170-WP-013-001

Catalogue Interoperability Protocol (CIP) Prototype Development Plan

White paper - Not intended for formal review or Government approval.

January 1997

Prepared Under Contract NAS5-60000

RESPONSIBLE ENGINEER

A. V. Gupte /s/ 1/6/97

Ajay V. Gupte, Senior Systems Engineer Date
EOSDIS Core System Project

SUBMITTED BY

Gge Percivall /s/ 1/6/97

George S. Percivall, Lead Systems Engineer Date
EOSDIS Core System Project

Hughes Information Technology Systems
Upper Marlboro, Maryland

This page intentionally left blank.

Abstract

This document defines the Catalog Interoperability Protocol (CIP) Prototype Development Plan and its components. The components that are integrated to construct the prototype are identified and described. Most of the components will be taken from existing software with the goal of maximizing software reuse. However, the Prototype also requires certain modules and components which do not currently exist and these are identified. The hardware and software that will be used for the development is described as are other resources necessary for the Project. The Plan also includes the milestones and the risks that are involved in completion of the Prototype.

Keywords:

Prototype, CIP, Development, Gateway, Translator.

This page intentionally left blank.

Contents

1. Introduction

1.1 Purpose 1-1

1.2 Organization 1-1

1.3 Review and Approval 1-1

2. Development Plan

2.1 Task Priorities 2-1

2.1.1 Re-host CIP-A Demonstrator 2-1

2.1.2 Translator Design and Implementation 2-1

2.1.3 Reflect ECS Data Model in CIP Collections 2-1

2.1.4 ECS to CIP Gateway 2-2

2.1.5 CIP-B Additions & Modifications 2-2

2.1.6 JEST/CIP Client 2-2

2.2 Development Resources 2-2

2.2.1 Hardware Resource 2-3

2.2.2 Software Resources 2-3

2.3 CIP Components 2-3

2.3.1 CIP Components 2-3

2.3.2 Description of CIP Prototype Components 2-5

2.4 Key Risks 2-7

2.4.1 Resource Risks 2-7

2.4.2 Re-hosting Risks 2-8

2.4.3 Implementation Risks 2-8

2.5 Milestones and Deliverables 2-9

2.5.1 Objective Milestones 2-9

2.5.2 Deliverables Time Frame 2-10

2.6 Prototype Demonstration 2-11

Abbreviations and Acronyms

List of Figures

2-1. CIP-Prototype Components.....2-4

2-2 Data Communication Stack for the CIP/ECS Catalogue Translator.....2-7

1. Introduction

1.1 Purpose

The prototype development is driven by two primary objectives as defined in ESD #25. The first objective is to demonstrate the feasibility of the concept of a Catalogue Translator. The second objective is to prove the feasibility of a CIP (extended Z39.50) based ICS prototype (or CIP-Prototype) while reusing as many existing components as possible and conforming to HITS Software Engineering methods. This document defines the components that currently exist, although by no means does it limit the use of "newly found" software.

In this document, we will first define the priorities in the iterative development of the components that make up the complete CIP-Prototype. We will identify the component that will be built at various stages of the project. We will then show the integration plan for bringing together all the required components into a cohesive whole with supporting milestone and risks.

The lessons learned from the Prototype, including those learned through the development phase, will be incorporated into the CIP-Release C Specification, to make it more compatible.

This document was developed with input from Yonsook Enloe of ESDIS and George Percivall, Ananth Rao and Eric S. Martin of HITS.

1.2 Organization

This paper is organized as follows:

- The task priorities are listed and described. (Section 2.1)
- The Resources available to the project are defined. (Section 2.2)
- The CIP Components are presented (Section 2.3)
- The Key Risks and Milestones are marked and defined. (Sections 2.4, 2.5 & 2.6)

1.3 Review and Approval

This White Paper is an informal document approved at the Office Manager level. It does not require formal Government review or approval; however, it is submitted with the intent that review and comments will be forthcoming. Further discussion based on this paper shall result in Requirements and Design Document which will be used for the actual Prototype Development.

