled version.

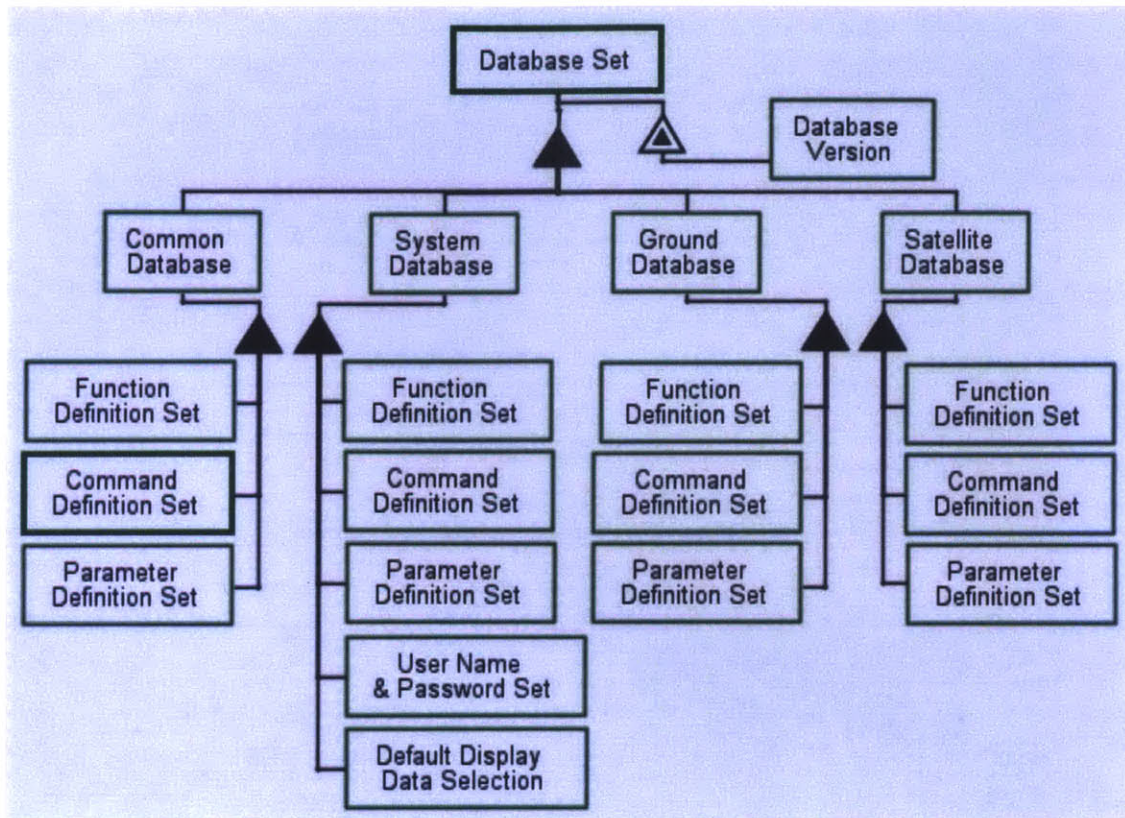


Figure 16. SD2.6 Database Set unfolded

OPL of SD2.6

Database Set exhibits Database Version.

Database Set consists of System Database, Ground Database, Satellite Database, and Common Database.

System Database consists of Parameter Definition Set, Function Definition Set, Command Definition Set, User Name & Password Set, and Default Display Data Selection.

Ground Database consists of Parameter Definition Set, Function Definition Set, and Command Definition Set.

Satellite Database consists of Parameter Definition Set, Command Definition Set, and Function Definition Set.

Common Database consists of Function Definition Set, Command Definition Set, and Parameter Definition Set.

Database Set is a set of definitions and it plays a key role in governing the processes, functions, and interaction of objects and processes in the system. It can be divided into several subsets as depicted in Figure 16. To avoid repetition of definitions, **Common Database** contains all definitions that are used by many subsets of **Database Set**. For specific definitions in one subset that are different from others, the dedicated definitions are defined separately. For example, **Function Definition Set**, **Command Definition Set**, and **Parameter Definition Set** that are specific to **Satellite** are kept in **Satellite Database**.

