The Space Department of
The Johns Hopkins University
Applied Physics Laboratory

SOFTWARE DEVELOPMENT PLAN for the T I M E D

GLOBAL ULTRAVIOLET IMAGER -GUVI- DATA PROCESSING PAYLOAD OPERATIONS CENTER

Document: 7366-9201

Revision B

February 6, 1998

Authors: Michele Weiss
Kevin Pettee

Signature Page

GUVI Software Quality Assurance Engineer	Harry Utterback
Project Scientist	Larry Paxton
GUVI Lead Engineer	Bernard Ogorzalek
GUVI POC Lead Software Engineer	Michele Weiss
CPI Lead Software Engineer	Scott Evans
GUVI Program Manager	Tom Pardoe

Table of Contents

1. Scope

1.1. Identification

This document describes the software development plan for the TIMED Global Ultraviolet Imager (GUVI) Data Processing (DP) Payload Operations Center (POC), hereafter referred to simply as "GUVI DP POC".

1.2. System Overview

The Thermosphere, Ionosphere, Mesosphere Energetics and Dynamics (TIMED) program is a NASA funded mission whose development and operations are being managed by Johns Hopkins University Applied Physics Lab (JHU/APL). The development of the GUVI DP POC will be a combined effort between JHU/APL and Computational Physics, Inc. (CPI). Details of the GUVI DP POC itself are found in GUVI POC Software Requirements Specification (Reference 6).

The GUVI DP POC is being developed by JHU/APL and CPI and will be located at the JHU/APL facility for use by TIMED instrument and interdisciplinary investigators, mission operations, K-12 educators and the public at large. The GUVI DP POC is responsible for:

- Routine data product generation
- Data access and distribution

These functions will be developed as part of the Data Processing POC and will be developed by JHU/APL and CPI with JHU/APL acting as the lead engineering organization.

1.3. Document Overview

This document describes the plans for the various software development and maintenance activities required for the development and operation of the GUVI DP POC. It includes information about the software development methodology employed, quality assurance planning information, resource requirements and schedules.

1.4. Relationship to Other Plans

This plan has been developed as required by TIMED Software Quality Assurance Plan (Reference [1]), and according to the standards set by the Space Department Software Quality Assurance Guidelines (Reference [2]).

The structure of this document and the strategy for software development has been adapted principally from MIL-STD-498 (Reference [3]).

2. Referenced Documents

Reference [1] **TIMED Software Quality Assurance Plan**, JHU/APL 7363-9101, September 1996.

Reference [2] **Space Department Software Quality Assurance Guidelines**, SDO-9989, Sept. 1992.

Reference [3] MIL-STD-498, **Military Standard Software Development and Documentation**, AMSC No. N7069, Dec. 1994.

Reference [4] **Object Oriented Design with Applications**, G. Booch, 1991.

Reference [5] **TIMED System Requirements Document**, JHU/APL 7363-9001, Jan. 1997.

Reference [6] **TIMED GUVI POC SRS**, JHU/APL 7366-9200, Jan. 1998.

Reference [7] **GUVI Software Quality Assurance Plan**, JHU/APL 7366-9003, Apr. 1997.

Reference [8] **TIMED General Instrument Interface Specification, Section 8.0, Ground System and the Payload Operations Centers (POC's) Interface,** JHU/APL 7363-9050, Nov. 1997.

3. Overview of Work Required

The GUVI DP POC is partitioned into the two components shown in Table 1 which are defined in Section 3.2 of the GUVI POC Software Requirements Specification (SRS), Reference 6. Top level partitioning into these components is done to match the organizational structure of the SRS and to improve readability. This does not imply a specific design or data flow but an organizational structure for this document.

Each of these components has different attributes and as such, will be required to meet different standards (Table 1 of Reference 1).

Component Number	Component Name	Attributes
1	Routine Data Product Generation	Minimum Level
2	Data Access & Distribution	Minimum Level

Table 1. Component Software Quality Assurance Levels

None of the components are viewed as involving moderate technical risk, requiring a large development team, multiple organizations, or a long lifetime.

