**CHAPTER 1**

**INTRODUCTION**

**1.1 INTRODUCTION TO REAL TIME SATELLITE HEALTH MONITORING**

In the context of spaceflight, a satellite is an object which has been placed into orbit by human endeavour. Satellites are used for a large number of purposes. Common types include military and civilian Earth-observation satellites, communications satellites, navigation satellites, weather-monitoring satellites and scientific research satellites. Satellite orbits vary greatly, depending on the purpose of the satellite, and are classified in a number of ways. Eg: Low-Earth orbit, Polar orbit, geostationary orbit etc.

A satellite in orbit is an important asset to the country and the space agency that constructed it and maintains it. A satellite is placed in orbit for various scientific and technological applications like mobile communications, Direct-to-Home services, meteorological observations, telemedicine, tele-education, disaster warning, radio networking, search and rescue operations, remote sensing and scientific studies of the space.

Satellites are usually semi-independent computer-controlled systems. Satellite sub-systems attend to many tasks, such as power generation, thermal control, telemetry, attitude control and orbit control.

A satellite consists of various parameters that are meant to be under constant observation. Some of them are listed below:

* Total Spacecraft Load
* Wheel Speed
* Current
* Pressure
* Acceleration
* Velocity
* Temperature

**1.2 INTRODUCTION TO TELEMETRY AND ITS SIGNIFICANCE**

Telemetry is defined as remote measurement and reporting of information. The word is derived from Greek roots “tele” means remote, and “metron” means measure.

Once the satellite is launched and put into an orbit, monitoring the health condition of the satellite becomes a crucial factor in a space mission. The data that represents the health condition of the satellite is called telemetry data. The satellite transmits the telemetry data to a Ground station. This is known as satellite downlink. The analog data received by the ground station is decompressed, decrypted and converted into digital format. Further the binary data is processed by applying engineering unit conversion scheme specific to a sub-system telemetry parameter and the processed value is generated in real-time.

The processed telemetry data is reviewed by the telemetry analyst. If any anomaly is found or error has occurred on-board then necessary actions are implemented to correct the error. The analyst also studies the variations in one or more parameter values and constructs a behavior model of the instrument on-board.

Traditionally, the diagnosis of satellite anomalies during test and flight is slow and not very thorough due to a limited view of telemetry. The process is an off-line, linear analysis of guessing a root cause and then attempting to verify the guess through identifying and plotting telemetry parameter values over the appropriate time period. The desire of mission operators and testers is to have a capability to analyze all telemetry parameter values simultaneously and then be able to freely move through time to cover the entire problem space.

Telemetry visualization and interpretation places a significant role in any space mission. Its role is explained as follows:

A telemetry parameter value may represent the current working status of a satellite on board E.g.: Camera Switched on Status. Thus monitoring the parameter in real-time will help the satellite engineer to operate the instrument and get desired result.

A telemetry parameter value may represent the current health status of an instrument on-board E.g.: Fuel Tank Temperature. Thus monitoring the parameter in real-time so that it does not exceed a certain threshold will help prevent physical damage to the on-board instrument.

Visualizing a processed telemetry parameter value in real-time in different formats such as text and graph helps the satellite engineer or analyst to get a clear picture about the current status of the satellite on-board. She/he can operate the satellite and implement necessary actions based on this understanding.

* 1. **LITERATURE SURVEY**

A literature survey is a body of text that aims to review the critical points of current knowledge including substantive findings as well as theoretical and methodological contributions to a particular topic. Its ultimate goal is to bring the reader up-to-date with current literature on a topic and forms the basis for another goal, such as future research that may be needed in the area.

Literature survey on satellite telemetry visualization

Satellite health monitoring has been implemented in various fashions in the past. Initially a satellite ground control system that monitors and controls a satellite 24 hours a day was achieved through distributed architecture that assures high-availability. A telemetry visualization component was developed that achieves visualization through various methods such as flexible data access and presentation . An architecture for wireless tele-command (An instruction transmitted from the ground station to satellite to perform a desired function or operation) and telemetry was constructed using various wireless technologies such as ZigBee and BlueTooth.For real-time monitoring of testing and flight operations of a satellite, a sub-system engineer typically views a screen that displays real-time values of telemetry parameters of interest. An interactive visualization system provided the various telemetry display options such as grid view and time-history view of selected telemetry parameters .

China’s first lunar satellite Chang’E-1 (CE-1) was launched on October 24th 2007.During its flight to the Moon, a series of maneuvers was performed to keep the satellite in the correct trajectory. Among those maneuvers, the one performed for Lunar Orbit Insertion (LOI) was the most important since it directly decided whether the satellite could enter the mission orbit or not. Therefore, real-time monitoring and estimation of the satellite health status during this stage became crucial. A system was setup to accomplish this task using as few data as possible .

A system was developed for the purpose of monitoring of satellite activities in real-time by using the three-dimensional visualization tool. Visualizing a satellite activity based on telemetry parameter values provides an intuitive understanding of the spacecraft’s activities that far exceeds that available through textual display and graphs. In addition to three-dimensional visualization capabilities, standard graphing and plotting tools allow review of telemetry histories .The telemetry data of the spacecraft system contains hundreds or thousands of variables. A system was constructed to represent the principal satellite sub-system behaviour by a much smaller number of latent variables. The system detects an unexpected pattern in the telemetry data by reducing the dimensionality, recovering the original dimensions, comparing the recovered and original data .

**CHAPTER 2**

**PROPOSED SYSTEM**

**2.1 PROBLEM DEFINITION**

A satellite in orbit is an important asset to the country and the space agency that constructed, launched, maintains and operates it in orbit. The satellite starts transmitting the health status of the different sub-systems on-board through telemetry data to the ground station once it is placed it in orbit.It is crucial to monitor this telemetry data in real-time and operate the instruments on-board accordingly or take action to correct a telemetry parameter value immediately so as to prevent damage to any sub-system of the satellite. Visualizing the telemetry data in real-time different formats such as text, 3D and graphs will help in interpreting the current status of the satellite and resulting action (if required) will be quicker .Integrating the real-time textual display, real-time graph display and real-time 3D display into a single window will help create a better understanding of the current health status of the sub-system being monitored along with the other sub-systems on-board.

**2.2 FEATURES OF PROPOSED SYSTEM**

Feature-1 The software shall take time-wise samples of processed telemetry parameters as input data and deliver it to a visualization unit in real-time

Feature-2 The software shall have the provision of delivering consecutive samples of each parameter value at every execution for a given parameter.

Feature-3 The software shall have the provision of fetching samples of processed values of a particular telemetry parameter for a given time period.

Feature-4 The software shall have the provision of displaying an error message if any anomaly occurs.

Feature-5 The software shall have the provision of displaying the processed telemetry data in the form of a graph (versus time) in real-time between certain threshold values specific to each telemetry parameter. These thresholds are defined by default.

Feature-6 The software shall have the provision of displaying the processed values of a given telemetry parameter obtained for a desired time period in the form of a graph (versus time period).

Feature-7 The software shall have the provision of altering the threshold values of a telemetry parameter at the user’s discretion. The lower and higher threshold values are taken as input and the real-time graph display is updated with the new threshold values in real-time.

Feature-8 The software shall have the provision of displaying a notification in real-time every time the telemetry parameter value exceeds its threshold. Such a situation must be recognized by the software and an alert must be displayed in real-time independent of the telemetry parameter currently being monitored.

Feature-9 The software shall be portable. It should be executable on any machine meeting the requirements independent of the platform hosted on the machine.

**CHAPTER 3**

**SYSTEM REQUIREMENT SPECIFICATION**

**3.1 SOFTWARE REQUIREMENTS :**

The Software System Requirements can be specified as follows :

* Windows/Linux/Unix operating system
* JDK (Version 1.6.25)
* JFreeChart Java Library (Version 1.0.13)
* JCommon Java Library (Version 1.0.16)
* Java 3D API (Version 1.5.1)

**3.2 HARDWARE REQUIREMENTS :**

The hardware requirements are specified as follows:

* 2GHz 32-Bit Processor
* 1GB RAM
* 300GB Hard Disk
* built-in graphics card (512 MB memory capacity)
* USB port, NIC port
* 18/24 inch monitor
* keyboard and mouse

**3.3 TECHNOLOGY USED:**

**JAVA**

Java is a [programming language](http://en.wikipedia.org/wiki/Programming_language) originally developed by [James Gosling](http://en.wikipedia.org/wiki/James_Gosling) at [Sun Microsystems](http://en.wikipedia.org/wiki/Sun_Microsystems) (which has since merged into [Oracle Corporation](http://en.wikipedia.org/wiki/Oracle_Corporation)) and released in 1995 as a core component of Sun Microsystems' [Java platform](http://en.wikipedia.org/wiki/Java_(software_platform)). The language derives much of its [syntax](http://en.wikipedia.org/wiki/Syntax_(programming_languages)) from [C](http://en.wikipedia.org/wiki/C_(programming_language)) and [C++](http://en.wikipedia.org/wiki/C%2B%2B) but has a simpler [object model](http://en.wikipedia.org/wiki/Object_model) and fewer [low-level](http://en.wikipedia.org/wiki/Low-level_programming_language) facilities. Java applications are typically [compiled](http://en.wikipedia.org/wiki/Compiler) to [byte code](http://en.wikipedia.org/wiki/Java_bytecode) ([class file](http://en.wikipedia.org/wiki/Class_(file_format))) that can run on any [Java Virtual Machine](http://en.wikipedia.org/wiki/Java_Virtual_Machine) (JVM) regardless of [computer architecture](http://en.wikipedia.org/wiki/Computer_architecture). Java is a general-purpose, concurrent, class-based, object-oriented language that is specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that code that runs on one platform does not need to be recompiled to run on another. Java is currently one of the most popular programming languages in use, particularly for client-server web applications, with a reported 10 million users.

There were five primary goals in the creation of the Java language:

* It should be "simple, object-oriented and familiar"
* It should be "robust and secure"
* It should be "architecture-neutral and portable"
* It should execute with "high performance"
* It should be "interpreted, threaded, and dynamic"

One characteristic of Java is portability, which means that computer programs written in the Java language must run similarly on any hardware/operating-system platform. This is achieved by compiling the Java language code to an intermediate representation called [Java byte code](http://en.wikipedia.org/wiki/Java_bytecode), instead of directly to platform-specific [machine code](http://en.wikipedia.org/wiki/Machine_code). Java byte code instructions are analogous to machine code, but are intended to be [interpreted](http://en.wikipedia.org/wiki/Interpreter_(computing)) by a [virtual machine](http://en.wikipedia.org/wiki/Virtual_machine) (VM) written specifically for the host hardware. [End-users](http://en.wikipedia.org/wiki/End-user) commonly use a [Java Runtime Environment](http://en.wikipedia.org/wiki/Java_Virtual_Machine) (JRE) installed on their own machine for standalone Java applications, or in a Web browser for Java [applets](http://en.wikipedia.org/wiki/Applet).