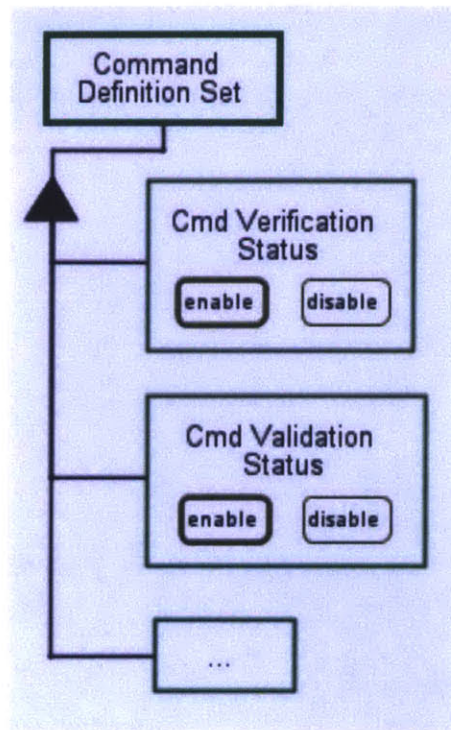


Figure 17. SD2.6.1 Command Definition Set unfolded

OPL of SD2.6.1

Command Definition Set consists of Cmd Verification Status, Cmd Validation Status, and

Cmd Verification Status can be enable or disable.

enable is initial.

Cmd Validation Status can be enable or disable.

enable is initial.

Command Definition Set of **Common Database** is shown in Figure 17. There are many common command definitions in this database set. **Cmd Verification**

Status and **Cmd Validation Status** are examples of common command parameter being used in the **Satellite Controlling** process.