The ideas expressed in this White Paper are valid from January 1997 through April 1997. Questions regarding technical information contained within this Paper should be addressed to the following ECS and/or GSFC contacts:

- ECS Contacts
  - Ajay V. Gupte, Senior Systems Analyst, 301-883-4178, agupte@eos.hitc.com.
- GSFC Contacts
  - Yonsook Enloe, ESDIS, 301-614-5122, yonsook@harp.gsfc.nasa.gov

Questions concerning distribution or control of this document should be addressed to:

Data Management Office
The ECS Project Office
Hughes Information Technology Systems
1616A McCormick Drive
Upper Marlboro, Maryland 20774-5372

2. Development Plan

2.1 Task Priorities

This section describes the priorities and the task order that must be met for the completion of the CIP prototype. Sections 2.1.1 through 2.1.6 are ordered in descending order of task priority. Refer to Figure 2-1 for component diagram.

2.1.1 Re-host CIP-A Demonstrator

The CEO CIP-Release A Demonstrator (CIP-AD) developed by CEO/Logica, based on the CIP-A specification, shall be re-hosted to the ECS Rational Apex/C++ environment. The components to be re-hosted are the Retrieval Manager and the WWW2CIP Gateway. The development platform used for the CIP-Demonstrator is not currently known and could result in compilation problems in a functional program source code. From most indications the Demonstrator was developed using the GNU public domain tools and will suffer mismatch problems. The GNU compilers are very forgiving and several compilation errors are only flagged as warnings in the GNU system. At times some of the computations result in erroneous solutions. The rational Apex/C++ environment is a commercial development environment which will clearly mark the problem areas. The Rational Rose product will be employed to reverse engineer the Demonstrator and in identifying the objects and classes that make up the software.

Reference: CIP-AD Systems Requirement Document (LUK.503.EC.22426/SR002) and CIP-AD ADD (LUK.503.EC.22426/AD001)

2.1.2 Translator Design and Implementation

The CIP to ECS translator as shown in Figure 2-2 will be based on CIP/RBS Demonstrator developed by ESA/SERCO. The CIP/RBA Translator component of the Demonstrator was developed on the Sun SPARC station-4 running SunOS 5.5 (Solaris 2.4), using GNU gcc 2.7.2, gdb-4.16. The SIL Z39.50 libraries that are used are the unofficial DBV-OSI version 1.9.3. The software was developed using the SCCS version control software. This component is known to have used the GNU Tools and will suffer from problems indicated in 2.1.1. Other components, shown in Figure 2-2. in the later sections, can be reused with modifications to the interface layer. The other aspect that will be designed will encompass the communication of CIP-Server with the ECS database. Existing classes and libraries developed for the Wisconsin Prototype can be reused without complex modifications.

2.1.3 Reflect ECS Data Model in CIP Collections

The Translator Design and Implementation also requires clear identification and reflection of the Collection schema within the database to translate the interaction between CIP and the existing ECS. The core section of the Translator, as defined in the earlier sections involves conversion of the CIP inquiries, the negotiation, the queries and data handling, into ECS inquiries. This

component would require, the mapping of all the requests between the CIP attributes and the ECS attributes. The task will include the data engineering necessary to map the CIP schema to the ECS schema, to define a CIP-collection structure to represent ECS collections, and to physically obtain access to ECS collections through an existing Data Server Prototype built for the ECS Evaluation package (EP 7). The task involves considering the automation of populating the databases in the Retrieval Manager.

2.1.4 ECS to CIP Gateway

The Prototype is also required to pass the inquiries from the ECS clients through ECS subsystems, e.g. DM, to the CIP Retrieval Managers, mandating the translation from ECS inquiries to CIP inquiries. This ECS to CIP Gateway does not exist and must be designed and implemented. The Data model in 2.1.3 will need to be extended to a two-way data model to allow the conversion of the inquiries from ECS to CIP and allow for passing of resulting data to the ECS from the CIP. The concept of Local Information Manager (LIM) and Distributed Information Manager (DIM) described in 442-TP-001-001 by Lynne M. Case will be reviewed during the design of this gateway. As this task need not be demonstrated at the April C.E.O.S. Meeting and to allow concentration on the first three priorities, this task will be delayed beyond April.

2.1.5 CIP-B Additions & Modifications

The CIP-Prototype would involve the upgrading of all the Demonstrator software to conform to the CIP-B specification and the removal of RBA specific code and insertion of ECS modules. Certain risks are involved in this phase. The Additions and Modifications between the CIP-A and CIP-B must be identified before implementing the changes in the CIP-Prototype. The CIP-B Specification is not yet available and this task will be postponed to later versions of the prototype.

Other additions may include the ordering handling beyond the functionality developed for the CIP-Demonstrator. It will be used to demonstrate the functionality of the system with attributes defined in the CIP-B Specification.