Standardized libraries provide a generic way to access host-specific features such as graphics, [threading](http://en.wikipedia.org/wiki/Thread_(computer_science)), and [networking](http://en.wikipedia.org/wiki/Computer_network).

A major benefit of using byte code is porting. However, the overhead of interpretation means that interpreted programs almost always run more slowly than programs compiled to native executables would. Just-in-Time (JIT) compilers were introduced from an early stage that compile byte codes to machine code during runtime.

Programs written in Java have a reputation for being slower and requiring more memory than those written in C.However, Java programs' execution speed improved significantly with the introduction of [Just-in-time compilation](http://en.wikipedia.org/wiki/Just-in-time_compilation) in 1997/1998 for [Java 1.1](http://en.wikipedia.org/wiki/Java_version_history), the addition of language features supporting better code analysis (such as inner classes, String Buffer class, optional assertions, etc.), and optimizations in the [Java Virtual Machine](http://en.wikipedia.org/wiki/Java_Virtual_Machine) itself, such as [Hotspot](http://en.wikipedia.org/wiki/HotSpot) becoming the default for Sun's JVM in 2000. Currently (February 2012), micro benchmarks show Java 7 is approximately 1.5 times slower than C.

Some platforms offer direct hardware support for Java; there are microcontrollers that can run Java in hardware instead of a software [Java Virtual Machine](http://en.wikipedia.org/wiki/Java_Virtual_Machine), and [ARM](http://en.wikipedia.org/wiki/ARM) based processors can have hardware support for executing Java byte code through its [Jazelle](http://en.wikipedia.org/wiki/Jazelle) option.

Automatic memory management

Java uses an [automatic garbage collector](http://en.wikipedia.org/wiki/Garbage_collection_(computer_science)) to manage memory in the [object lifecycle](http://en.wikipedia.org/wiki/Object_lifetime). The programmer determines when objects are created, and the Java runtime is responsible for recovering the memory once objects are no longer in use. Once no references to an object remain, the [unreachable memory](http://en.wikipedia.org/wiki/Unreachable_memory) becomes eligible to be freed automatically by the garbage collector. Something similar to a [memory leak](http://en.wikipedia.org/wiki/Memory_leak) may still occur if a programmer's code holds a reference to an object that is no longer needed, typically when objects that are no longer needed are stored in containers that are still in use. If methods for a nonexistent object are called, a "null pointer exception" is thrown.

One of the ideas behind Java's automatic memory management model is that programmers can be spared the burden of having to perform manual memory management. In some languages, memory for the creation of objects is implicitly allocated on the [stack](http://en.wikipedia.org/wiki/Stack_(data_structure)), or explicitly allocated and deallocated from the [heap](http://en.wikipedia.org/wiki/Dynamic_memory_allocation). In the latter case the responsibility of managing memory resides with the programmer. If the program does not deallocate an object, a [memory leak](http://en.wikipedia.org/wiki/Memory_leak) occurs. If the program attempts to access or deallocate memory that has already been deallocated, the result is undefined and difficult to predict, and the program is likely to become unstable and/or crash. This can be partially remedied by the use of [smart pointers](http://en.wikipedia.org/wiki/Smart_pointer), but these add overhead and complexity. Note that garbage collection does not prevent "logical" memory leaks, i.e. those where the memory is still referenced but never used.

Garbage collection may happen at any time. Ideally, it will occur when a program is idle. It is guaranteed to be triggered if there is insufficient free memory on the heap to allocate a new object; this can cause a program to stall momentarily. Explicit memory management is not possible in Java.

Java does not support C/C++ style [pointer arithmetic](http://en.wikipedia.org/wiki/Pointer_(computer_programming)), where object addresses and unsigned integers (usually long integers) can be used interchangeably. This allows the garbage collector to relocate referenced objects and ensures type safety and security.

As in [C++](http://en.wikipedia.org/wiki/C%2B%2B) and some other object-oriented languages, variables of Java's [primitive data types](http://en.wikipedia.org/wiki/Primitive_data_type) are not objects. Values of primitive types are either stored directly in fields (for objects) or on the [stack](http://en.wikipedia.org/wiki/Stack-based_memory_allocation) (for methods) rather than on the heap, as commonly true for objects (but see [Escape analysis](http://en.wikipedia.org/wiki/Escape_analysis)). This was a conscious decision by Java's designers for performance reasons. Because of this, Java was not considered to be a pure object-oriented programming language. However, as of Java 5.0, [auto boxing](http://en.wikipedia.org/wiki/Autoboxing) enables programmers to proceed as if primitive types were instances of their wrapper class.

Java contains multiple types of garbage collectors. By default, Hotspot uses the [Concurrent Mark Sweep collector](http://en.wikipedia.org/wiki/Concurrent_Mark_Sweep_collector), also known as the CMS Garbage Collector. However, there are also several other garbage collectors that can be used to manage the Heap. For 90% of applications in Java, the CMS Garbage Collector is good enough.

The syntax of Java is largely derived from [C++](http://en.wikipedia.org/wiki/C%2B%2B). Unlike C++, which combines the syntax for structured, generic, and object-oriented programming, Java was built almost exclusively as an object-oriented language. All code is written inside a class, and everything is an object, with the exception of the primitive data types (integers, floating-point numbers, Boolean values, and characters), which are not classes for performance reasons.

Unlike C++, Java does not support [operator](http://en.wikipedia.org/wiki/Operator_(programming)) [overloading](http://en.wikipedia.org/wiki/Operator_overloading) or [multiple inheritance](http://en.wikipedia.org/wiki/Multiple_inheritance) for classes. This simplifies the language and aids in preventing potential errors and [anti-pattern](http://en.wikipedia.org/wiki/Anti-pattern) design.

Java uses similar commenting methods to C++. There are three different styles of comments: a single line style marked with two slashes (//), a multiple line style opened with /\* and closed with \*/, and the [Javadoc](http://en.wikipedia.org/wiki/Javadoc) commenting style opened with /\*\* and closed with \*/. The Javadoc style of commenting allows the user to run the Javadoc executable to compile documentation for the program.

**3.2 INTEGRATED DEVELOPMENT ENVIRONMENT: NETBEANS IDE 7.1**

NetBeans  refers to both a  [platform](http://en.wikipedia.org/wiki/Platform_(computing)) framework for  desktop applications, and an [integrated development environment](http://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) for developing with Java, [JavaScript](http://en.wikipedia.org/wiki/JavaScript), [PHP](http://en.wikipedia.org/wiki/PHP), Python. The NetBeans IDE 7.0 no longer supports [Ruby](http://en.wikipedia.org/wiki/Ruby_(programming_language)) and [Ruby on Rails](http://en.wikipedia.org/wiki/Ruby_on_Rails), but a third party has begun work on a separate plug-in.

The NetBeans IDE is written in Java and can run on Windows, Mac OS, Linux, Solaris and other platforms supporting a compatible [JVM](http://en.wikipedia.org/wiki/Java_Virtual_Machine). A pre-existing JVM or a [JDK](http://en.wikipedia.org/wiki/Java_Development_Kit) is not required.

The NetBeans platform allows applications to be developed from a set of modular [software components](http://en.wikipedia.org/wiki/Software_component) called modules. Applications based on the NetBeans platform (including the NetBeans IDE) can be extended by [third party developers](http://en.wikipedia.org/wiki/Third_party_developer).

The NetBeans Platform is a reusable [framework](http://en.wikipedia.org/wiki/Software_framework) for simplifying the development of [Java Swing](http://en.wikipedia.org/wiki/Java_Swing) desktop applications. The NetBeans IDE bundle for Java SE contains what is needed to start developing NetBeans plugins and NetBeans Platform based applications; no additional SDK is required.

Applications can install modules dynamically. Any application can include the Update Center module to allow users of the application to download [digitally-signed](http://en.wikipedia.org/wiki/Digital_signature) upgrades and new features directly into the running application. Reinstalling an upgrade or a new release does not force users to download the entire application again.

The platform offers reusable services common to desktop applications, allowing developers to focus on the logic specific to their application. Among the features of the platform are:

* User interface management (e.g. menus and toolbars)
* User settings management
* Storage management (saving and loading any kind of data)
* Window management
* Wizard framework (supports step-by-step dialogs)
* NetBeans Visual Library
* Integrated development tools

NetBeans IDE is a free, open-source, cross-platform IDE with built-in-support for Java Programming Language.

NetBeans IDE is an [open-source](http://en.wikipedia.org/wiki/Open_source) integrated development environment. NetBeans IDE supports development of all Java application types ([Java SE](http://en.wikipedia.org/wiki/Java_Platform,_Standard_Edition) (including [JavaFX](http://en.wikipedia.org/wiki/JavaFX" \o "JavaFX)),  [Java ME](http://en.wikipedia.org/w/index.php?title=Java_Platform,_Mobile_Edition&action=edit&redlink=1), [web](http://en.wikipedia.org/wiki/Web_application), [EJB](http://en.wikipedia.org/wiki/EJB) and [mobile](http://en.wikipedia.org/wiki/MIDlet) applications) out of the box. Among other features are an [Ant](http://en.wikipedia.org/wiki/Apache_Ant)-basedprojectsystem,Mavensupport, [refactorings](http://en.wikipedia.org/wiki/Refactoring), [versioncontrol](http://en.wikipedia.org/wiki/Version_control_system) (supporting [CVS](http://en.wikipedia.org/wiki/Concurrent_Versions_System), [Subversion](http://en.wikipedia.org/wiki/Subversion_(software)), [Mercurial](http://en.wikipedia.org/wiki/Mercurial_(software)) and [Clearcase](http://en.wikipedia.org/wiki/Clearcase)).

Modularity

All the functions of the IDE are provided by modules. Each module provides a well defined function, such as support for the [Java language](http://en.wikipedia.org/wiki/Java_(programming_language)), editing, or support for the [CVS](http://en.wikipedia.org/wiki/Concurrent_Versions_System) versioning system, and SVN. NetBeans contains all the modules needed for Java development in a single download, allowing the user to start working immediately. Modules also allow NetBeans to be extended. New features, such as support for other programming languages, can be added by installing additional modules. For instance, [Sun Studio](http://en.wikipedia.org/wiki/Sun_Studio_Compiler_Suite), Sun Java Studio Enterprise, and Sun from [Sun Microsystems](http://en.wikipedia.org/wiki/Sun_Microsystems) are all based on the NetBeans IDE.