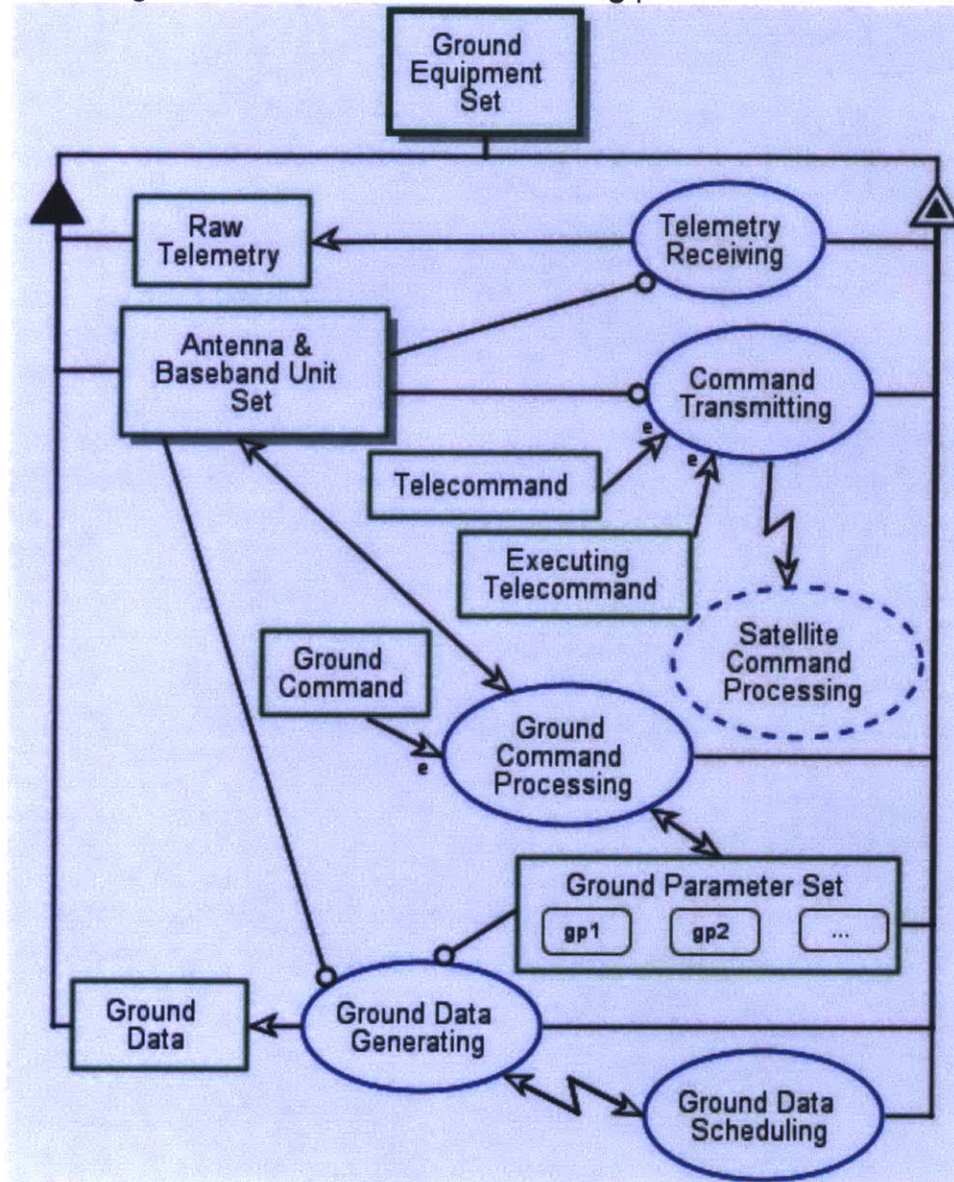


Figure 18. SD2.7 Ground Equipment Set unfolded

OPL of SD2.7

Ground Command triggers Ground Command Processing.

Telecommand triggers Command Transmitting.

Executing Telecommand triggers Command Transmitting.

Ground Equipment Set is physical.

Ground Equipment Set exhibits Ground Parameter Set, as well as Telemetry Receiving, Ground Command Processing, Command Transmitting, Ground Data Generating, and Ground Data Scheduling.

Ground Parameter Set can be gp1, gp2, or

Telemetry Receiving requires Antenna & Baseband Unit Set.
Telemetry Receiving yields Raw Telemetry.
Ground Command Processing affects Antenna & Baseband Unit Set and Ground Parameter Set.
Ground Command Processing consumes Ground Command.
Command Transmitting requires Antenna & Baseband Unit Set.
Command Transmitting consumes Executing Telecommand and Telecommand.
Command Transmitting invokes Satellite Command Processing.
Ground Data Generating requires Ground Parameter Set and Antenna & Baseband Unit Set.
Ground Data Generating yields Ground Data.
Ground Data Generating invokes Ground Data Scheduling.
Ground Data Scheduling invokes Ground Data Generating.
Ground Equipment Set consists of Raw Telemetry, Ground Data, and Antenna & Baseband Unit Set.
Antenna & Baseband Unit Set is physical.
Satellite Command Processing is environmental.

A simplified model of **Ground Equipment Set** is illustrated in Figure 18. **Ground Equipment Set** shall, at a minimum, perform the following functions: **Ground Commanding Processing**, **Ground Data Generating**, **Telemetry Receiving**, and **Command Transmitting**. The **Ground Equipment Set** consists of **Antenna & Baseband Unit Set**, which are the primary physical devices used to link and exchange the signals between **Satellite Controlling System** and **Satellite**. Status and parameters of **Ground Equipment** are changed and controlled by **Operator Group** via **Satellite Controlling** process and **Ground Command Processing** process. **Ground Data Generating** collects status and parameters information from **Ground Parameter Set** and **Antenna & Baseband Unit Set**, and creates **Ground Data** object that is recorded by **Ground Data Recording** process. Telemetry from **Satellite** is sent through **Antenna & Baseband Unit Set** and processed by **Telemetry Receiving** process.