3.1. Component 1: Routine Data Product Generation

The Routine Data Product Generation component retrieves the telemetry data, reformats, performs algorithmic processing on the data and generates the data products and is not considered mission impact. For this reason, the Routine Data Product Generation component is considered to be at the minimum level of criticality as defined in Reference 1.

3.2. Component 2: Data Access & Distribution

The Data Access and Distribution component generates survey products and a catalog as well as distributing the data to the users, to the MDC and to an archive and is not considered mission impact. For this reason, the Data Access and Distribution component is considered to be at the minimum level of criticality as defined in Reference 1.

3.3. Schedule Constraints

The GUVI DP POC development must be coordinated with the rest of the GUVI and TIMED program. The Data Processing POC will be delivered shortly before spacecraft launch.

An overview of the GUVI project schedule is shown in section 0.

3.4. Some Software Development Terms Defined

The following terms are used in this document to describe the GUVI DP POC software development process. To prevent ambiguity and possible confusion with the terms' connotation in other contexts, they are defined in Table 2.

Term	Definition	Examples of use
System	The set of all related software.	GUVI Data Processing POC
Component	Top-level partition of the software system.	Routine Data Product Generation, Data Access & Distribution
Build	The executable file(s) yielded by compiling a specified set of modules; may apply to a given Component, or to the entire system.	Prototype build 1 of the Data Processing POC
Module	Lower-level group of logically related classes, assembled to achieve a function.	Science Algorithms, Data Reformatter
Unit	Lowest level partition; an independently testable function, procedure or task.	function WriteData - called to output disk Level 1C data

Table 2. Software Development Terms

4. Plans for Performing General Software Development Activities

4.1. Software Development Process

Over the course of the TIMED mission and the GUVI program, there will be five builds of the GUVI DP POC software. Scheduling of these builds is described in Section 6 of this document. These builds will satisfy the requirements from Integration and Test through extended mission and are described as follows:

Prototype Builds: The requirements for the Data Processing POC user interface will be refined through a series of interim prototype builds. These builds will occur on an as needed basis and are not scheduled for specific times. The development process for these builds will consist of internal reviews and presentations through a series of Technical Interchange Meetings (TIMs).

Launch Ready Build: This build will be delivered three months before the TIMED spacecraft launch. During this build, the Data Processing POC will be delivered for C/D functionality.

Post-Launch Build: This build will be delivered shortly before the end of the TIMED mission. During this build, the Data Processing POC will be delivered for mission operations/data acquisition (MO/DA) functionality which includes planned enhancements of the GUVI data products, GUVI catalog and the GUVI survey products.

Post-Mission Build: This build will be delivered after the end of the TIMED mission. During this build, the Data Processing POC will be delivered for extended mission functionality that includes additional enhancements in the GUVI scientific algorithms and survey products.

4.2. General Plans for Software Development

The overall plan for system development will have an initial period of conceptualization, planning and preparation in which it will be decided which software to use for development and which hardware will need to be purchased. This will be followed by a software specification where the software requirements are gathered. This will be completed prior to the start of software development. In addition, a series of Prototype builds will also be utilized to aid in the user interface definition and planning phase.

4.2.1. Software Development Methods

Software will be developed according to the life-cycle models and detailed software development activities described in Section 5.

4.2.2. Standards for Software Products

Software system development will follow a methodology of planning, requirements analysis, design and testing. All of this will be documented according to section 4.2.6.

Software headers will be located at both the file level and at the function, procedure, class or task level.

File headers will, at a minimum, contain the following information:

1. the filename
2. the purpose
3. design assumptions and constraints, as appropriate
4. notes
5. change history which will include date, description of change and person making change

The change history will contain the following information:

1. date of change
2. name of developer making the change
3. brief summary of the change(s) made

Function, procedure, class or task level headers will, at a minimum contain the following information:

1. the filename (if function is in a file with no file header, i.e. a separate)
2. the purpose
3. a description of the inputs and outputs
4. a description of exceptions and/or errors generated
5. design assumptions and constraints, as appropriate
6. notes
7. change history (if function is in a file with no file header)

Reusable software that has been modified more than 30 percent must contain the header information defined above for files, functions, procedures, classes and tasks. Reusable software that has been modified less than 30 percent will, at a minimum contain the following information:

1. commented source code changes

In addition, all source code must be well organized and formatted. Although we do not adopt a single coding standard here, all software will be reviewed and must be deemed suitable for maintenance. Details of the code review process can be found in Section 5.7.1.