Modules that are part of the NetBeans IDE :

### NetBeans Profiler

The NetBeans Profiler is a tool for the monitoring of Java applications: It helps developers find memory leaks and optimize speed. Formerly downloaded separately, it is integrated into the core IDE since version 6.0.

The [Profiler](http://en.wikipedia.org/wiki/Profiler_(computer_science)) is based on a Sun Laboratories research project that was named JFluid. That research uncovered specific techniques that can be used to lower the overhead of profiling a Java application. One of those techniques is dynamic byte code instrumentation, which is particularly useful for profiling large Java applications. Using dynamic byte code instrumentation and additional algorithms, the NetBeans Profiler is able to obtain runtime information on applications that are too large or complex for other profilers. NetBeans also support Profiling Points that let you profile precise points of execution and measure execution time.

GUI Design Tool

Formerly known as project Matisse, the GUI design-tool enables developers to prototype and design [Swing](http://en.wikipedia.org/wiki/Swing_(Java)) [GUIs](http://en.wikipedia.org/wiki/Graphical_user_interface) by dragging and positioning GUI components.

The GUI builder has built-in support for JSR 295 (Beans Binding technology), but the support for JSR 296 ([Swing Application Framework](http://en.wikipedia.org/wiki/Swing_Application_Framework)) was removed in 7.1, without prior warning.

NetBeans JavaScript Editor

The NetBeans JavaScript editor provides extended support for [JavaScript](http://en.wikipedia.org/wiki/JavaScript), Ajax, and [CSS](http://en.wikipedia.org/wiki/CSS).

JavaScript editor features comprise [syntax highlighting](http://en.wikipedia.org/wiki/Syntax_highlighting), [refactoring](http://en.wikipedia.org/wiki/Code_refactoring), [code completion](http://en.wikipedia.org/wiki/Autocomplete) for native objects and functions, generation of JavaScript class skeletons, generation of [Ajax](http://en.wikipedia.org/wiki/Ajax_(programming)) [callbacks](http://en.wikipedia.org/wiki/Callback_(computer_science)) from a template; and automatic [browser compatibility](http://en.wikipedia.org/wiki/JavaScript#Compatibility_considerations) checks.

CSS editor features comprise [code completion](http://en.wikipedia.org/wiki/Autocomplete) for styles names, quick navigation through the navigator panel, displaying the CSS rule declaration in a List View and file structure in a Tree View, sorting the outline view by name, type or declaration order (List & Tree), creating rule declarations (Tree only), refactoring a part of a rule name (Tree only).

**3.3 JCREATOR IDE**

JCreator is a powerful IDE for Java.JCreator is the development tool for every programmer that likes to do what he does best: programming. It is faster, more efficient and more reliable than other Java IDE’s. Therefore it is the perfect tool for programmers of every level, from learning programmer to Java-specialist.

JCreator provides the user with a wide range of functionality such as : Project management, project templates, code-completion, debugger interface, editor with syntax highlighting, wizards and a fully customizable user interface With JCreator you can directly compile or run your Java program without activating the main document first. JCreator will automatically find the file with the main method or the html file holding the java applet, then it will start the appropriate tool. JCreator is written entirely in C++, which makes it fast and efficient compared to the Java based editors/IDE's.

JCreator is a powerful interactive development environment (IDE) for Java technologies that provides more power at your fingertips than all the ordinary editors. What makes JCreator a cut above the rest? Just look at what JCreator can do for you:

* Manage projects with ease in an interface that is much like Microsoft Visual Studio.
* Define your own color schemes in XML for unlimited ways to organize your code.
* Use JCreator to wrap around your existing projects and use different JDK profiles.
* Get down to writing code quickly with our project templates.
* Make viewing your project a breeze with our class browser.
* Debug with an easy, intuitive interface. No need for silly DOS prompts!
* Walk through our wizards and cut to the chase of writing your project quickly and easily.
* Manage and exchange your code with our Ant and CVS integration.
* Save valuable time on Class path configuration; let JCreator do it for you.
* Customize our user interface the way that you like it.
* Set up your own run-time environments to run your application as an applet, in a JUnit environment, or in a DOS window.
* Experience the satisfaction of faster speed, yet lower system requirements than with other IDEs.
* Extend Your Power

Unlike most IDEs, JCreator has two types of tools that you can configure. The first type is the Java Development Kit (JDK) tools. You can use JDK tools to compile, debug, and run the project. You can attach these tools to your project using the Project Settings dialog box. If no project is available, JCreator runs the default projects. You can easily create your own tools for calling the JDK applications, such as the following:

* Compiler
* Ant builder
* Interpreter
* Applet viewer
* Debugger

The second type of tool is more general and allows you to extend the capabilities of JCreator to fit your needs by allowing you to call external functions and utilities. You can assign these general tools to the Wrench buttons located on the Tools toolbar in the workspace. These buttons display tooltips, such as User Tool 1, User Tool 2, and so forth. These tools can have many uses, such as the following:

* XML validator
* RMI compiler
* JAVA code formatter
* Version control
* Batch files

JCreator is backwards-compatible with all previous versions of the JDKs published by Sun Microsystems. With one click, you can toggle between different JDK profiles. Each profile defines the JDK home path and any additional files for the class path, such as the following:

* JDK 1.6.0
* JDK 1.6.0 - J2EE
* JDK 1.6.0 - J2ME
* JDK 1.5.0
* JDK 1.4.2
* JDK 1.2.2
* JDK 1.0.2
* Fast for the Future

Faster and more efficient than Java-based IDEs, JCreator is a powerful tool that is written entirely in a native Windows language. In addition, JCreator is professionally designed to meet Microsoft Windows interface guidelines; you can work quickly and efficiently with the intuitive Windows interface.

Light-Weight, Heavy-Weight

Right at your finger tips, you have all the power that you need. Other IDEs are bloated with features you'll never use. JCreator is faster and more efficient than most IDEs, especially Java-based editors. You'll use all the features in JCreator Pro, instead of wading through many features that you'll never use. Don't be fooled with the light-weight simplicity. JCreator is packed with power. Even your largest projects run with ease in JCreator.

Accelerate your development cycle

As a feature-rich Java IDE, JCreator has always helped you write code faster. Now, we help you write code better—with our powerful, fully-integrated debugger. You get even better productivity through an easy-to-use, fully-loaded Java debugger with advanced features. You'll quickly produce clean, bug-free code. Now, you can accelerate your development cycle—and say "code complete" in no time!

Installation of JCREATOR

Before using JCreator, install the Java Development Kit (JDK) from Oracle, Sun Microsystems. You will need to download and install both the JDK and the Java API documentation.

Java Development Kit (JDK): Go to the download page for JDK Version 1.7: http://www.oracle.com/technetwork/java/javase/downloads/index.html.

Click on the [download JDK] button of JDK 7 update 2, and you will be forwarded to the download page.

Double check that the download page is for Java SE Development Kit 7u2

Select windows as platform and click on the link with the red arrow

Install the JDK in the default directory C:\Program Files\Java\jdk1.7.0\.

Java API Documentation : You can download the documentation, or bookmark the following link http://download.oracle.com/javase/. Go to the download page for JDK Version 1.7: http://www.oracle.com/technetwork/java/javase/downloads/index.html".

Scroll to the bottom till the section Additional Resources

Download the file jdk-7-doc.zip.

Unzip the archive in the same directory as the JDK; that is, C:\..\jdk1.7.0\.

Using the Setup Wizard to Start JCreator

Launch JCreator.

Follow the Setup wizard, setting the paths to C:\..\jdk1.7.0\ and C:\..\jdk1.7.0\docs.

Starting JCreator Without Using the Setup Wizard

With JCreator active, open the Options window from the Configure menu.

Click the JDK Profiles option.

Click the default item in the list, and click Edit.

Ensure that the Name field shows the correct version of the selected JDK directory; in this case, JDK version jdk1.6.0.

Set the JDK Home path by clicking the Browse button next to this field.

Browse to the root directory of the JDK installation: C:\..\jdk1.7.0\. Click OK.

Click the Documentation tab, and Add the path: C:\..\jdk1.7.0\docs. Click OK.

Close the windows.

**CHAPTER 4**

**DESIGN**

**4.1 USE-CASE DIAGRAM**

A use case diagram in the Unified Modelling Language (UML) is a type of behavioural diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. Unified Modelling Language (UML) is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by the Object Management Group.UML includes a set of graphic notation techniques to create visual models of object-oriented software-intensive systems.

The following use-cases have been derived from the captured requirements. They are :

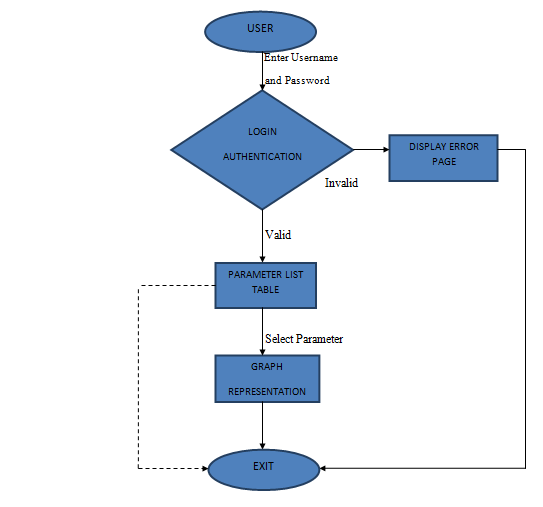
* Data Acquisition: The data has to be acquired and delivered in real-time.
* Visualization: The data delivered has to be acquired and visualized in real-time

They are represented in the form of use-case diagrams as follows:



**Fig. 4.1 Use-case diagram**

**4.2 CONTROL FLOW DIAGRAM**



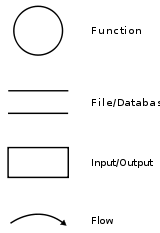
**Fig. 4.2 Control Flow diagram**

**4.3 DATA FLOW DIAGRAM**

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an [information system](http://en.wikipedia.org/wiki/Information_system), modeling its process aspects. Often they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the [visualization](http://en.wikipedia.org/wiki/Data_visualization) of [data processing](http://en.wikipedia.org/wiki/Data_processing) (structured design).

A DFD shows what kinds of data will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel (which is shown on a [flowchart](http://en.wikipedia.org/wiki/Flowchart)).

OVERVIEW

[](http://en.wikipedia.org/wiki/File:Data-flow-diagram-notation.svg)

It is common practice to draw the [context-level data flow diagram](http://en.wikipedia.org/wiki/System_context_diagram) first, which shows the interaction between the system and external agents which act as data sources and data sinks. On the context diagram the system's interactions with the outside world are modeled purely in terms of data flows across the systemboundary. The context diagram shows the entire system as a single process, and gives no clues as to its internal organization.