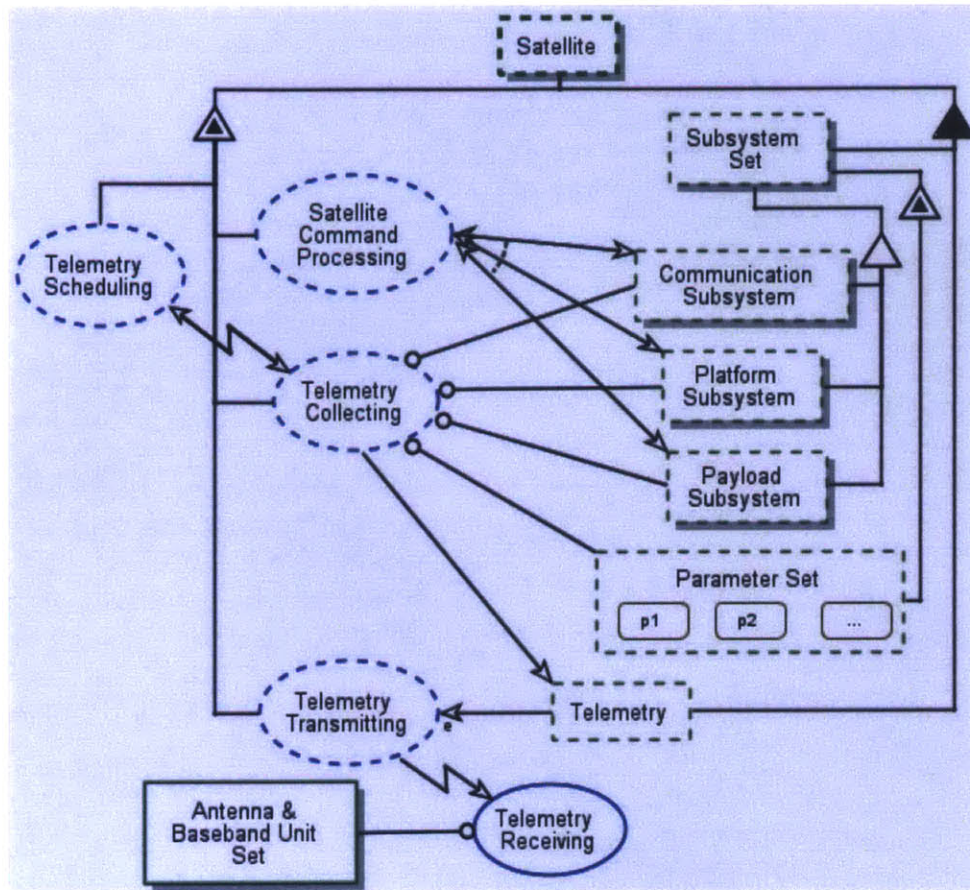


Figure 19. SD3 Satellite unfolded

OPL of SD3

Antenna & Baseband Unit Set is physical.

Satellite is environmental and physical.

Satellite exhibits Satellite Command Processing, Telemetry Collecting, Telemetry Transmitting, and Telemetry Scheduling.

Satellite Command Processing is environmental.

Satellite Command Processing affects Communication Subsystem either Platform Subsystem, or Payload Subsystem.

Telemetry Collecting is environmental.

Telemetry Collecting requires Parameter Set, Platform Subsystem, Communication Subsystem, and Payload Subsystem.

Telemetry Collecting yields Telemetry.

Telemetry Collecting invokes Telemetry Scheduling.

Telemetry Transmitting is environmental.

Telemetry Transmitting consumes Telemetry.

Telemetry Transmitting invokes Telemetry Receiving.

Telemetry Scheduling is environmental.

Telemetry Scheduling invokes Telemetry Collecting.
Satellite consists of Subsystem Set and Telemetry.
Subsystem Set is environmental and physical.
Subsystem Set exhibits Parameter Set.
Parameter Set is environmental.
Parameter Set can be p1, p2, or
Telemetry is environmental.
Telemetry triggers Telemetry Transmitting.
Platform Subsystem is environmental and physical.
Platform Subsystem is a Subsystem Set.
Payload Subsystem is environmental and physical.
Payload Subsystem is a Subsystem Set.
Communication Subsystem is environmental and physical.
Communication Subsystem is a Subsystem Set.
Telemetry Receiving requires Antenna & Baseband Unit Set.

To demonstrate the interaction between **Satellite Controlling System** and **Satellite**, a simplified model of a satellite is presented in Figure 19, showing three main processes. **Satellite Command Processing** receives the **Telecommand** and **Executing Telecommand** from **Satellite Controlling System** with the help of **Antenna & Baseband Unit Set**. Depending on the information in the command, **Satellite Command Processing** distributes **Telecommand** and **Executing Telecommand** to the specific unit in **Subsystem Set**. **Telemetry Collecting** regularly collects data from each **Satellite's** subsystems and **Satellite's Parameter Set**, and then generates **Telemetry** object that is transmitted to **Satellite Controlling System** by **Telemetry Transmitting** process.