2.1.6 JEST/CIP Client

The modification of the Java Earth Science Tool (JEST) to include CIP Client application will be left for later development and addition to the suite of clients which would access the CIP based Retrieval Managers.

2.2 Development Resources

Adequate Hardware and Software resources are available for the development of the prototype. The Hardware available in the Room 1028 at Hughes and the software currently available is described in the following sections.

2.2.1 Hardware Resource

The hardware that is available to the Prototype Development Team includes the following:

- Two SUN Sparc 20/50, 128 MB, 2 GBytes, CD-ROM machine running SunOS 5.4. (Rm. 1028)
- One HP 715/64, 64MB, 2Gbytes, CD-ROM machine running HP-UX 9.07. (Rm. 1028)
- One SGI Indigo2, 320 MB, 6Gbytes, CD-ROM machine running IRIX 5.3. (Rm. 1028)
- Two X-Terminals with 16MB. (Rm. 1028)
- Three Desktop machines running Windows95 with Telnet, X-Terminal session capabilities.

2.2.2 Software Resources

2.2.2.1 Rational Apex C/C++

Rational Apex is an integrated, interactive, software engineering environment for total life-cycle control of complex software projects. It runs on open-systems platforms and is based on industry standards. Rational Apex contains an editor, a compiler, debugging tools and configuration management and version control tools in an integrated development environment.

2.2.2.2 Rational ROSE

Rational Rose is a tool-kit of graphical object-oriented software-engineering to support the capture, communication, and consistency checking of object-oriented analysis and design decisions. It automates the Booch ’93 method and the Object Modeling Techniques of O-O Analysis and design.

2.2.2.3 Configuration Manager

ClearCase from Atria Software provides a comprehensive Software Configuration Management (SCM) capabilities including Version Control, Workspace Management, Build Management, parallel development facilities and process control. The UNIX version runs on all platforms required for this prototype.

2.3 CIP Components

The CIP components that ware required for the complete CIP-Prototype are defined in this section. Figure 2-1, shows the connectivity among the components and each component is described in the later sections.

2.3.1 CIP Components

The dotted lines are shown for encompassing the complete inter-activity among components, however, these fall outside the scope of the CIP-Prototype.