This context-level DFD is next "exploded", to produce a Level 0 DFD that shows some of the detail of the system being modeled. The Level 0 DFD shows how the system is divided into sub-systems (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the system as a whole. It also identifies internal data stores that must be present in order for the system to do its job, and shows the flow of data between the various parts of the system.

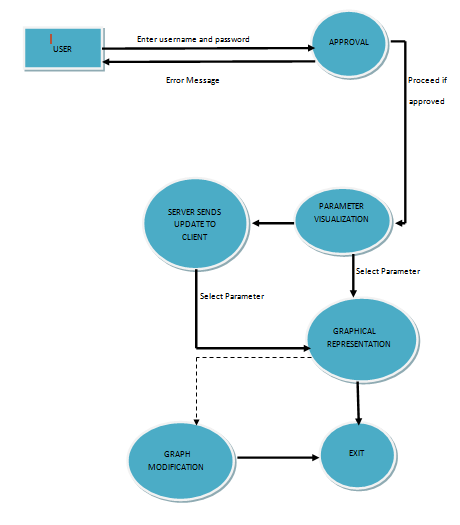
Data flow diagrams were proposed by [Larry Constantine](http://en.wikipedia.org/wiki/Larry_Constantine), the original developer of structured design, based on Martin and Estrin's "data flow graph" model of computation.

Data flow diagrams are one of the three essential perspectives of the structured-systems analysis and design method [SSADM](http://en.wikipedia.org/wiki/SSADM). The sponsor of a project and the end users will need to be briefed and consulted throughout all stages of a system's evolution. With a data flow diagram, users are able to visualize how the system will operate, what the system will accomplish, and how the system will be implemented. The old system's dataflow diagrams can be drawn up and compared with the new system's data flow diagrams to draw comparisons to implement a more efficient system. Data flow diagrams can be used to provide the end user with a physical idea of where the data they input ultimately has an effect upon the structure of the whole system from order to dispatch to report. How any system is developed can be determined through a data flow diagram model.

In the course of developing a set of leveled data flow diagrams the analyst/designers is forced to address how the system may be decomposed into component sub-systems, and to identify the [transaction data](http://en.wikipedia.org/wiki/Transaction_data) in the [data model](http://en.wikipedia.org/wiki/Data_model).

Data flow diagrams can be used in both Analysis and Design phase of SDLC.

There are different notations to draw data flow diagrams (Yourdon & Coad and Gane & Sarson), defining different visual representations for processes, data stores, data flow, and external entities.

****

**Fig. 4.3 Data Flow Diagram**

**CHAPTER 5**

**IMPLEMENTATION**

An implementation is a realization of a technical specification or algorithm as a program, software component, or other computer system through programming and deployment. A programming language implementation is a system for executing programs written in a programming language.

The objects mentioned in the object-oriented design will be coded using Java SE6.Java Platform, Standard Edition or Java SE is a widely used platform for programming in the Java language. It is the Java Platform used to deploy portable applications for general use. In practical terms, Java SE consists of a virtual machine, which must be used to run Java programs, together with a set of libraries (or "packages") needed to allow the use of file systems, networks, graphical interfaces, and so on, from within those programs.

The object-oriented paradigm is used to realize the different units and sub-units of the software .Java is the programming language used for this purpose. Java refers to a number of computer software products and specifications from Sun Microsystems, a subsidiary of Oracle Corporation, that together provide a system for developing application software and deploying it in a cross-platform environment. Java is used in a wide variety of computing platforms from embedded devices and mobile phones on the low end, to enterprise servers and supercomputers on the high end. Java is used in mobile phones, Web servers and enterprise applications, and while less common on desktop computers; Java applets are sometimes used to provide improved and secure functionalities while browsing the World Wide Web.

Java eliminates certain low-level constructs such as pointers and has a very simple memory model where every object is allocated on the heap and all variables of object types are references. Memory management is handled through integrated automatic garbage collection performed by the Java Virtual Machine (JVM).

The Java Platform: An edition of the Java platform is the name for a bundle of related programs from Sun that allow for developing and running programs written in the Java programming language. The platform is not specific to any one processor or operating system, but rather an execution engine (called a virtual machine) and a compiler with a set of libraries that are implemented for various hardware and operating systems so that Java programs can run identically on all of them

* Java Card: A technology that allows small Java-based applications (applets) to be run securely on smart cards and similar small-memory-footprint devices.
* Java ME (Micro Edition): Specifies several different sets of libraries (known as profiles) for devices that are sufficiently limited that supplying the full set of Java libraries would take up unacceptably large amounts of storage.
* Java SE (Standard Edition): For general-purpose use on desktop PCs, servers and similar devices.
* Java EE (Enterprise Edition): Java SE plus various APIs useful for multi-tier client–server enterprise applications.

Java Platform, Standard Edition or Java SE is a widely used platform for programming in the Java language. It is the Java Platform used to deploy portable applications for general use. In practical terms, Java SE consists of a virtual machine, which must be used to run Java programs, together with a set of libraries (or "packages") needed to allow the use of file systems, networks, graphical interfaces, and so on, from within those programs.

Certain Java libraries will be used. They are

* Java Development Kit (JDK)
* Java Swings API
* JFreeChart
* JCommon
* Java 3D API
* Java RMI API

Java Development Kit (JDK): The Java Development Kit (JDK) is a Sun Microsystems product aimed at Java developers. The JDK has as its primary components a collection of programming tools, including:

* Java – the loader for Java applications. This tool is an interpreter and can interpret the class files generated by the javac compiler. Now a single launcher is used for both development and deployment. The old deployment launcher, jre, no longer comes with Sun JDK.
* javac – the compiler, which converts source code into Java byte code
* jar – the archiver, which packages related class libraries into a single JAR file. This tool also helps manage JAR files.
* javadoc – the documentation generator, which automatically generates documentation from source code comments
* jdb – the debugger
* jps – the process status tool, which displays process information for current Java processes
* javap – the class file disassembler
* applet viewer – this tool can be used to run and debug Java applets without a web browser
* javah – the C header and stub generator, used to write native methods
* javaws – the Java Web Start launcher for JNLP applications
* extcheck – a utility which can detect JAR-file conflicts
* apt – the annotation-processing tool [1]
* jhat – (experimental) Java heap analysis tool
* jstack – (experimental) utility which prints Java stack traces of Java threads
* jstat – (experimental) Java Virtual Machine statistics monitoring tool
* jstatd – (experimental) jstat daemon
* jinfo – (experimental) this utility gets configuration information from a running Java process or crash dump.
* jmap – (experimental) This utility outputs the memory map for Java and can print shared object memory maps or heap memory details of a given process or core dump.
* idlj – the IDL (Interface Description Language)-to-Java compiler. This utility generates Java bindings from a given Java IDL file.
* policytool – the policy creation and management tool, which can determine policy for a Java runtime, specifying which permissions are available for code from various sources
* VisualVM – visual tool integrating several command-line JDK tools and lightweight[clarification needed] performance and memory profiling capabilities
* wsimport – generates portable JAX-WS artifacts for invoking a web service.
* jrunscript – Java command-line script shell.

The JDK also comes with a complete Java Runtime Environment, usually called a private runtime. It consists of a Java Virtual Machine and all of the class libraries present in the production environment, as well as additional libraries only useful to developers, such as the internationalization libraries and the IDL libraries. An interface description language or IDL is a specification language used to describe a software component's interface.

|  |  |
| --- | --- |
| Class Name | Description |
| JFrame | An extended version of java.awt.Frame that adds support for the JFC/Swing component architecture. |
| JPanel | JPanel is a generic lightweight container. |
| JInternalFrame | A lightweight object that provides many of the features of a native frame, including dragging, closing, becoming an icon, resizing, title display, and support for a menu bar. |
| JLabel | A display area for a short text string or an image, or both. |
| JButton | An implementation of a "push" button. |
| JComboBox | A component that combines a button or editable field and a drop-down list. |

Java Swings API: Swing is the primary Java GUI widget toolkit. It is part of Sun Microsystems' Java Foundation Classes (JFC) — an API for providing a graphical user interface (GUI) for Java programs. Swing was developed to provide a more sophisticated set of GUI components than the earlier

Table 5.1 Some of the javax.swing package classes used

Abstract Window Toolkit (AWT). The Abstract Window Toolkit (AWT) is Java's original platform-independent windowing, graphics, and user-interface widget toolkit.AWT is also the GUI toolkit for a number of Java ME profiles.

Swing provides a native look and feel that emulates the look and feel of several platforms, and also supports a pluggable look and feel that allows applications to have a look and feel unrelated to the underlying platform. It is more powerful and flexible components than AWT. In addition to familiar components such as buttons, check box and labels, swings provide several advanced components such as tabbed panel, scroll panes, trees, tables and lists.

Unlike AWT components, Swing components are not implemented by platform-specific code. Instead they are written entirely in Java and therefore are platform-independent. The term "lightweight" is used to describe such an element.

GUI: The GUI of the software is built using javax.swing package

|  |  |  |
| --- | --- | --- |
| Class Name | Method Signature | Description |
| JFrame | void setDefaultCloseOperation(int operation) | Sets the operation that will happen by default when the user initiates a "close" on this frame. |
| JLabel | public void setText(String text) | Defines the single line of text this component will display. |
| public String getText() | Returns the text string that the label displays. |
| JPanel | public void setLayout(LayoutManager mgr) | Sets the layout manager for this container. |

Table 5.2 some of the javax.swing package methods used

|  |  |  |
| --- | --- | --- |
| Listener Name | Event | Description |
| ActionListener | ActionEvent | A semantic event which indicates that a component-defined action occurred. This high-level event is generated by a component (such as a Button) when the component-specific action occurs (such as being pressed). |
| WindowListener | WindowEvent | A low-level event that indicates that a window has changed its status. This low-level event is generated by a Window object when it is opened, closed, activated, deactivated, iconified, or deiconified, or when focus is transferred into or out of the Window. |
| ItemListener | ItemEvent | A semantic event which indicates that an item was selected or deselected. This high-level event is generated by an ItemSelectable object (such as a List) when an item is selected or deselected by the user. |

Table 5.3 Events and event listeners used

Package javax.swing.table : Provides classes and interfaces for dealing with javax.swing.JTable.

Class DefaultTableModel

All Implemented Interfaces:

Serializable, TableModel

public class DefaultTableModel

extends AbstractTableModel

Implements Serializable

This is an implementation of TableModel that uses a Vector of Vectors to store the cell value objects.

Constructor Summary

DefaultTableModel()

Constructs a default DefaultTableModel which is a table of zero columns and zero rows.