4.2.3. Reusable Software Products

4.2.3.1. Incorporating Reusable Software Products

During the preliminary design phase for each component, candidate reusable software shall be identified for evaluation. Information about software used for previous builds as well as other programs (especially SSUSI developed for DMSP Block 5D-3 satellites) shall be collected. All reusable software is subject to the same acceptance criteria as the rest of the system.

4.2.3.2. Developing Reusable Software Products

To the extent that is reasonable, software developed for the GUVI DP POC should be reusable in later releases of the software. There is no requirement to identify or develop software that is reusable for projects other than GUVI DP POC, however good programming practices will be applied to aid future projects.

4.2.4. Handling of Critical Requirements

4.2.4.1. Safety Assurance

There are no safety requirements for the GUVI DP POC.

4.2.4.2. Security Assurance

The security requirements for the GUVI DP POC are as follows:

1. Protection from an active intrusion that disrupts operations
2. Protection from an active intrusion that modifies or falsifies data products
3. Protection from a passive intrusion that acquires data without authorization
4. Protection from inadvertent unauthorized data release

For each component, a security risk analysis should be performed.

4.2.4.3. Privacy Assurance

There are no critical privacy requirements.

4.2.4.4. Assurance of Other Critical Requirements

4.2.4.4.1. Spacecraft Safety

There are no spacecraft safety requirements.

4.2.5. Computer Hardware Resource Utilization

The hardware resource requirements will be presented in the GUVI DP POC Software Design Document.

4.2.6. Documentation and Review Plan

The GUVI DP POC software is composed of several components that will be developed by APL and CPI. Each organization will be required to contribute to the formal system-level documentation. Where the software development, integration, and testing is shared between organizations, the documents describing those activities and results can also be shared. The operations of the DP POC and the User Interface shall be documented in User’s Manual(s) and in addition, the Software Test Plan (STP) and the Software Test Description (STD) can be merged into a single software test document.

Tables 3 and 4 present a matrix of the system level documentation and reviews that are required. Unless otherwise noted, the documents listed below are described in Reference [3]. The documents listed in Table 4 can be combined in any manner and the exact documents specified are meant as guidelines for the technical information that needs to be documented.

System-Level and Component Reviews		Responsible Organizations	
		APL	CPI
	Conceptual Design Review (CoDR)	X	
	Instrument Preliminary Design Review (PDR)	X	
	Software PDR (SPDR)	X	X
	DP POC Critical Design Review (DP CDR)	X	X
	TIMs (as needed)	X	X
	Consent to Ship (CSR)	X	X
Data Product Generation	Design Walk-through	X	X
	Code Walk-through	X	X
Data Access & Distribution	Design Walk-through	X	
	Code Walk-through	X	

Table 3. GUVI DP POC (and Component) Review

Documents	Responsible Organizations	Deliverable by		
		SPDR	DP CDR	CSR
SW Development Plan	APL	X		
SW Requirements Specification	APL	X	X	X
Operational Concept Document	APL	X	X	X
SW Design Document	APL, CPI		X	X
Language Independent Descriptions (LIDs)	APL, CPI		X	X
SW Test Plan	APL, CPI	X	X	X
SW Test Description	APL, CPI		X	X
SW Version Description	APL, CPI			X
User's Manual	APL, CPI			X

Table 4. TIMED GUVI DP POC Publications

4.2.7. Access for Sponsor Review

NASA will have full access to all development facilities.

5. Plans for Performing Detailed Software Development Activities

5.1. Project Planning and Oversight

5.1.1. Software Development Planning

Management, monitoring, and quality control functions relevant to the software development process begin early in the design phase, where fundamental decisions regarding architecture, fault tolerance, and availability of the system are made. As development continues through the coding, testing, and integration phases, management is a continual, daily process, punctuated and effectuated by the series of technical and management reviews listed in Table 3.

