Ocean Observatories Initiative

Cyberinfrastructure
System and Software Engineering Plan
(SSEP)

Project	Ocean Observatories Inititative/Cyberinfrastructure
Project Manager	Matthew Arrott
Responsible	Alan Chave
Created on	July 3, 2007
Last Changed on	November 6, 2007
Document Status	PDR Final

Table of Contents

Document History

Version	Date	Author	Version Comments
ID-0.1	05-Jul-07	Ingolf K.	Initial Version, Chapter 6
ID-0.2	08-Jul-07	Alan C.	Chapters 1-5 included
ID-0.3	08-Jul-07	Michael M.	Minor updates in chapter 6
ID-1.0	08-Jul-07	Ingolf K.	Minor edits, added applicable DoDAF documents
ID-1.1	29-Aug-07	Alan C,	Major edit and reorganization
ID-1.2	04-Sep-07	Claudiu F.	Roles and Responsibilities for Req. Docs
ID-1.3	12-Sep-07	Alan	Aligned with WBS
ID-1.4	05-Nov-07	Alan	Aligned with PEP, PDR Final version

1 Scope

1.1 OOI Overview

The OOI comprises three types of interconnected observatories spanning global, regional and coastal scales. The global component addresses planetary-scale problems via a network of moored buoys linked to shore via satellite. A regional cabled observatory will 'wire' a single region in the Northeast Pacific Ocean with a high-speed optical and power grid. The coastal component of the OOI will expand existing coastal observing assets, providing extended opportunities to characterize the effects of high frequency forcing on the coastal environment. The OOI Cyberinfrastructure (OOI/CI) constitutes the integrating element that links and binds the three types of marine observatories and associated sensors into a coherent system-of-systems. Indeed, it is most appropriate to view the OOI as a single entity that will allow scientists and citizens to view particular phenomena irrespective of the observing elements (e.g. coastal, global, regional, ships, satellites, IOOS…) from which the observations derive.

1.2 SSEP Purpose

The purpose of the SSEP is description of the plan for technical management of the OOI/CI project, including technical planning, monitoring and control, and the processes that will be invoked to implement the plan. The SSEP addresses a series of questions regarding the system deliverables:

What system will be delivered?
What tasks must be accomplished to deliver it?
What is the order in which the tasks must be completed?
What are the task dependencies?
What are the final acceptance criteria?
Who will be responsible for each task?
How will each task be carried out?

The SSEP incorporates and combines relevant aspects of both the System Engineering Management Plan and the Software Development Plan. It covers all stages in the system life cycle, from requirements definition through deployment and into operations.

1.3 SSEP Organization

Section 1 of the SSEP provides a top-level description of the document, including its relationship to other OOI/CI project- and system-level plans. Section 2 contains a list of applicable documents that either incorporate or are incorporated into the SSEP by reference. Section 3 is an executive summary level overview of the purpose of the system, a description of the system and a system breakdown structure that graphically depicts its major elements. Section 4 is a high level description of the stages of the system life cycle. Section 5 contains detailed coverage of the system-level planning and control activities that will enable the system life cycle stages. Section 6 is a detailed description of the software development processes. Section 7 is a description of how community stakeholders will provide input into the development process.

1.4 Relationship to Other Plans

The SSEP is incorporated by reference into the Cyberinfrastructure Implementing Organization Project Execution Plan, to which it is subordinate.

The SSEP incorporates by reference the CI Science User Requirements Document, the CI System Requirements Document, the CI Interoperability Plan, the CI Integration and Verification Plan, the CI Deployment and Acceptance Plan, the CI Security Management Plan, and the Department of Defense Architectural Framework (DoDAF) System Architecture Documents.

2 Applicable Documents

Cyberinfrastructure Project Execution Plan PM-001v8r0 2 Nov 2007

CI Work Breakdown Structure CI_WBS_v0r03_20071030_

[DoDAF 1.5 Volume 1] http://www.defenselink.mil/cio-nii/docs/DoDAF_Volume_I.pdf
[DoDAF 1.5 Volume 2] http://www.defenselink.mil/cio-nii/docs/DoDAF_Volume_II.pdf
[DoDAF 1.5 Volume 3] http://www.defenselink.mil/cio-nii/docs/DoDAF_Volume_III.pdf
[DoDAF 1.0 Deskbook]

3 System Overview

3.1 System Purpose

Next generation studies of dynamic, interacting processes in the Earth-ocean-climate system require new in situ approaches to complement the more traditional ship-based, expeditionary science that has dominated oceanographic research for the past century or more. Routine, long-term measurement of episodic oceanic processes is crucial to continued growth in our understanding and predictive modeling of complex natural phenomena that are highly variable and span enormous scales in space and time. This access will be enabled by innovative ocean observatory facilities providing unprecedented levels of power and communication to access and manipulate real-time sensor networks deployed within the ocean. These facilities will empower entirely new approaches to science and enable education and outreach capabilities that will dramatically impact the general understanding of, and public attitude toward, the ocean sciences.