DefaultTableModel(int numRows, int numColumns)

Constructs a DefaultTableModel with numRows and numColumns of null object values.

|  |  |  |
| --- | --- | --- |
| Method Name | Method | Description |
| addColumn | public void addColumn(Object columnName Vector columnData) | Adds a column to the model. The new column will have the identifier columnName. columnData is the optional vector of data for the column. If it is null the column is filled with null values. Otherwise, the new data will be added to model starting with the first element going to row 0, etc. This method will send a tableChanged notification message to all the listeners. |
| setValueAt | public void setValueAt(Object aValue, int row, int column) | Sets the object value for the cell at column and row. A value is the new value. This method will generate a tableChanged notification. |

Fig. 5.4 Methods of Default Table class

Java Awt Package: Graphics Class

The Graphics class is the abstract base class for all graphics contexts that allow an application to draw onto components that are realized on various devices, as well as onto off-screen images. A Graphics object encapsulates state information needed for the basic rendering operations that Java supports. This state information includes the following properties:

The Component object on which to draw.

A translation origin for rendering and clipping coordinates.

* The current clip.
* The current color.
* The current font.
* The current logical pixel operation function (XOR or Paint).
* The current XOR alternation color (see setXORMode(java.awt.Color)).

Coordinates are infinitely thin and lie between the pixels of the output device. Operations that draw the outline of a figure operate by traversing an infinitely thin path between pixels with a pixel-sized pen that hangs down and to the right of the anchor point on the path. Operations that fill a figure operate by filling the interior of that infinitely thin path. Operations that render horizontal text render the ascending portion of character glyphs entirely above the baseline coordinate. The graphics pen hangs down and to the right from the path it traverses. This has the following implications:

* If you draw a figure that covers a given rectangle, that figure occupies one extra row of pixels on the right and bottom edges as compared to filling a figure that is bounded by that same rectangle.
* If you draw a horizontal line along the same y coordinate as the baseline of a line of text, that line is drawn entirely below the text, except for any descenders.

All coordinates that appear as arguments to the methods of this Graphics object are considered relative to the translation origin of this Graphics object prior to the invocation of the method.

All rendering operations modify only pixels which lie within the area bounded by the current clip, which is specified by a Shape in user space and is controlled by the program using the Graphics object. This user clip is transformed into device space and combined with the device clip, which is defined by the visibility of windows and device extents. The combination of the user clip and device clip defines the composite clip, which determines the final clipping region. The user clip cannot be modified by the rendering system to reflect the resulting composite clip. The user clip can only be changed through the setClip or clipRect methods. All drawing or writing is done in the current color, using the current paint mode, and in the current font.

|  |  |  |
| --- | --- | --- |
| Method Name | Method Signature | Description |
| drawLine | void drawLine(int x1, int y1, int x2, int y2) | Draws a line, using the current color, between the points (x1, y1) and (x2, y2) in this graphics context's coordinate system. |
| drawString | void drawString(String str, int x, int y) | Draws the text given by the specified string, using this graphics context's current font and color. |
| fillOval | void fillOval(int x, int y, int width, int height) | Fills an oval bounded by the specified rectangle with the current color. |
| setColor | void setColor(Color c) | Sets this graphics context's current color to the specified color. |
| setFont | void setFont(Font font) | Sets this graphics context's font to the specified font. |

Table 5.5 Methods of the Graphics class

Java io package :

Provides for system input and output through data streams, serialization and the file system.

Interface summary

* DataInput : The DataInput interface provides for reading bytes from a binary stream and reconstructing from them data in any of the Java primitive types.
* DataOutput : The DataOutput interface provides for converting data from any of the Java primitive types to a series of bytes and writing these bytes to a binary stream.

Class Summary

* DataInputStream A data input stream lets an application read primitive Java data types from an underlying input stream in a machine-independent way.
* DataOutputStream A data input stream lets an application write primitive Java data types to an output stream in a portable way.
* FileInputStream A FileInputStream obtains input bytes from a file in a file system.

A data input stream lets an application read primitive Java data types from an underlying input stream in a machine-independent way. An application uses a data output stream to write data that can later be read by a data input stream.

Data input streams and data output streams represent Unicode strings in a format that is a slight modification of UTF-8. (For more information, see X/Open Company Ltd., "File System Safe UCS Transformation Format (FSS\_UTF)", X/Open Preliminary Specification, Document Number: P316. This information also appears in ISO/IEC 10646, Annex P.)

All characters in the range '\u0001' to '\u007F' are represented by a single byte:

0 bits 0-7

The null character '\u0000' and characters in the range '\u0080' to '\u07FF' are represented by a pair of bytes:

1 1 0 bits 6-10

1 0 bits 0-5

Characters in the range '\u0800' to '\uFFFF' are represented by three bytes:

1 1 1 0 bits 12-15

1 0 bits 6-11

1 0 bits 0-5

The two differences between this format and the "standard" UTF-8 format are the following:

The null byte '\u0000' is encoded in 2-byte format rather than 1-byte, so that the encoded strings never have embedded nulls.

Only the 1-byte, 2-byte, and 3-byte formats are used.

Class Data Input Stream :

All Implemented Interfaces:

* DataInput
* public class DataInputStream
* extends FilterInputStream
* implements DataInput

Constructor Detail : DataInputStream

public DataInputStream(InputStream in)

Creates a FilterInputStream and saves its argument, the input stream in, for later use. An internal

Parameters: in - the input stream.

|  |  |  |
| --- | --- | --- |
| Method Name | Method Signature | Descpription |
| readUTF | Publicfinal String readUTF() throws IOException | Bytes for this operation are read from the contained input stream. |
| readUTF | public static final String readUTF(DataInput in) throws IOException | Reads from the stream in a representation of a Unicode character string encoded in Java modified UTF-8 format; this string of characters is then returned as a String. The details of the modified UTF-8 representation are exactly the same as for the readUTF method of DataInput. |

Table 5.6 Methods of DataInputStream Class

Class DataOutputStream

All Implemented Interfaces:

DataOutput

public class DataOutputStream

extends FilterOutputStream

implements DataOutput

A data input stream lets an application write primitive Java data types to an output stream in a portable way. An application can then use a data input stream to read the data back in.

Constructor Summary

DataOutputStream(OutputStream out)

Creates a new data output stream to write data to the specified underlying output stream.

|  |  |  |
| --- | --- | --- |
| Method Name | Method Signature | Descpription |
| writeUTF | public final void writeUTF(String str)  throws IOException | Writes a string to the underlying output stream using UTF-8 encoding in a machine-independent manner.  First, two bytes are written to the output stream as if by the writeShort method giving the number of bytes to follow. This value is the number of bytes actually written out, not the length of the string. Following the length, each character of the string is output, in sequence, using the UTF-8 encoding for the character. If no exception is thrown, the counter written is incremented by the total number of bytes written to the output stream. This will be at least two plus the length of str, and at most two plus thrice the length of str. |

Table 5.7 Method of DataOutputStreamClass

Class FileInputStream

public class FileInputStream extends InputStream

A FileInputStream obtains input bytes from a file in a file system. What files are available depends on the host environment.

|  |  |  |
| --- | --- | --- |
| Class Name | Method Signature | Descpription |
| FileInputStream | public FileInputStream(String name) throws FileNotFoundException | Creates a FileInputStream by opening a connection to an actual file, the file named by the path name name in the file system. A new FileDescriptor object is created to represent this file connection. |
| FileInputStream | public FileInputStream(File file) throws FileNotFoundException | Creates a FileInputStream by opening a connection to an actual file, the file named by the File object file in the file system. A new FileDescriptor object is created to represent this file connection. |
| FileInputStream | public FileInputStream(FileDescriptor fdObj) | Creates a FileInputStream by using the file descriptor fdObj, which represents an existing connection to an actual file in the file system. |

Table 5.8 Methods of FileInputStream Class

Package java.net

Provides the classes for implementing networking applications.

public class ServerSocket extends Object

This class implements server sockets. A server socket waits for requests to come in over the network. It performs some operation based on that request, and then possibly returns a result to the requester. The actual work of the server socket is performed by an instance of the SocketImpl class. An application can change the socket factory that creates the socket implementation to configure itself to create sockets appropriate to the local firewall.

Constructor Summary

ServerSocket(int port)

Creates a server socket on a specified port. A port of 0 creates a socket on any free port.The maximum queue length for incoming connection indications (a request to connect) is set to 50. If a connection indication arrives when the queue is full, the connection is refused.If the application has specified a server socket factory, that factory's createSocketImpl method is called to create the actual socket implementation. Otherwise a "plain" socket is created.

|  |  |  |
| --- | --- | --- |
| Method Name | Method Signature | Descpription |
| accept | public Socket accept() throws IOException | Listens for a connection to be made to this socket and accepts it. The method blocks until a connection is made. A new Socket s is created and, if there is a security manager, the security manager's checkAccept method is called with s.getInetAddress().getHostAddress() and s.getPort() as its arguments to ensure the operation is allowed. This could result in a SecurityException. |
| close | public void close() throws IOException | Closes this socket. |

Table 5.9 Methods of ServerSocket Class

Class Socket

public class Socket extends Object

This class implements client sockets (also called just "sockets"). A socket is an endpoint for communication between two machines. The actual work of the socket is performed by an instance of the SocketImpl class. An application, by changing the socket factory that creates the socket implementation, can configure itself to create sockets appropriate to the local firewall.

Constructor Summary

Protected Socket()

Creates an unconnected socket, with the system-default type of SocketImpl.

Socket(InetAddress address, int port)

Creates a stream socket and connects it to the specified port number at the specified IP address.

|  |  |  |
| --- | --- | --- |
| Method Name | Method Signature | Descpription |
| close | public void close() throws IOException | Closes this socket. |

Table 5.10 Method of Socket Class

BREEZY GUI

It is an often repeated statistic that 70% of the programming effort goes into interface development. Anyone who has used Java's complex Abstract Windowing Toolkit (AWT) has probably experienced the truth of this statistic first hand. While powerful and flexible, the AWT provides an ongoing challenge for any programmer, involving as it does the coordination of layout classes, event classes, and listener interfaces.

Fortunately, object-oriented systems are highly malleable, and we have been able to extend the AWT to create an easy to use and lightweight interface toolkit called BreezyGUI. BreezyGUI sits on top of the AWT, giving access to essential AWT capabilities while hiding the AWT's complexities. Users of BreezyGUI specify a window's appearance and associated event handling routines in the context of BreezyGUI's simplified yet realistic framework.

Not surprisingly BreezyGUI does not support all the features and nuances of the native AWT; however, it is able to handle the needs of developers in many situations.