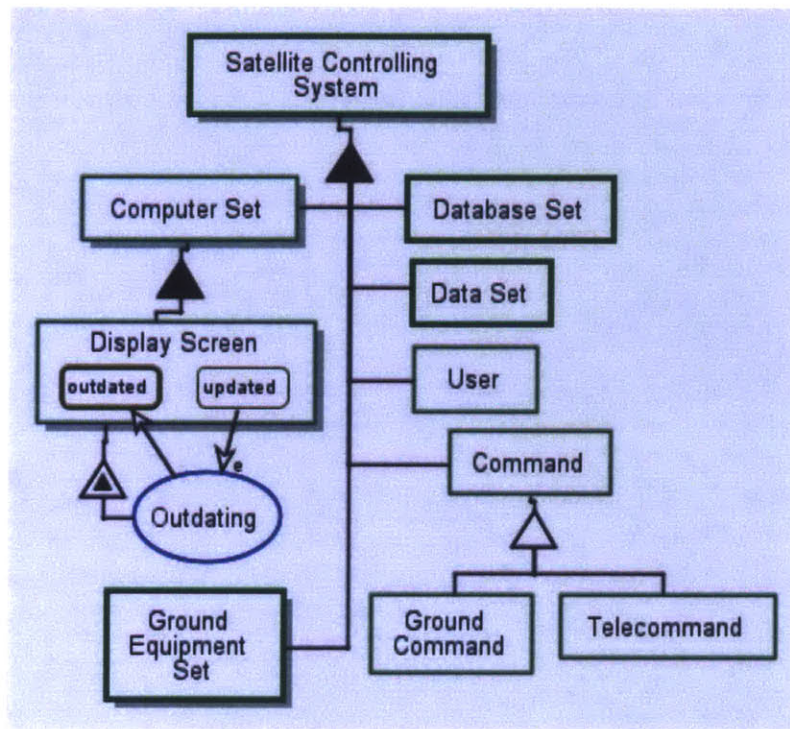


Figure 20. SD4 Satellite Controlling System unfolded

OPL of SD4

Satellite Controlling System is physical.

Satellite Controlling System consists of Database Set, Data Set, Computer Set, Ground Equipment Set, Command, and User.

Computer Set is physical.

Computer Set consists of Display Screen.

Display Screen is physical.

Display Screen can be outdated or updated.

outdated is initial.

Display Screen exhibits Outdating.

Outdating changes Display Screen from updated to outdated.

Display Screen triggers Outdating when it enters updated.

Ground Equipment Set is physical.

Ground Command is a Command.

Telecommand is a Command.

Figure 20 shows the structure of **Satellite Controlling System**. This is not an exhaustive set of the required objects, but it represents the objects that are needed to perform and simulate the simplified model of the system.

Chapter 4. Guidelines for Satellite Ground Control Software

Independently of the complexity of the satellite and the number of subsystems and base functions that the satellite has, the OPM model comprises only a few main top-level functions in satellite ground control system. These base functions are the main subprocesses that include **Data Collecting & Processing**, **Commanding**, **Data Displaying**, and **Database Managing**. Thus, the guidelines specified in this chapter as requirements are related primarily to these base functions.

As the focus of this research is not on the security of the software, the standard requirements of software security are assumed to be incorporated into the general requirement and will not be stated in the guidelines below. Each guideline is followed by a Motivation clause, which links it to the OPM model and how this model relates to the rationale behind that specific guideline.

Proposed guidelines and their motivation

1. Development tools and programing languages

- 1.1. The number of development tools and programing languages shall be kept to a minimum to avoid unnecessary development and maintenance complexity.

Motivation: The software developers, who integrated several legacy programs that are coded with different programing languages into their software, found great difficulty in their development, system integration, and maintenance phases.

- 1.2. The selected tools and programing languages shall be platform- and operating-system-independent.

Motivation: This requirement increases the flexibility of the software when there is a need to change or upgrade hardware, and it opens opportunities to work on mobile computers such as tablet and smartphone.

- 1.3. Coding changes due to any operating system upgrade will be zero or negligible.

Motivation: Due to the long lifetime of satellite mission comparing with the operating system lifecycle, there will be at least one operating system upgrade within the satellite mission lifetime. Developer needs to plan at the design and development phase to accommodate the foreseeable operating system update.