5.1.2. Test Planning

The STP will describe the various levels of testing to be performed on each major component of the DP POC.

Unit test procedures can either be code to test the software unit or a plan on how the unit will be tested, as appropriate. Unit test procedures will be presented as part of module design and will be required at each code walk-through; see section 5.7.2. Unit test procedures consist of anything necessary to test out the source code and can be drivers, debug statements embedded into the source code, inputs/outputs for testing by demonstration, etc.

5.1.3. Following and Updating Plans, Including Intervals for Management Review

Each of the reviews indicated in Table 3 is subject to the corrective action process described in section 5.17.

5.2. Establishing a Software Development Environment

5.2.1. Software Engineering Environment

Preliminary analysis indicates that the algorithmic portion of the Routine Data Product Generation component will be developed in Ada 95 on a Silicon Graphics workstation with the remainder being developed in C++ on a UNIX workstation. The user interface portion of the Data Access & Distribution component will be developed primarily in Java with the remainder being developed in C++ on a UNIX workstation. Other options may be considered, evaluated, and incorporated, e.g., using Perl, HTML, CGI, etc. and

5.2.2. Software Development Folders

Software development folders (SDFs) will be maintained for each component and each module within the components. SDFs, at a minimum, should contain source code, design and code walk-through packages, action items, developer notes, test data, unit test programs, and makefiles. SDFs can consist of a combination of paper files and electronic files.

The SDFs are subject to inspection at any time. Software developers will be responsible for the maintenance of SDFs for their respective modules or components.

5.2.3. Non-deliverable Software

Non-deliverable software such as drivers for unit test procedures, software stubs, etc. will not be kept under configuration management (CM) control. However, non-delivered software for testing will be kept under CM and will physically reside in a separate location from source code. Non-deliverable software used for development that is of general utility may also be kept under CM control and will physically reside in a separate location from source code.

5.3. System Requirements Analysis

5.3.1. User Refinements of System Requirements

System requirements will be collected through interviews with the principle users of the GUVI DP POC (GUVI mission operations, the science teams, etc.). TIMED mission level requirements are documented in Reference 5 and GUVI software requirements are documented in Reference 6.

The GUVI DP POC will be developed incrementally according to the build schedule defined in section 4.1. In addition, the Prototype builds and the TIMs will be utilized as mechanisms to aid in the refinement of the system requirements.

5.3.2. System Requirements

The system requirements are formulated and published utilizing the SRS DID contained in Reference [3]. A final SRS will be reviewed at the GUVI POC SPDR and at the GUVI DP POC CDR.

5.4. System Design

5.4.1. System-Wide Design Decisions

System design decisions affecting GUVI requirements, the development schedule, and/or functioning will be recorded in the relevant requirements and design documentation. These documented decisions will be reviewed at SPDR, DP CDR, and Technical Information Meetings (TIMs).

5.4.2. System Architectural Design

The system architecture will be investigated with respect to hardware and software and will be presented as part of the GUVI POC SPDR and DP CDR.

5.5. Software Requirements Analysis

Software requirements will be formulated and published in the GUVI POC SRS (Reference 6). The SRS will be presented as a draft document for the GUVI instrument PDR and as formal documents for the subsequent SPDR and DP CDR.

GUVI DP POC external interfaces will be documented in the GIIS (Reference 8). GUVI DP POC internal interfaces will be documented in the GUVI DP POC Software Design Document (SDD).

5.6. Software Design

5.6.1. Component-Wide Design Decisions

Component-wide design decisions will be documented in the SDD and presented as part of the GUVI POC SPDR, DP CDR and TIMs.

5.6.2. Component Architectural Design

The component architectural design will be formulated and published in the SDD.

5.6.3. Component Detailed Design

Component detailed designs will be formulated and published using the SDD DID contained in Reference [3]. The SDD will be presented as a formal document for DP CDR.

Component detailed design will be presented in design walk-throughs. After the system has been designed and the design has been accepted, individual developers will code and write unit test procedures which will be presented in code walk-throughs.