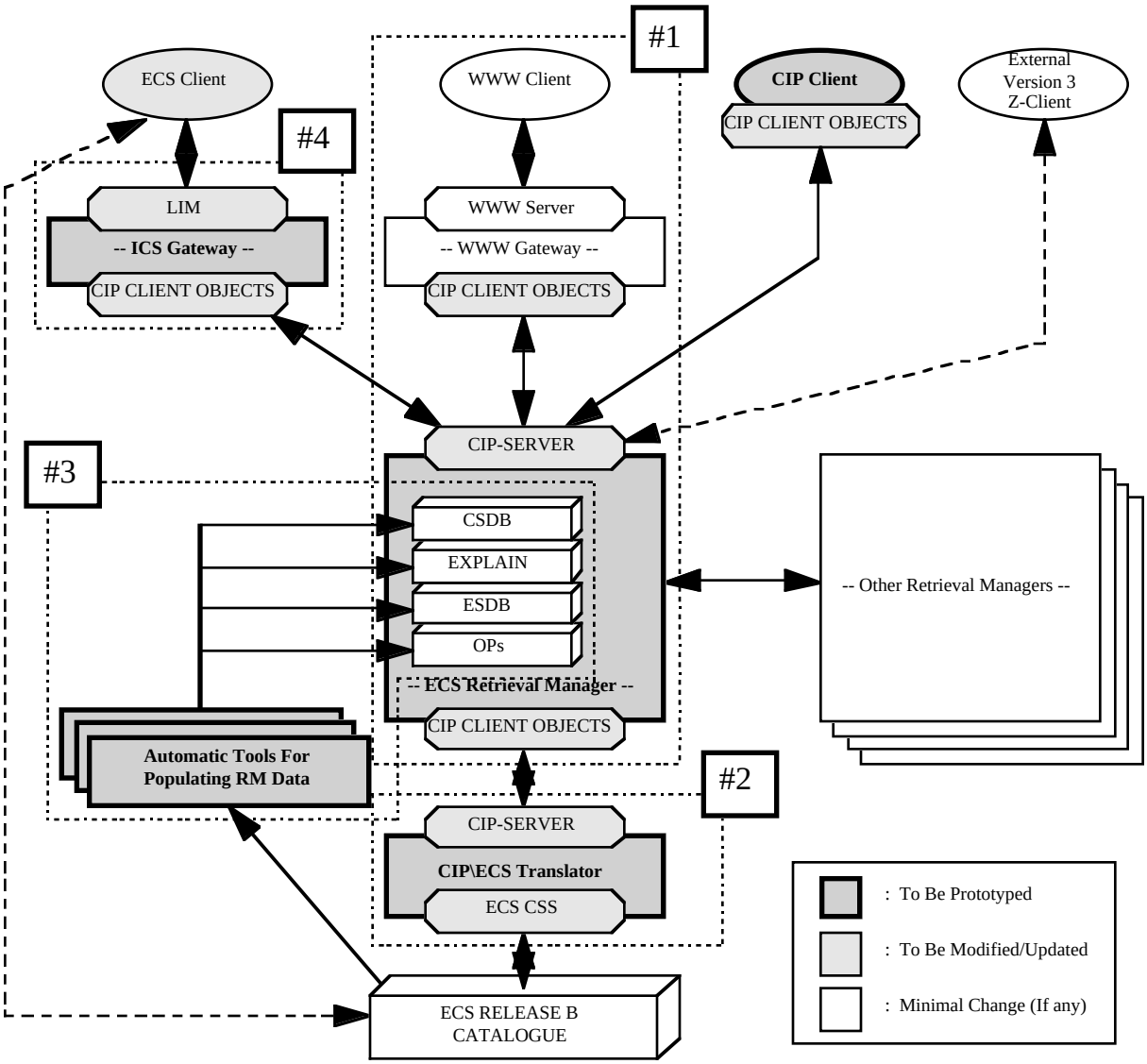


Figure 2-1. The Components of the CIP Prototype

As indicated in the Objective, we will reuse all possible components to construct the CIP-Prototype.

2.3.2 Description of CIP Prototype Components

The components shown in Figure 2-1. are described in the following section.

2.3.2.1 WWW Client

Any COTS Web Browser will be sufficient for the demonstration purposes. However, the browser selected for the CIP-Demonstrator would be more appropriate for conformity. The forms based user-interface developed for the demonstrator can be used 'as is' without changes.

2.3.2.2 WWW Gateway

Various modules that make up the WWW-Gateway are defined below.

2.3.2.2.1 WWW Server

Any COTS Web Server will suffice, however, The Web Server used by the CIP-Demonstrator would be better suited for incorporation into the CIP-Prototype. The Server will recognize the CGI request and pass it on to the CGI Program. It would also package the result returned by the CGI as forms and pass to the Web Browser for user-display. Hence, all the modifications that are required for CIP-Release B compliance will be made to the HTML Forms.

2.3.2.2.2 CGI Program

The CGI Program will take the requests from the Web Server and interact with the CIP-client Application (using the CIP-Client Objects) for required data. The results (status or data) returned to it by the CIP application are converted to HTML forms and returned to the Web Server. This component should exist in the CIP-Demonstrator and it will only need modifications for the additional functionality required for compliance with CIP-Release B specification.

2.3.2.2.3 CIP-Client Objects

This component exists in the CIP-Demonstrator and will only need to be upgraded for additional functionality for CIP-release B compliance and re-hosting to the APEX environment. It is made up of a set of class-libraries that would encompass the Z39.50 Version 3 protocol and the CIP-Protocol. This component may not exist as needed in the CIP-Demonstrator, it will be modified for accommodating all the user applications. The additional Z39.50 functionality that might be needed, could be available in other sources such as the ISITE product from CNIDR.

2.3.2.3 ECS Retrieval Manager

The Retrieval Manager component exists in the CIP-Demonstrator and will be re-hosted to the APEX environment and upgraded to include the additional functionality required for the CIP-Release B compliance beyond the Demonstrator. The required databases that logically reside

within the ECS Retrieval Manager will be populated by means other than the Automatic Tools which will be built at a later stage in the final prototype.

2.3.2.3.1 CIP-Server

This component of the ECS Retrieval Manager (and of CIP\ECS Translator) requires major modifications to be compliant with CIP-Release B specifications. The Server is also required to service non-CIP external Z39.50 version 3 clients. Since CIP-Protocol extends the Z39.50 services, only the Z39.50 version 3 services will be available to the external non-CIP clients. It is unclear how the CIP-Demonstrator handles this requirement.