We expect BreezyGUI to be especially appealing to those new to Java who normally face a very steep learning curve as they attempt to master the AWT. By beginning GUI development with BreezyGUI, programmers can explore event-driven programming, the properties of Java's basic window components, and the design of GUI-based applications in an easily understood setting. Later, as the need arises, they can transfer this knowledge to the more formidable setting of the native AWT. The time they spend using BreezyGUI will not have been wasted. Starting off easy is often the best and quickest way to mastery a difficult topic, and the last chapter of this manual shows the reader how to make the transition from BreezyGUI to the full AWT. We anticipate that even experienced programmers will sometimes prefer to use BreezyGUI, especially when they need to get an interface up and running quickly. Finally, we provide the full source code for BreezyGUI and invite experienced programmers to extend the framework to suit their particular needs.

BreezyGUI is easy to use, and this tutorial will get you off to a quick start. The tutorial makes the assumption that you are already familiar with the basics of Java and that you know how to write, compile, and run simple Java programs using classes you have designed yourself. However, you do not need to know anything about creating GUIs. That is the purpose of this tutorial. Here, in addition to learning BreezyGUI's powerful but simple features, you will explore general principles for constructing GUI based applications. These include:

Constructing interfaces that use labels, data entry fields, radio buttons, check boxes, list boxes, command buttons, and menus.

Nearly everything you learn will apply directly to Java's Abstract Windowing Toolkit (AWT), the ultimate and complex official tool for creating GUI applications in Java.

The Breezy Skeleton The basic structure of all BreezyGUI programs. A similar structure exists for applets and dialogs.

Doing it the AWT way : For the sake of comparison the temperature conversion program is rewritten using the native AWT. If you are not an experienced Java programmer, you will find this material incompressible, so do not hesitate to skip it.

Laying out window controls How to lay out an interface in BreezyGUI and properties of some of the basic window controls.

Messages and methods for window controls Programs manipulate controls by sending them messages. In this section we present methods for manipulating several basic classes of controls.

1. Applets -- BreezyGUI works equally well for creating applets as it does for creating stand alone applications. An applet is a Java program that is download with a web page and runs inside a browser window. Applets can add a great deal of useful functionality to a website.
2. More controls and other features -- BreezyGUI supports the creation of fairly sophisticated user interfaces. Read this material when you are ready to add more power, interest, and flexibility to your interfaces.

* Message boxes Displaying simple pop up messages.
* Text fields and text areas Using single and multiline text controls.
* Formatted output Justify information in a string before outputting it to the user.
* Menus and titles Adding drop down menus and titles to an application.
* Lots more controls Demonstration of checkboxes, radio buttons, scrolling lists, and choice lists.

1. Digression on software engineering -- Read this when you are ready to write large programs.The model/view pattern structuring large programs so that there is maximum independence between the graphical user interface and the underlying application. .
2. Graphics and mouse events -- Graphics opens up a whole new world of applications.

* Simple graphics Drawing in BreezyGUI windows.
* Mouse events Responding to mouse events.

1. Dialogs and multiple windows -- Dialogs and multiple windows add flexibility and power to the user interface.

* Dialogs Using BreezyGUI to create modal dialogs.

Applications with multiple windows Larger applications often require the use of multiple windows or view on an underlying model.

1. Features for terminal input -- Terminal I/O will never go away completely, but it can be made a little easier.

* Terminal Input Doing rudimentary terminal input is annoyingly complicated in Java and involves features that are perplexing to beginning programmers. BreezyGUI provides a class with several methods to simplify the input of data from the keyboard.

An Example:

import java.awt.\*;

import BreezyGUI.\*;

public class <name of class> extends GBFrame

{  
<create how the window will look - number of cells, labels, buttons, etc.>

public void buttonClicked (Button buttonObject)

{  
 <do what needs to be done when a button is clicked> }

public static void main (String[] args)

{  
Frame frm = new <name of class>( );

frm.setSize (<width of window>, <height of window>);

frm.setVisible (true);

}

}

**CHAPTER 6**

**SNAPSHOTS**

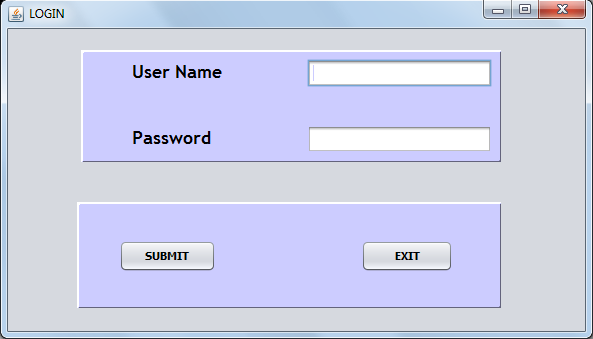
****

Figure 6.1 Login Page (GUI)

Access to the satellite parameters is limited to only a set of users. A user is supposed to enter his own unique username and password which is then verified. On successful login, the user is directed to the next set of modules.

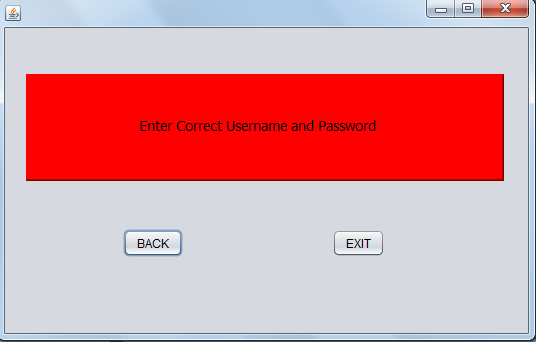
****

Figure 6.2 Error Notification

Upon entering an invalid (username, password) combination, the user is notified with an error message.

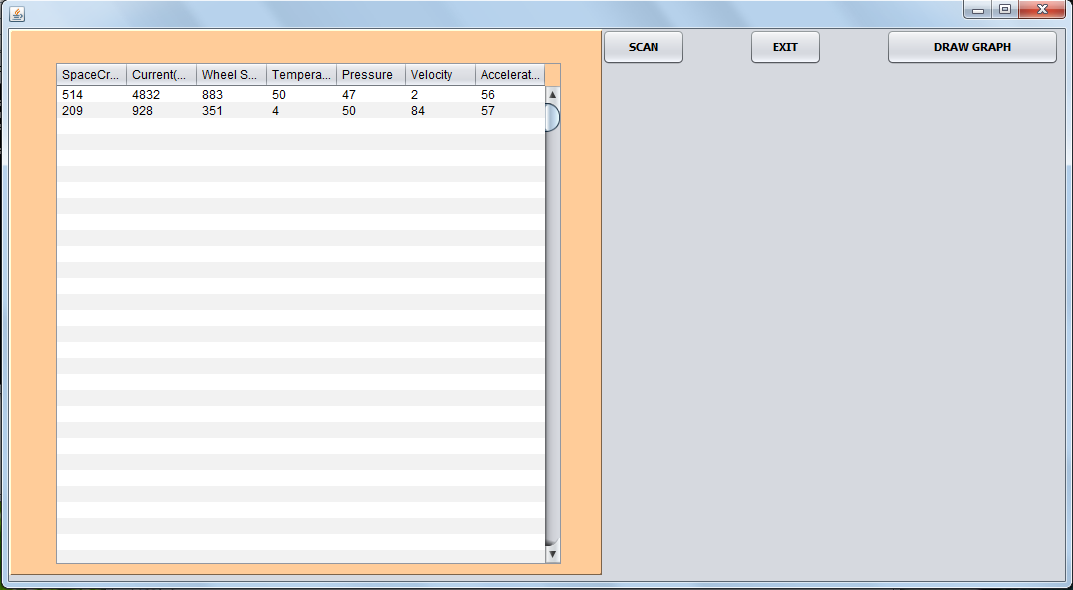
****

Figure 6.3 Server Side Parameter Visualisation

The user gets a list of all the real time parameter values,which can then be simply scanned or a graph may be plotted for each parameter individually to analyse their respective variations so as to determine the health of the satellite.This is the server end of the system.

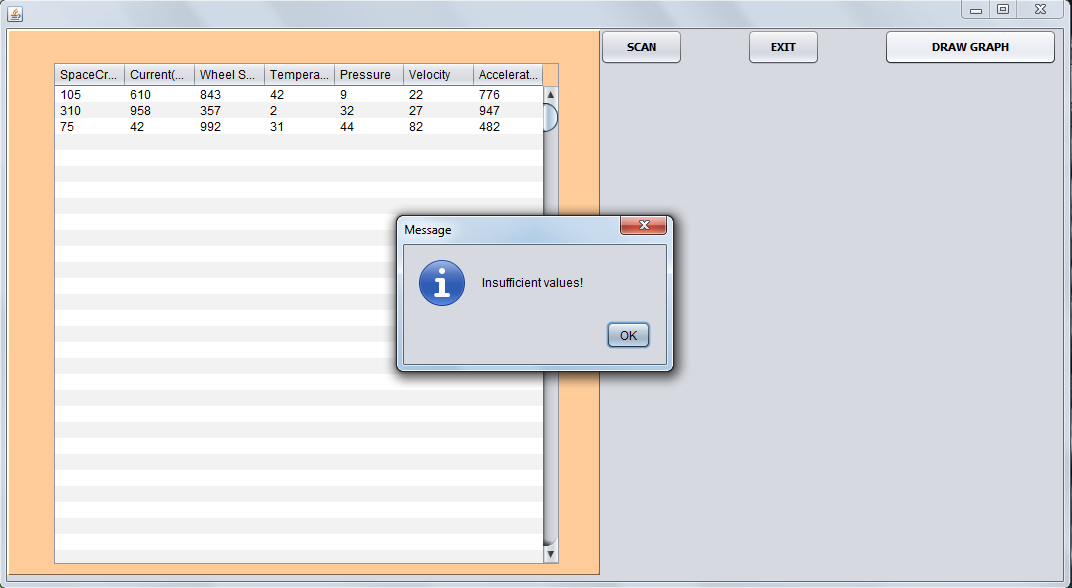
****

Figure 6.4 Server Side Error Notification

The condition for plotting a graph is that a minimum of 10 values are required.If draw graph option is selected by the user when the values are insufficient,an error message is displayed in a separate dialog box.