5.7. Software Implementation and Unit Testing

5.7.1. Software Implementation

During the design phase, and before coding, the implementation will go through a design walk-through. The design for each module will be presented through any combination of Program Design Language (PDL), flow charts, data flow diagrams, context diagrams, etc. After being reviewed by the respective teams, this serves as the basis for coding. That is, the software units will be coded according to the requirements, constraints, logic, control flow, and other parameters specified in the design walk-through. Problems found with the SDD during the coding phase will require the use of the corrective action process, which allows corrections and improvements to be reviewed in an orderly manner (see section 5.17). After coding and prior to testing, the implementation will go through a code walk-through. For a code walk-through the source code must be compiled with no errors.

All members of the development team may attend code walk-throughs, but they should comprise the following: software lead, component lead, and developer. The following items will be reviewed during the code walk-through:

1. Fulfillment of all requirements associated with that module
2. Adherence of the code to the documented design
3. Adherence of the code to reasonable coding standards
4. Correctness of programming logic
5. Adequate test cases and data proposed for unit testing.

Reusable software that has been modified more than 30 percent must have design and code walk-throughs. Reusable software that has been modified less than 30 percent are not required to be reviewed and only the software changes should be unit tested.

5.7.2. Unit Testing

Unit testing will be the sole responsibility of the individual developer. The test method and test cases used by the developer to verify correct functioning will be reviewed at the code walk-through. Test cases and procedures should be recorded in the appropriate SDF.

Critical units may require performance profiling as defined at the design phase to ensure performance requirements can be met.

Each development team member should perform unit tests according to the plan recorded in the appropriate SDF. Changes in the data or procedures should be recorded in the SDF as well. Developers are responsible for revision and re-testing. Re-testing should be done according to the plan described in the SDF. Results of unit tests should be recorded by the developer in the appropriate SDF. Following an unsuccessful test, the result should be reported to the component lead engineer. Minor changes to the software being tested or the test data/procedures can be made by the developer and re-tested. Significant changes will require a new code walk-through. The component lead engineer will determine whether a change is minor or significant.

5.8. Unit Integration and Testing

Following unit testing as performed by the software developer, the code will be delivered and integration testing will be performed according to the STP and the STD. Failed tests will be reported to the component lead engineer for appropriate action.

5.9. Component Qualification Testing

5.9.1. Independence in Component Qualification Testing

Developers who have significantly participated in the design or implementation of a component should not be assigned to carry out qualification testing for that component. These developers, however, may be asked to provide test data or technical assistance in carrying out the test.

5.9.2. Preparing for and Performing Component Qualification Testing

The component to be tested shall be documented in the STD. This will provide traceability of the tests to be conducted with the requirements allocated to the component in the SRS. The assigned developer will perform component testing according to the STD.

Following an unsuccessful test, the component will be returned to the component developers for revision and a software trouble report (STR) will be generated. Upon completion of this revision, the software will be re-tested and placed in the SDF and a new code/design walk-through will be scheduled if the change required is significant.

Upon completion of successful testing, the developer responsible for the testing will close the STR.

5.10. Component/Hardware Integration Testing

No specialized hardware components are expected to be developed or required for the GUVI DP POC.

Suitably constructed component qualification testing will assure that the software functions correctly on the target hardware.

5.11. System Qualification Testing

Only the final releases of the GUVI DP POC system software will undergo formal system qualification testing. Initial unit testing will be independent and informal; tests will become formalized and made available to the integration team at the direction of the GUVI POC lead engineer. This test plan is illustrated schematically in Figure 1.