2.3.2.4 CIP\ECS-Translator

The CIP\ECS Translator (CIP-Server/ECS Catalogue Translator) is the largest component that has to be built for the CIP-Prototype. As shown in Figure 2-2, the Translator has two aspects to it -- the CIP-Server and the ECS Catalogue Interface. The CIP-Server can be carved out of the CIP/RBA Translator and upgraded to meet the requirements of the CIP-Release B specification, however, the ECS Catalogue Interface must be added to it. The ECS CSS components and classes will be of use in this aspect.

2.3.2.5 Automatic Tools for Populating the RM Database

The tools that are required for automatic updating and additions to the RM Databases will be identified in the design phase and developed after the initial prototype. The schema translation that will be developed for the CIP-Prototype is expected to result in better understanding of these tools and their requirements.

2.3.2.6 ICS Gateway

This component will be added to the CIP-Prototype after the initial priorities are complete. The component involves construction of the LIM (and DIM) in the communications to and from the ECS clients. See Figure 2-2 below, for the LIM functionality which would use the LIM Proxy Objects (Client-API) on the client-side and the LIM Server (Server-API) for the server-side processing.

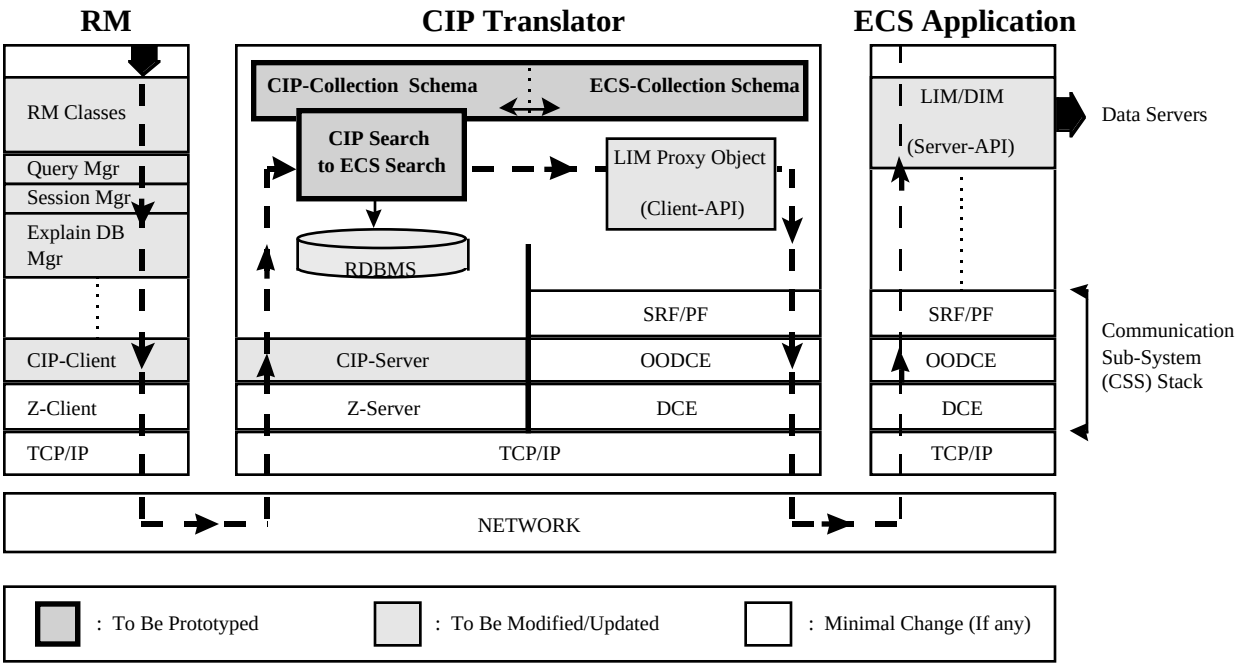


Figure 2-2. Data Communication Stack for the CIP/ECS Catalogue Translator

2.4 Key Risks

The risk that are involved with the project are described in this section. Although, none of the risks are expected to result in major problems, the goal was to clarify the implications of the risk of successful completion of the project. The primary result could be potential delay in meeting the project deadlines and no more.