- 1.4. Virtual machine is an acceptable solution to overcoming the problem of operating system change or upgrade.

Motivation: Virtual machine is a widely used solution to the change in operating system. However, developers are encouraged to explore and propose more suitable and efficient solutions.

2. Software Structure and Programming Style

- 2.1. Concepts of object-oriented programming, such as encapsulation, abstraction, inheritance, and code reusability shall be used.

Motivation: The object-oriented programming concepts are efficient and flexible. If the software is well developed following these concepts, it will readily accommodate the future changes in processes and customizable functions

- 2.2. Modularity structure and database-driven application concepts shall be implemented.

Motivation: Developing modular software and database-driven application concepts enhances flexibility and visibility in software structure and functions. For example, developers and users can modify existing features or create new ones by modifying the database without any change in software code.

- 2.3. The high-level functions and features shall be developed based on the base functions, and the functionalities will be defined by parameters, configurations, and definitions in the sets of the pre-defined and user-defined database.

Motivation: As the OPD in Figure 6 shows, the main system function is modeled as being comprised of five main base functions: **Satellite Controlling, Data Displaying, Data Collecting & Processing, Commanding and Database Managing**. The parameters as shown in the OPM model are **Function Definition Set, Command Definition Set** and **Parameter Definition Set**. The main processes are linked to the database in the OPM model, as shown in Figure 3, Figure 6, Figure 7, Figure 9, Figure 12, Figure 15, and Figure 16.

- 2.4. All hardware-dependent functions shall be embedded in their designated modules, and their interface functions and definitions shall be carefully coded and documented in full detail.

Motivation: As shown in Figure 10, the **Formatting & Displaying** process is defined as one separated module dedicated to communication with the **Display Screen**. The documentation of interface functions and definitions of these hardware-dependent functions is very important for future software upgrades, software errors troubleshooting, and system maintenance.

- 2.5. Any hardware change shall affect only the changed hardware related modules, and the required re-coding shall not propagate and be invisible to other modules and main processes.

Motivation: By applying modularity and database-driven concepts to the hardware-dependent functions, the re-coding required by hardware change shall affect only the hardware related modules. This requirement helps prevent coding errors or bugs propagating from one module to the others, and it is likely that the system verification and validation of each module will be easy as developer can focus on only the modules related to the changed hardware.

3. Data Collecting & Processing

- 3.1. System and activities status, including system **Initialization Status**, **Log-In Status**, and all **User** activities and requests, shall be recorded in detail.

Motivation: As specified in the OPM model, most processes generate triggering messages that are recorded by **System Data Recording** or **Activity Recording** processes. For example, in Figure 4 the **Initializing** process generates **Initialization Status** object that triggers **Activity Recording** to record its **successful** status or triggers **System Data Recording** to record its **failed** to initialize status. The same practice is applied to the **Logging In** process (and in other processes) in Figure 5 that the **Logging In** process creates **Log-In Status** object that triggers **Activity Recording** to record its **successful** status or triggers **System Data Recording** to record its **failed** to log-in status.

- 3.2. The **Data Collecting & Processing** process shall allow **User** to trace and analyze process errors and provide clear, filterable, and searchable information of process issues and all messages related to system and **User** activities.

Motivation: Not only well-documented-and-recorded data are important, searching, filtering and formatting these document and data records are very helpful for users to quickly understand and get the information and evidence they are looking for.

- 3.3. The **Data Collecting & Processing** process shall support event analysis and event report.

Motivation: As shown in Figure 10 and Figure 11, the **Selected Data Set** or the **Default Display Data Selection**, the pre-defined set of data and formatting in the **Database Set**, will readily help **User** to perform event analysis and produce event report.

- 3.4. The name of record file(s) and file(s) location shall be configurable by **User**.

Motivation: To enable users define their own file names and locations, the software developer has to design the system to allow users to configure their preferences not by changing software code but by setting the **Parameter Definition Set** in the **Database Set**.

- 3.5. All data shall be processed and all necessary statistics, such as minimum, maximum, and average values, shall be generated and easily retrievable.