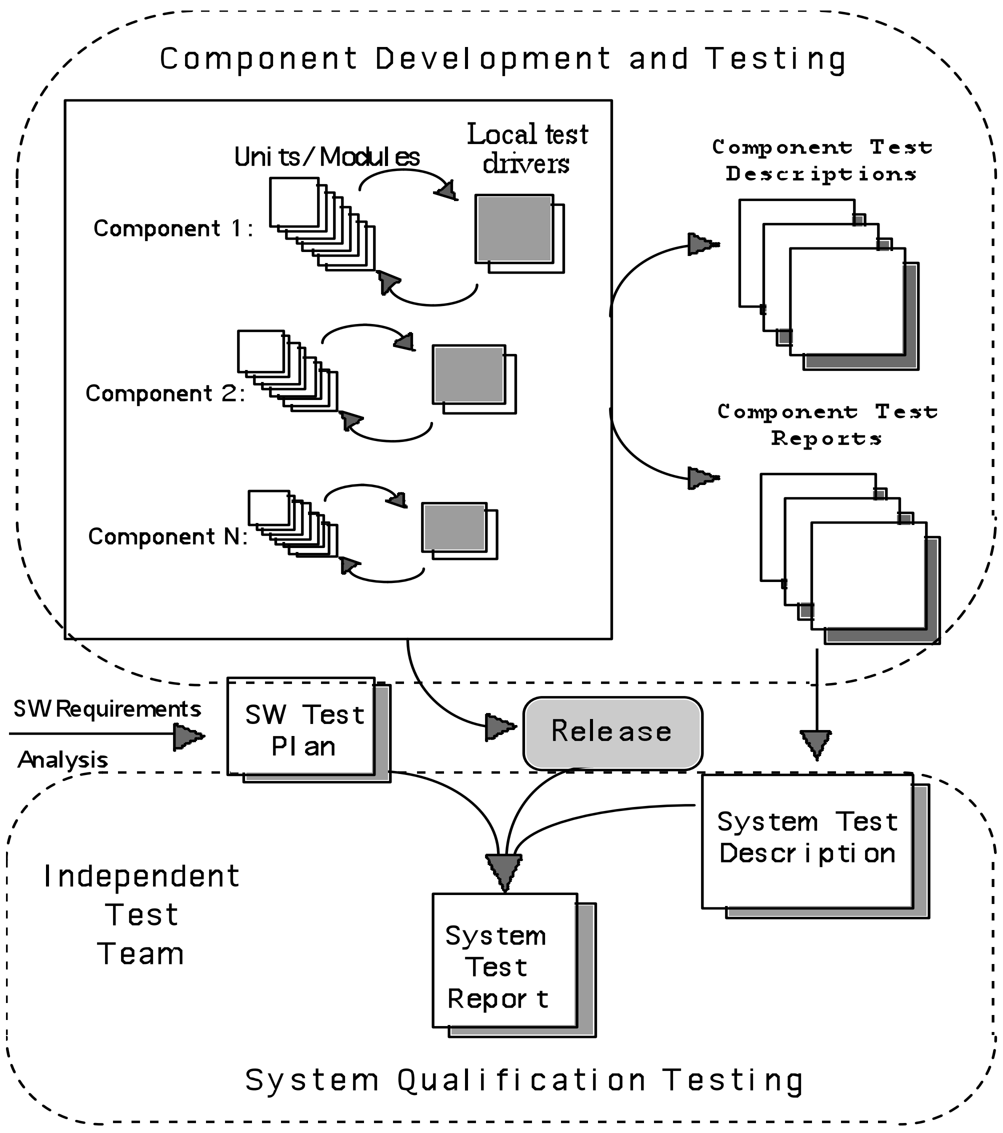

Figure 1. Unit, Component, and System Level Testing

5.11.1. Independence in System Qualification Testing

System qualification testing should not be conducted by members of the software development team. An independent test team (ITT) should be assembled to plan and carry out system testing. This team should have full access to the development team for technical assistance and the generation of data. However, because of inadequate funding the test team will probably consist of the GUVI DP POC software developers.

5.11.2. Testing on the Target Computer System

All system tests will occur under operational conditions. These will be further discussed in the STP after requirements have been defined.

5.11.3. Preparing for and Performing System Qualification Testing

An ITT should be formed to perform system qualification. Both the selection process and the results will be documented in a memo to the Program Manager. The ITT should perform system qualification test activities according to the STD.

The ITT will be responsible for the production of STRs using the results of the system qualification test.

5.12. Preparing Software for Use

5.12.1. Preparing the Executable Software

The system qualification (formal and informal) will be tested in place, and therefore will be operational after it is tested. No additional preparation will be needed.

5.12.2. Preparing User Manuals

Software User Manual(s) will be prepared to document the user interface elements of the DP POC as well as the operations of the DP POC and can consist of one or several separate documents. These manuals will be reviewed by the GUVI POC operations staff, and will serve as a basis for their training and will be delivered for each of the builds of the DP POC software as needed.