2.4.1 Resource Risks

Currently the required prototype development team is in place and gearing up for the design and development effort. However, these members of the development team are new to the ECS and EOSDIS on the whole and the risk lies in the on-going training and learning that will continue while the prototype is under development. This could potentially result in retro-fitting of newer modules into the prototype as the on-going learning results in better approaches based on clarification of the nuances of the project. The risk is however reduced due to the adequate supervision and guidance available from ECS engineers outside the team. The other risk aspect is any planned or unplanned vacations by the primary engineers which could effectively delay the schedule.

The hardware and software that is available to the effort is more than adequate, although, there is a risk that Rm. 1028 could be redeployed.

2.4.2 Re-hosting Risks

The recompiling of the CEO CIP-Demonstrator source code under Rational Apex/C++ environment could result in some mismatch within the existing source code due to differences in the development environment as indicated earlier. The strength of the Rational Apex re-engineering environment with the debugging tools however reduces the risk to a degree. The development team is currently learning the software and the risk is mitigated as the group strengthens its knowledge.

The re-hosting will drive the schedule to some degree as this is a variable not completely understood at this point. Any delay in the availability of the CIP-Demonstrator could further delay the timely completion of the CIP-Prototype. This risk must be carefully considered as the primary components required for the CIP-Prototype will be reused from those existing in the Demonstrator.

2.4.3 Implementation Risks

The re-hosting would involve upgrading the software to conform to CIP-B specification and the removal of RBA components. The upgrading from CIP-A to CIP-B is expected to be extensive and a concern, especially since the final CIP-B specification is not currently available. The key risk however, is in the binding of the RBA modules within the software. If the software is loosely bound, i.e. the RBA modules are clearly partitioned from the CIP modules, the migration to ECS would, at the minimum, involve module replacement and conformance to the existing API interface to a large degree. If however, the software is tightly bound, the transition could result in minimal reuse and rewrite of major components.

Another risk exists in the differences between the ECS and ICS architecture, for instance, where ECS has a centralized data dictionary, CIP has distributed collections which could result in poorly translated queries and result sets. Other members of ECS, outside the prototype development group will be instrumental in defining the mapping of these values. The other component that is not well understood is LIM and the DIM schema which will essentially translate from the ECS to CIP.

The overall implementation of the CIP-Prototype using components from a variety of sources carries with it a inherent risk in the integration problems. The development plan is based on a conceptual and theoretical understanding of the components without adequate factual knowledge. The concept of integration of the various components is valid, however, an unforeseeable amount of risk exists.

The final implementation risk is based on an assumption that at some point, other Retrieval manager for applicable databases (such as RBA) would be integrated into the system to show the complete feasibility of the CIP-prototype. This could add some complexity if not taken into account from the start.

2.5 Milestones and Deliverables

The milestones are defined in the following sections.

2.5.1 Objective Milestones

The milestones that are needed for a successful implementation of the CIP-Prototype are defined below. Each milestone is measured in terms of weeks of time required and resources allocated to meet the milestone. The primary members are Ajay Gupte (AG), Ananth Rao (AR) and Eric Martin (EM). *The responsible engineer for each milestone is identified in Bold face font*. Other senior members are expected to contribute in advisory mode and not used in direct calculation of time required.

2.5.1.1 Requirements Analysis (2 weeks; AG, Whole Group)

A group discussion of SDD scenarios that involves the translator could result in a handful of requirements which will become part of the ICS-URD. This meeting of the core members will be held on January 15, 1997.

2.5.1.2 Design (5 weeks; Group Discussions and developed by AR, AG, EM)

The Design document would be made up of the following elements:

- Contain the actual description of the database schema for translation of the CIP Inquiries to ECS.
- It would identify the components of the CIP-Demonstrator which can be directly reused, those that require modifications before reuse and new modules.
- Any Local or Distributed Information Managers (LIM's & DIM's) that are required.
- The Object Model that reflects the entire Prototype and its class descriptions.
- Conversion of Data from ECS to RM Database while defining all the databases and additional function-by-function description
- Identification of automatic tools for populating the RM databases from the ECS data.

The Design Review is scheduled for February 19, 1997.

2.5.1.3 Re-hosting the Demonstrator (3 weeks; EM, AG, AR)

This is dependent upon receiving the source code from the CEO/Logica as defined in the CIP-AD referred earlier in section 2.1.1. The code will be reverse engineered to identify the class hierarchy and Object Model. It would also involve correcting the errors that could come up due to the differences between Rational Apex/C++ and compilers used by CEO/Logica.