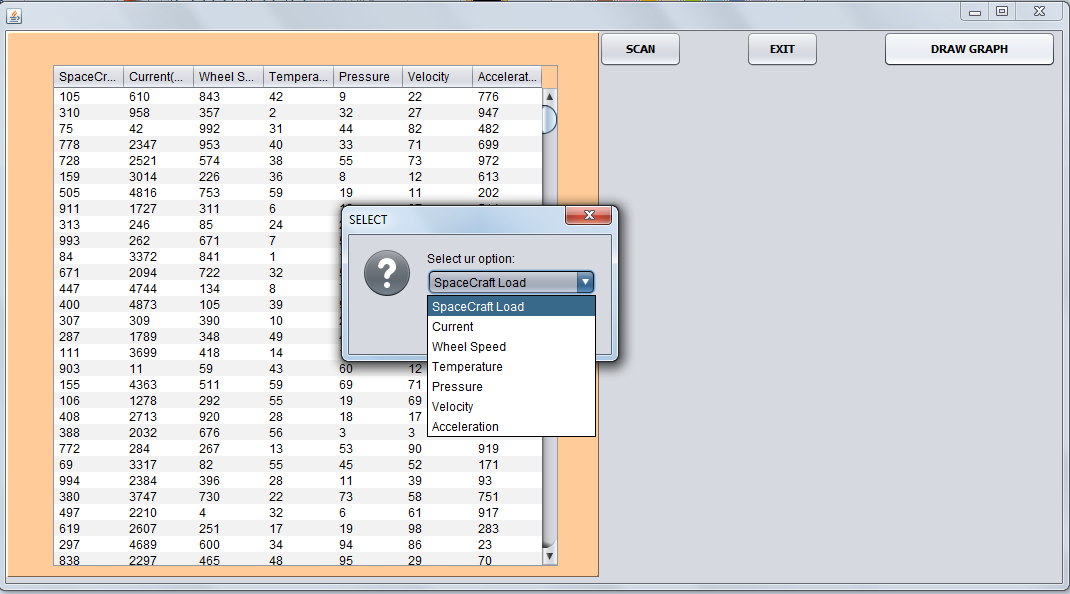
****

Figure 6.5 Parameter selection for plotting a graph on server side

If draw graph option is selected,the user is allowed to choose the parameter for which he wishes to plot a graph.

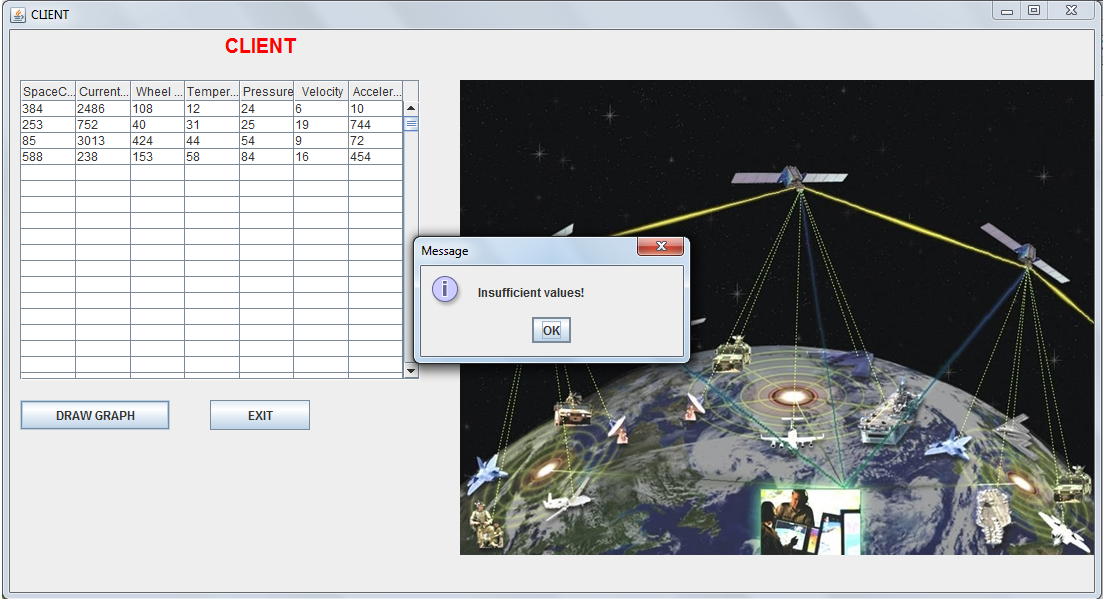
****

Figure 6.6 Client Side Error Notification

The condition for plotting a graph is that a minimum of 10 values are required.If draw graph option is selected by the user when the values are insufficient,an error message is displayed in a separate dialog box.

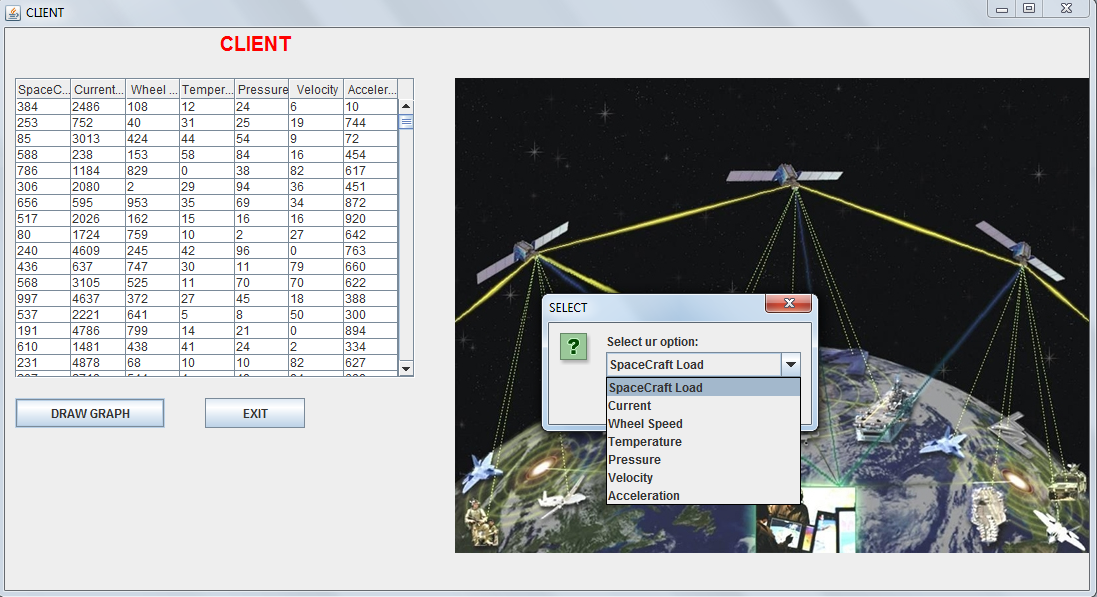
****

Figure 6.7Parameter selection for plotting a graph on client side

If draw graph option is selected,the user is allowed to choose the parameter for which he wishes to plot a graph.

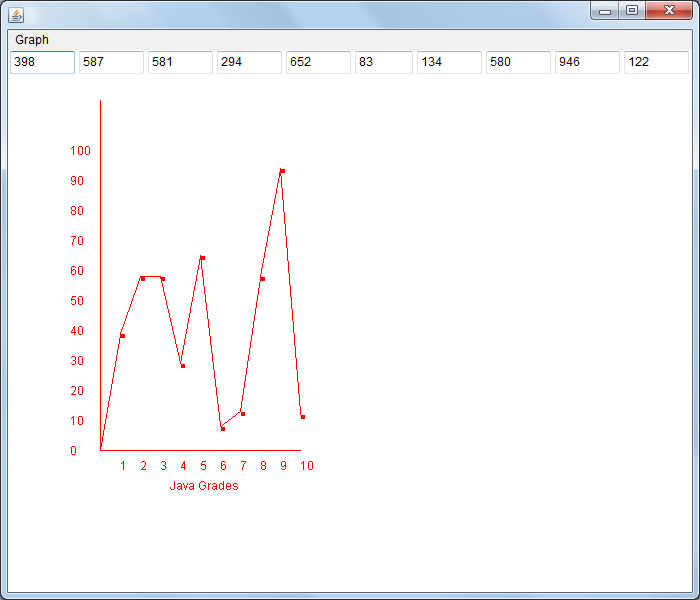
****

Figure 6.8 Graph depicting the behaviour of selected parameter

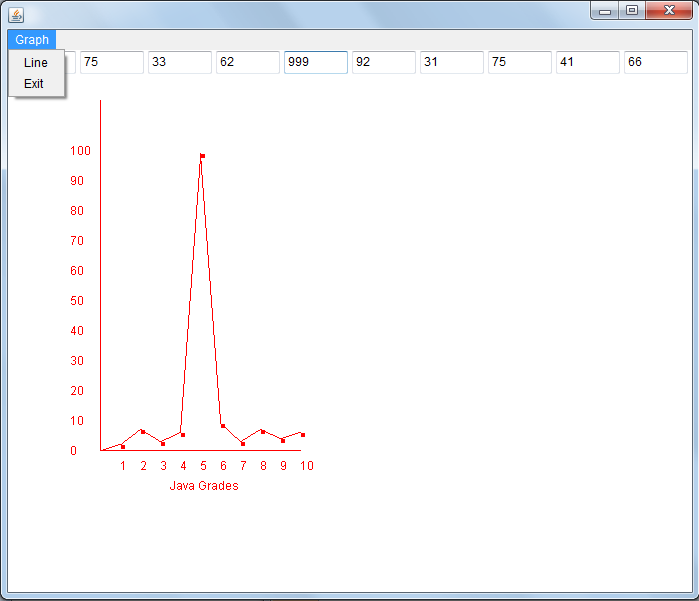
****

Figure 6.9 Graph depicting the behaviour of selected parameter

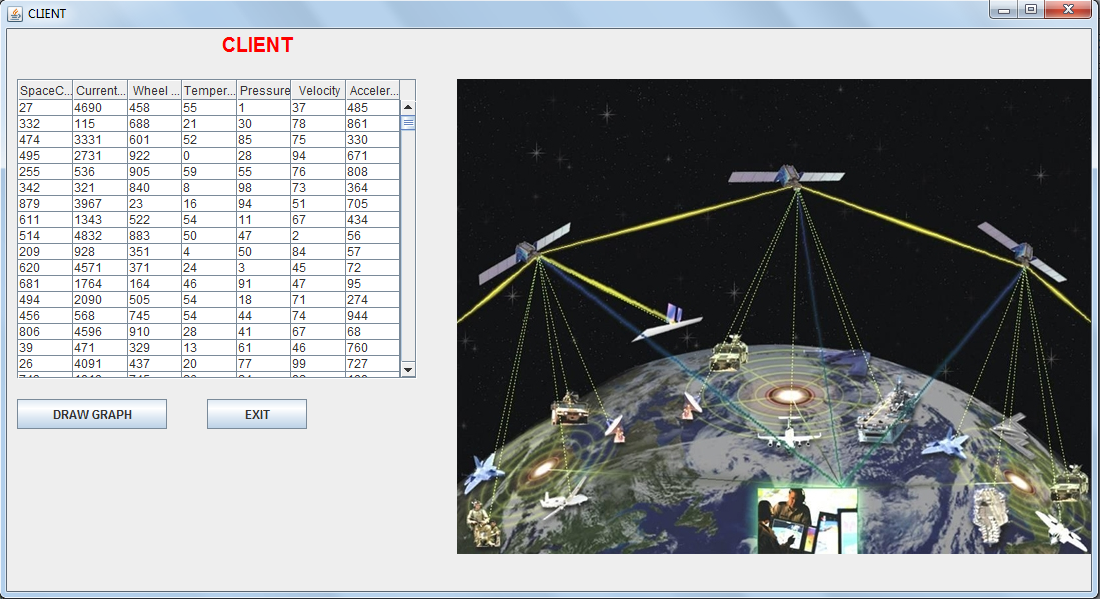
****

Figure 6.10 Client side parameter value list

**CHAPTER 6**

**TESTING**

**6.1 VERIFICATION**

Software verification is a broader and more complex discipline of software engineering whose goal is to assure that software fully satisfies all the expected requirements. Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing also provides an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs (errors or other defects).