5.13. Preparing for Software Transition

There will be no formal transition of the GUVI DP POC system to a support agency. However, in order to simplify software maintenance and operation, the quality, accuracy, and timeliness of the GUVI DP POC SDD, STD, and STP will ensure they are sufficient for use as software maintenance documentation.

5.14. Software Configuration Management

5.14.1. Configuration Identification

All deliverable software code and documents will be identified and controlled by a configuration management process.

5.14.2. Configuration Control

All software source files and make files used to create the production system will be placed under CM control. An automated CM tool will be utilized to provide CM control.

Appropriate setup files, static data files and project documentation will also be placed under CM control. Each item will be identified by its name, number and revision number. The CM system will have the capability to track versions and releases of the software and provide file revision security.

5.15. Software Product Evaluation

5.15.1. In-Process and Final Software Product Evaluations

The TIMED project will oversee in-process and final software product evaluations that will be conducted as part of GUVI POC SPDR, DP CDR, and TIMs.

5.16. Software Quality Assurance Evaluations

Software quality assurance (SQA) evaluations should be performed. SQA should insure that each activity appearing in this document is being performed, and that all software products are undergoing suitable validation, testing and corrective action. SQA should maintain records of these evaluations in the Software Development Folders.

5.17. Other Software Development Activities

The GUVI POC lead engineer will issue reports which addresses risk. All component Lead Engineers will issue reports to the GUVI POC lead engineer, also addressing risk, as explained below.

5.17.1. Risk Management

System and software risk areas will be identified and documented during the cycles of development. The GUVI POC lead engineer will maintain this throughout the project that records the risk areas, mitigation strategies, implementation tradeoffs and ultimate results of implementation. Risk areas will be maintained until they are resolved and verified by the component lead developer.

5.17.2. Improvement of Project Processes

The GUVI POC lead engineer will periodically assess the processes used on the project to determine their suitability and effectiveness. Based on these assessments, necessary and beneficial improvements will be identified and proposed, and implemented if approved.

6. Schedules and Activity Network

The GUVI POC lead engineer is maintaining the project schedule in Microsoft Project. Below is a summary of the baseline schedule. More detailed project schedules will be presented at SPDR, DP CDR and TIMs.

1997: Jan ...Feb ...Mar ...Apr ...May ... Jul... Nov ...Dec

▯ Instrument PDR — ▯ Software PDR

1998: Jan ...Feb ...Mar ...Apr ...May ... Jul... Nov ...Dec

▯ Instrument CDR — ▯ DP POC CDR

7. Project Organization and Resources

7.1. Project Organization

Figure 2 depicts the project organization for the GUVI POC system development effort.