2.5.1.4 ECS CSS Integration (2 week; EM, AG, AR)

The ECS CSS (Class Library) and the Object Model will be integrated into the Prototype Project with changes required to meet the requirements of the CIP-Prototype. Any over-head built into the existing system will be maintained unless prohibitive (additional 3-4 weeks).

2.5.1.5 Development/Implementation (12 weeks, AG, AR, EM)

The creation of the CIP-Prototype would involve the following development. The components gathered will be integrated to create the complete Prototype.

- The components will be integrated with newer and more up-to-date interfaces.
- The CIP-A to CIP-B modifications will be made to the existing source.
- The actual CIP-Translator will be constructed using the new Translation schema.
- The ECS components will be integrated into the Translator.
- Other CIP/Z39.50 modification to comply with the Z39.50 version 3 requirements.
- The creation of the Explain Component within the ECS/CIP modules.

2.5.1.6 Testing (2-3 weeks, AG, All)

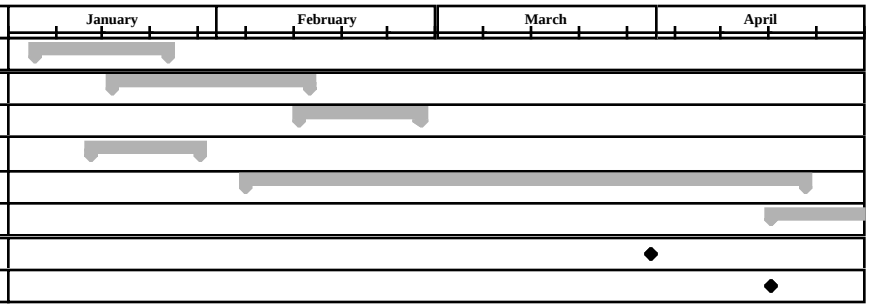
The iterative testing would be required to eliminate any error in the system. Each error will be prioritized, cataloged, tracked, reported and closed by the testers and the developers. The priorities will be identified, as defined in the Software Non-Conformance Report (NCR) for the ECS Project (521-CD-000-007) and Developed Software Maintenance Plan for the ECS Project (614-CD-001-002), in the following order:

- Level 1 Mission Critical Problem: such as loss of Data Sets, etc.
- Level 2 Major Problem : those posing system wide impact affecting data/system.
- Level 3 Minor Problem : those posing no system impact
- Level 4 Nuisance Problem : such as arrangement of screen, colors etc.
- Level 5 Closed Problem : known issue with prior disposition.

2.5.2 Deliverables Time Frame

See following Schedule.

ID	Task Name	Dur	Start	Finish
1	Requirements Analysis	2w	1/2/97	1/22/97
2	Design	5w	1/10/97	2/13/97
3	Re-hosting the Demonstrator	3w	2/9/97	2/26/97
4	ECS Integration	3w	1/8/97	1/29/97
5	Development/Implementation	12w	2/3/97	4/25/97
6	Testing	3w	4/18/97	5/8/97
7	Internal Demo	1d	3/31/97	3/31/97
8	Demo @ CEOS Meeting	1d	4/15/97	4/15/97



2.6 Prototype Demonstration

The following prototype demonstration schedule is expected:

April: Limited demo @ PTT meeting for Access Sub-Group, ECS and NASA.

July: Additions Implemented, available for further tests.

September: Full Demonstration @ Sub-Group Meeting

Other demos as necessary for ECS purposes.

This page intentionally left blank.

Abbreviations and Acronyms

API	Application Programmer Interface
CGI	Common Gateway Interface
CNIDR	Center for Networked Information Discovery and Retrieval
COTS	Commercial Off The Shelf
CSS	Communication Sub-System
DCE	Distributed Computing Environment
DIM	Distributed Information Manager
ESA	European Space Agency
ESRIN	European Space Research INstitute
GNU	Gnu's Not UNIX
HTML	Hyper-Text Markup Language
ICS	Interoperable Catalogue System
IP	Internet Protocol
JEST	Java Earth Science Tool
LIM	Local Information Manager
OODCE	Object Oriented Distributed Computing Environment
PF	Process Framework
PTT	Protocol task Team
RBA	Reference Browse Archive
RM	Retrieval Manager
SCCS	Source Code Control System
SIL	Satellites International Limited
SRF	Server Request Framework
TCP	Transmission Control protocol
URD	User Requirements Document
WWW	World Wide Web

This page intentionally left blank.