Software testing, depending on the testing method employed, can be implemented at any time in the development process. However, most of the test effort occurs after the requirements have been defined and the coding process has been completed. As such, the methodology of the test is governed by the software development methodology adopted.

The software has been verified in two different stages. They are

* CSU/CSC Testing
* Integration Testing

**6.2 CSU TESTING**

Computer Software Unit(CSU) testing is a method by which individual units of source code are tested to determine if they are fit for use. A unit is the smallest testable part of an application. In object-oriented programming a unit is usually a method or an object Unit tests are typically written and run by software developers to ensure that code meets its design and behaves as intended. The goal of unit testing is to isolate each part of the program and show that the individual parts are correct. A unit test provides a strict, written contract that the piece of code must satisfy. As a result, it affords several benefits

* Facilitates change: Unit testing allows the programmer to refactor code at a later date, and make sure the module still works correctly.
* Simplifies integration: Unit testing may reduce uncertainty in the units themselves and can be used in a bottom-up testing style approach. By testing the parts of a program first and then testing the sum of its parts, integration testing becomes much easier.
* Documentation: Unit testing provides a sort of living documentation of the system. Developers looking to learn what functionality is provided by a unit and how to use it can look at the unit tests to gain a basic understanding of the unit's API.
* Design: When software is developed using a test-driven approach, the unit test may take the place of formal design. Each unit test can be seen as a design element specifying classes, methods, and observable behavior.

**6.3 INTEGRATION TESTING**

In integration testing stage, individual software modules (units) are combined and tested as a group. The interfaces between the different modules are tested. It occurs after unit testing and before system testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, tests those aggregates, and delivers as its output the integrated system ready for system testing. The purpose of integration testing is to verify functional, performance, and reliability requirements placed on major design items.

Some different types of integration testing are top-down, sandwich and bottom-up.

* Bottom-Up Testing is an approach to integrated testing where the lowest level components are tested first, then used to facilitate the testing of higher level components. The process is repeated until the component at the top of the hierarchy is tested.
* Top-Down Testing is an approach to integrated testing where the top integrated modules are tested and the branch of the module is tested step by step until the end of the related module. The main advantage of the Bottom-Up approach is that bugs are more easily found. With Top-Down, it is easier to find a missing branch link.
* Sandwich Testing is an approach to combine top down testing with bottom up testing.

Integration testing was carried out in incremental fashion. The testing format has been for the File Reader, Server and Visualization Window is described as follows

* The Main\_Window and Server objects were integrated and tested.
* The Visualization Window was integrated with the Server objected and tested.
* The File Reader object was integrated with Server object and tested.
* The File Reader, Server and Visualization Window objects were integrated and tested.

Similarly integration testing was carried out for all the modules of the software.

**6.4 VALIDATION**

Validation is the process of checking that a product, service, or system meets specifications and that it fufulfill its intended purpose in the desired environment. Validation is a Quality assurance process of establishing evidence that provides a high degree of assurance that a product, service, or system accomplishes its intended requirements. This often involves acceptance of fitness for purpose with end users and other product stakeholders. It is sometimes said that validation can be expressed by the query "Are you building the right thing?" and verification by "Are you building it right?" "Building the right thing" refers back to the user's needs; while "building it right" checks that the specifications are correctly implemented by the system.

Validation is defined as the process of testing the software against the requirements captured earlier. All the requirements have been mentioned in chapter 1 (section 1.4). The design traceability matrix provided in chapter 3 (section 3.2) provides a mapping of the requirements to the different modules of the software.

Validation process was carried out as follows

* Integration Testing
* System Testing

**6.4.1 INTEGRATION TESTING**

Integration testing was carried out in incremental fashion in the environment were the software is to be used. The testing format has been for the File Reader, Server and Visualization Window is described as follows

* The Main\_Window and Server objects were integrated and tested.
* The Visualization Window was integrated with the Server objected and tested.
* The File Reader object was integrated with Server object and tested.
* The File Reader, Server and Visualization Window objects were integrated and tested.

Similarly integration testing was carried out for all the modules of the software.

**6.4.2 SYSTEM TESTING**

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. As a rule, system testing takes, as its input, all of the "integrated" software components that have successfully passed integration testing and also the software system itself integrated with any applicable hardware system(s). It seeks to detect defects both within the "inter-assemblages" and also within the system as a whole. The software units that are integrated together are called assemblages.

In the final phase of, all the modules were integrated into a single and this system was tested as a whole for various input test cases in the desired environment. The system was then tested for flat file input that contained comma-separated, time-tagged values. Special test cases were generated that tested the full functionality of the system

**CONCLUSION**

A real-time satellite health monitoring tool was developed that takes input from a flat file database that contains comma-separated, time-tagged telemetry parameter values and visualizes the data in the form of text, graph and. Real-time data acquisition and delivery was simulated. The tool integrates textual display and graph display into a single entity called a visualization window. The tool is capable of displaying the graph that represents the variation of a desired telemetry parameter between the threshold values. The tool provides a feature of changing the threshold values of a parameter. The threshold values are updated in real-time. Threshold exceed event-based event description is another feature of the tool. The tool is capable of displaying the variation a desired parameter for a given period of time (T1 to T2). The tool has a flexible user interface. The main window displays desired sub-system parameter values. All the features of the tool were implemented and tested successfully.

**FUTURE ENHANCEMENTS**

The current version of the software has been developed based on the available time and schedule of the project. It can be enhanced by adding the following features

* Real-time acquisition of raw data and applying engineering unit conversion to generate processed parameter values.
* Real-Time archival of processed data
* Inclusion of tele-command functions along with telemetry functions.
* Event-based commanding for operational aspects of handling observed anomalies/events.

Real-Time acquisition and processing of raw data

The current version of the software acquires processed parameter values from a flat file database and visualizes the data in different formats. The software can be enhanced so as to acquire raw data from a shared memory interface and apply engineering unit conversion schemes to obtain processed parameter values as output. These values in turn can be visualized in different formats such as text, graph and 3D.

Real-time archival of processed data

The current version of the software acquires processed data from flat file and visualizes it depending on the user’s choice of parameter. If the same input is provided by the user, data has to be retrieved once again from the flat file. The software can be enhanced so as to acquire data and archive it in real-time. This archive can be used as a quick reference for data that is already processed .this eliminates the necessity of repeating the data retrieval process every time the user logs a request for the same processed value(s)[2].

Inclusion of tele-command functions

The current version of the software helps the user to interpret the status of a sub-system and in turn the satellite in orbit itself. It can be enhanced so as to provide options to the user to operate on sub-systems or payloads through tele-commanding. Satellite tele-commanding is the process of instructing a sub-system of a satellite in orbit to perform one or more operations or functions[3]. The user can make the satellite sub-system or payload perform a certain function

Eg: Switching on an on-board camera and capturing an image.

Event-based commanding

The current version of the software visualizes processed telemetry parameter values in different formats such as text, graph and 3D .The software initializes threshold values specific to each telemetry parameter. The threshold values can be altered by the user. Visualizing the values of the satellite sub-system parameters in different formats helps the user construct a better understanding of the health status of that sub-system and thus the satellite itself. The software can be enhanced so as to provide event-based tele-command functions. An event is generated every time the telemetry parameter value exceeds its threshold value. Tele-command function(s) can be defined for different events specific to each telemetry parameter to correct its value and thus maintain the health status of the sub-system and the satellite[3]. When an event is generated, the software identifies the appropriate measures to be taken to correct the telemetry parameter value and transmits them to the ground station antenna. The antenna in-turn will transmit to the satellite in orbit.

Eg:

Event: Fuel Tank 1 temperature (Propulsion sub-system) exceeds the lower threshold value

Action taken: A telecommand is transmitted to the satellite that instructs the thermal sub-system to switch on the heater nearest to fuel tank 1.Thus the parameter value is altered to a tolerable value.

**BIBLIOGRAPHY**

[1] Design and Prototyping of High Availability Architecture for Satellite Ground Control System, by In Jun Kim, Won Chan Jung, Myungja Kim, and Jaehoon Kim Communications Satellite Development Group, ETRI, 2005

[2] hiflyViews: New Generation Telemetry Visualization, by Juan-Carlos Gil, Thomas Morel, Luis Pastor GMV S.A Madrid, Spain, 2006

[3]Wireless Telecommand and Telemetry System for Satellite, by Satish Sharma, P.N.Ravichandran, Sunil, P.Lakshminarsimhan, R.Seshaiah Digital Systems Group, ISRO Satellite Center, Airport road, Bangalore-560017, India, 2007

[4]An Interactive Visualization System for Analyzing Spacecraft Telemetry ,by Cancro, G. Truner, R.Nguyen , L.Li, A. Sibol, D. Gersh, J. Piatko, C.Montemayor , J.Mckerracher, P. Johns Hopkins Univ., Hopkins, 2007

[5] Real-Time Monitoring Of the Chang’E-1 Lunar Orbit Insertion, byMingyuan Wang Xian Shi Nianchuan Jian Rui Yan Jinsong Ping   
Shanghai Astron. Obs., Chinese Acad. of Sci., Shanghai, China, 2009

[6] Three-Dimensional tools for Telemetry Visualization and Analysis, by John R. Wright, Jet Propulsion Laboratory, California Institute of Technology, 2009.

[7] Spacecraft telemetry data monitoring by dimensionality Reduction Techniques, by Yairi, T. Inui, M. Yoshiki, A. Kawahara, Y. Takata, N. Sch. of Eng., Univ. of Tokyo, Tokyo, Japan, 2010.