Motivation: The OPD in Figure 13 emphasizes that both **Raw Telemetry Set** and **Processed Telemetry Set** are recorded. Thus, there is no need to reprocess the **Raw Telemetry** every time it is requested to display its processed value.

- 3.6. The parameters of statistical information and data processing, such as number of sample and period of statistical calculation, shall be configurable.

Motivation: Almost all the times users eventually find out that they need to change predefined parameters, such as the number of samples or time

developer has to design the software so that allows users to change these parameters via the **Parameter Definition Set** in the **Database Set**.

4. Commanding

- 4.1. The **Commanding** process shall facilitate **User's** command selection by providing filterable command list.

Motivation: The OPD in Figure 7 shows that the **Command Selecting** process uses both information from **Working Database Set** and **Data Set** to facilitate command selection by the **User**.

- 4.2. The **Commanding** process shall handle all kind of commands related to satellite and ground equipment.

Motivation: As illustrated in Figure 7, the **Commanding** process can process both **Ground Command** and **Telecommand**.

- 4.3. Different command types and structures shall be defined in the **Database Set** and not by the different subprocesses.

Motivation: As shown in Figure 16 and Figure 17, the command types and structures are defined by the **Command Definition Set**, the **Parameter Definition Set**, the **Function Definition Set** and other definitions in the **Database Set**.

- 4.4. The **Command Generating**, the **Command Transmitting**, the **Command Verifying**, the **Command Executing**, and the **Command Validating** processes shall be defined by the **Command Definition Set**, the **Parameter Definition Set**, the **Function Definition Set**, and interface control documents or user manuals provided by satellite and hardware manufacturers.

Motivation: The design of the **Command Generating**, the **Command Transmitting**, the **Command Verifying**, the **Command Executing**, and the **Command Validating** processes are based on the **Command Definition Set**, the **Parameter Definition Set**, the **Function Definition Set**, and other definitions in the **Database Set** that are derived from various documents, mainly interface control documents and user manuals provided by satellite and hardware manufacturers.

- 4.5. The steps of the **Commanding** process shall be configurable by changing parameters in the **Command Database**.

Motivation: This requirement provides users with flexibility to operate their **Satellite** in different scenarios. For example, in emergency situations, users want to send command to the **Satellite** at the fastest rate; thus, they set the status of both **Cmd Verification Status** and **Cmd Validation Status** to **disable** in the **Command Definition Set** (in Figure 17). As shown in the OPD in Figure 8, by this setting, the **Telecommand Sending** process is completed quickly without waiting for **Command Verifying** and **Command Validating**.

- 4.6. The software system shall allow **User** to introduce new commands or change existing commands by modifying or creating the **Database Set** without any change in the process code.

Motivation: This requirement provides users with flexibility to incorporate future need for new commands, which can be fulfilled by creating and modifying the **Database Set** without any change in the software code.

5. Data Displaying

- 5.1. The **Data Displaying** process shall handle all kinds data related to the system – both the satellite, and the ground equipment.

Motivation: As the OPD in Figure 9 shows, the **Data Displaying** process can display the data from the **Data Set**, which, as shown in Figure 14, contains data items related to system (**System Data Set** and **Activity Record Set**), satellite (**Satellite Data Set**), and ground equipment (**Ground Data Set**).

- 5.2. The **Data Formatting** of the **Data Set** from system, satellite, and ground equipment shall be governed by the **Database Set** and not by subprocesses.

Motivation: The OPD in Figure 11 shows that the **User Requested Displaying** process uses the **Working Database Set** to determine the format of the selected **Data Set** to be displayed.

- 5.3. The software system shall allow **User** to change or create a new display format of any **Data Set** by modifying or creating the **Database Set** without any change in the process code.

Motivation: The display format of different types (**System Data Set**, **Activity Record Set**, **Satellite Data Set**, and **Ground Data Set** in Figure 14) of the **Data Set** is defined by the definition set, such as **Function Definition Set** and **Parameter Definition Set**, in the **Database Set**, shown in Figure 16.

6. Database Managing

- 6.1. The **Database Managing** process shall provide the authorized **User** with easy access to the database set and facilitate creating new database set or changing the existing database set.

Motivation: As illustrated in Figure 15, only **User** who is **administrator** or **engineer** is allowed to modify the **Database Set**.

- 6.2. The **Database Set** shall have **Database Version** for tracking and recording the creation and modification of the **Database Set**.