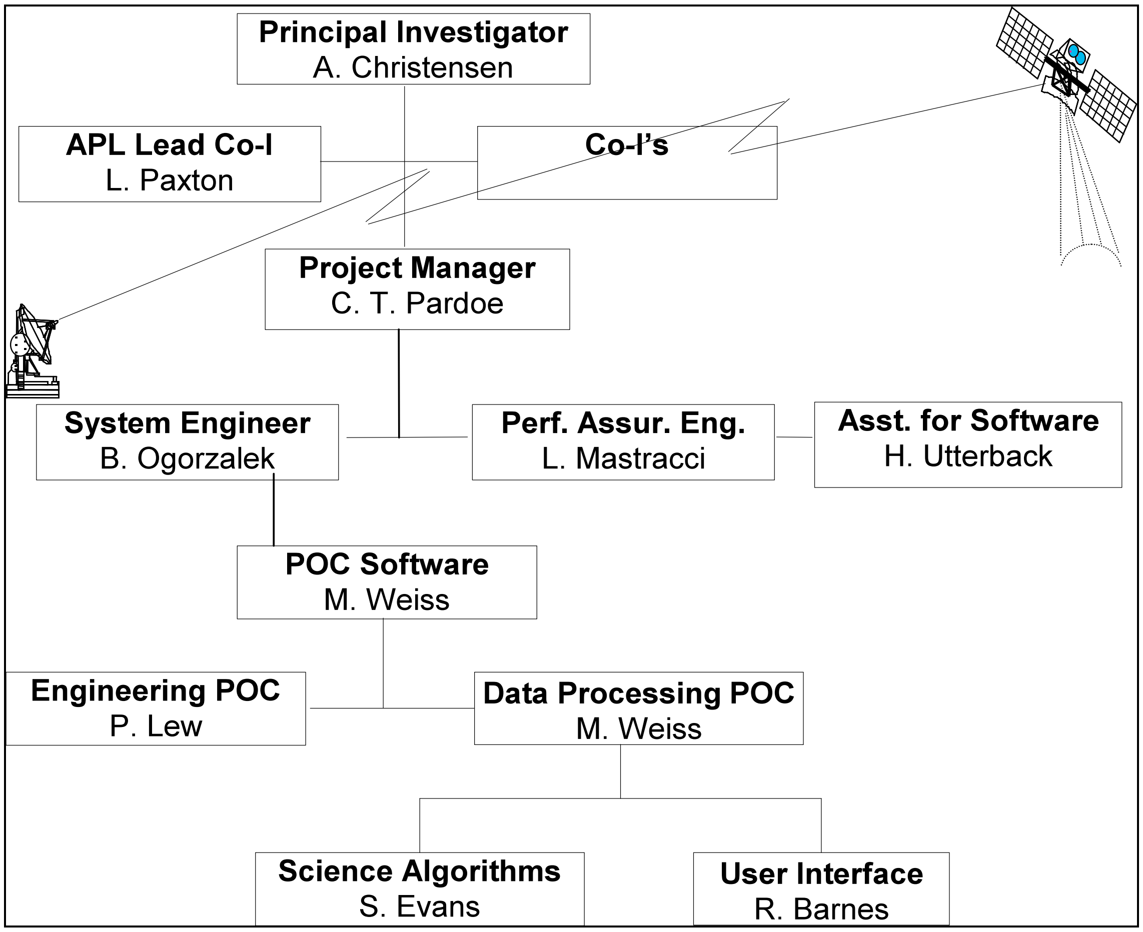

Figure 2. GUVI POC Development Organization

8.

Notes

8.1. GUVI Project Schedule

This section is included for informational purposes only. These are the planned key dates for several TIMED program milestones.

Software PDR	May-97
Instrument CDR	Jan-98
DP POC CDR	Mar-98
GUVI I&T	July-98
Delivery to S/C	Jan-99
GUVI S/C I&T	Mar-99

Table 5. GUVI Project Schedule

8.2. Acronyms

APL	The Johns Hopkins University/Applied Physics Laboratory
CCB	Configuration Control Board
CDR	Critical Design Review
CM	Configuration Management
CoDR	Conceptual Design Review
CPI	Computational Physics, Inc.
CSR	Consent to Ship Review
DID	Data Item Description
DMSP	Defense Meteorological Satellite Program
DP	Data Processing
GSE	Ground Support Equipment
GUVI	Global Ultraviolet Imager
HW	Hardware
I&T	Integration and Test
ICD	Interface Control Document
ITT	Independent Test Team
JHU/APL	See APL
LIDs	Language Independent Descriptions
MO/DA	Mission Operations/Data Acquisition
NASA	National Aeronautics and Space Administration
OCD	Operational Concept Document
PDL	Programming Design Language
PDR	Preliminary Design Review
POC	Payload Operations Center
RMR	Risk Management Report
SDD	Software Design Document
SDF	Software Development Folder
SDP	Software Development Plan
SPDR	Software Preliminary Design Review
SQA	Software Quality Assurance
SRS	Software Requirements Specification
SSUSI	Special Sensor Ultraviolet Spectrographic Imager
STD	Software Test Description
STP	Software Test Plan
STR	Software Trouble Report
SUM	Software Users Manual

SVD	Software Version Description
TIM	Technical Interchange Meeting
TIMED	Thermosphere Mesosphere Energetics and Dynamics experiment
URL	Uniform Resource Locator
WWW	World Wide Web

End of GUVI POC Software Development Plan