Motivation: The OPD in Figure 15 shows that the **Creating & Modifying** creates the **Modified Database Set** and updates its **Database Version** for tracking and reference purpose.

- 6.3. A database translation process (not modeled) shall be provided to facilitate the translation of manufacturers' database formats into the software database format.

Motivation: While rare, there are a few times during the satellite mission lifetime that the satellite manufacturer updates and sends out the updated version of the satellite database to the satellite owner. The satellite owner and operator can then use the tool for translating the old database to the new database format used by the new software, or vice versa. Asking the

software developer to translate the updated database during the maintenance period has proved to be difficult and expensive.

Chapter 5. Conclusion and Discussion

This research provides guidelines for developing satellite ground control software that are grounded in a relatively detailed OPM conceptual model of the system-to-be. The underlying concepts of this model and guidelines include database-driven application, object-oriented, and modularity concepts. Instead of coding each different function differently, only the common base functions shall be coded, and combining some of these basis functions will form specific high-level required functions. The formation and combination of these functions will be governed by the main code and a database. These concepts will provide the satellite operator with the flexibility to create new features, and help software developers to find and fix bugs. The OPM model along with the suggested guidelines are expected to help alleviate the problems identified in the various stages of satellite ground control software lifecycle.

The concepts of database driven application, object-oriented programming and modularity have potential significant benefits, but these bear some costs, as discussed next.

A well-designed database driven and modular application will make it easier to manage requirements change or the needs for new features during and after the development of the satellite ground control software. This architecture also makes it easier and quicker to find and fix bugs.

On the flip side, a considerable upfront investment of design effort is required to design and code all the base functions that govern the formation and combination of high-level functions.

Future Research

Further studies related to the topic of this research include the following.

1. Creating a more elaborate and executable model to demonstrate and verify the effectiveness of the proposed concept and system architecture.
2. Using Physical-Informational Essence Duality (PIED) concept with OPM modeling to help analyze and understand satellite anomalies.
3. Developing tool to generate system requirement directly from existing OPM models.

Bibliography

- [1] W. C. Straka, "The role of COTS software in satellite ground control systems," in , *IEEE Aerospace Conference, 1997. Proceedings*, 1997, vol. 4, pp. 149–158 vol.4.
- [2] W. Anderson and J. McAuley, "Commercial off-the-shelf product management lessons learned - satellite ground control system (SGCS) upgrade," in *Fifth International Conference on Commercial-off-the-Shelf (COTS)-Based Software Systems, 2006*, 2006, p. 8 pp.–.
- [3] D. Dori, *Object-Process Methodology -- A Holistic System Paradigm*. Springer Verlag, Berlin, Heidelberg, New York, 2002.
- [4] T. Vardanega, "An evolutionary approach to the construction of new-generation software-intensive satellite control systems," in *Proceedings of the Joint Workshop on Parallel and Distributed Real-Time Systems, 1997*, 1997, pp. 263–268.
- [5] J. W. Cutler and C. A. Kitts, "Mercury: a satellite ground station control system," in *1999 IEEE Aerospace Conference, 1999. Proceedings*, 1999, vol. 2, pp. 51–58 vol.2.
- [6] N. Groleau, L. Kiser, F. Girouard, A. Hopkins, T. Morgan, S. Chakrabarti, T. Cook, D. Cotton, and P. Dell, "A Fully Implemented Semi-Automated Ground-Control System for the TERRIERS Satellite," *AIAAUSU Conf. Small Satell.*, Sep. 1996.
- [7] I. J. Kim, W. C. Jung, M. Kim, and J.-H. Kim, "Design and prototyping of high availability architecture for satellite ground control system," in *The 7th International Conference on Advanced Communication Technology, 2005, ICACT 2005*, 2005, vol. 2, pp. 1044–1048.
- [8] "CMMI Institute - the home of Capability Maturity Model Integration," *CMMI Institute*. [Online]. Available: <http://cmmiinstitute.com/>. [Accessed: 23-Mar-2014].
- [9] "Data-driven programming," *Wikipedia, the free encyclopedia*. 16-May-2014.
- [10] M. Riaz, "Maintainability prediction of relational database-driven applications: A systematic review," pp. 263–272, 2012.