To accomplish this paradigm shift, ocean scientists require at least seven infrastructural capabilities that they do not have now. They must be able to:

- Fully and quantitatively characterize selected volumes of the ocean, the atmosphere overhead and the lithosphere beneath;
- Receive information about all interrelated components of the system simultaneously, in real-time;
- Recognize departures from the norm and observe emergent phenomena to conduct interactive experiments within the environment;
- Reconfigure observational-sampling systems in response to events;

- Assimilate in situ data efficiently into models that expand the space/time view of them and feed back onto the measurement protocols;
- Continue and expand this in real-time;
- Interact with the oceans for decades.

The Ocean Sciences Division of the National Science Foundation has implemented the Ocean Observatories Initiative (OOI) to focus on science, technology, education, and public awareness activities needed to develop and deploy a network of science-driven ocean observing systems that implement this vision. This will provide users with the means to characterize the oceans for decades.

3.2 System Description

The CI architecture makes a clear distinction between the activities that users engage in (collection, assimilation, surveillance and adaptive response) and the resources (data, instruments, networks, analysis processes, computational models and behavior systems) they employ to complete the activities. The interactive nature of the activities and the constraints imposed on the resources represent a substantial extension to data analysis systems as typified by existing data Grid architectures. While data management and analysis are essential elements of the CI, they are not sufficient to support interactive observation and response activities in cyber-physical coupled systems. The architecture extends the capabilities of data and computational Grid architectures by incorporating the ability to employ, couple, and control shared resources operating across the ocean observatory enterprise in real-time over extended periods.

Throughout the observatory network, strategically placed computation and storage infrastructure will be provided at CyberInfrastructure Points of Presence (CyberPoPs). These include integrated real-time data processing and archive sites located at a few central facilities and at marine observatory shore stations or control centers. In situ computation and storage resources will be located within science instrument interface modules (SIIMs) and selectively within the marine networks. Large computational models will be run on the national Grid infrastructure (i.e., the Teragrid and the Open Science Grid). Finally, observatory participants will be able to securely incorporate computation and storage capabilities within their instruments, instrument platforms (e.g., AUVs and gliders) and research facilities.

The resource elements extending beyond storage, data and computation needed to realize the high level CI capabilities can be generalized as:

- Dynamic data sources (e.g., data, product and event streams).
- Taskable elements (e.g., instruments, AUVs, and ocean modeling systems).
- Executable processes (e.g., behaviors, workflows, scripts, and compiled code).

By leveraging and structuring the shared characteristics of the complete set of resource elements, the architecture provides an extensible model for the operation, presentation, and composition of resources. The architecture then provides a standard basis for provisioning, managing, and sharing resources across the physical, technical, and organizational domains of the observatory network. This resource network model provides a consistent and scalable approach to the federation, governance, and operation of all types of resources; one that enables pervasive and universal access without compromising control and security.

The resource network concept facilitates the federation, governance, and management of interactions between system activities and a base set of resource types. It includes information as well as managed and taskable resources. The information resource supplies the interaction semantics for accessing (acquisition and presentation of) data and its associated declarations of identity along with structural and semantic contexts. Information resources can represent immutable (unchangeable), mutable (changeable), and dynamic (continually changing) content with respect to the activity using them. The managed resource extends the information resource to include declaration of operational state and governance context. The taskable resource provides the interaction semantics for the control (configure, command and monitor) of a managed resource that has a dynamic, command-driven operational state.

The organization and structuring of resources taken in the planned architecture recognizes the process-driven nature of an ocean observatory, ranging from simple data collection protocols to a complete investigation model. Figure 1 illustrates the relationships between observatory activities (upper left) and the five core resource networks it employs: the Control, Data, Processing, Instrument, and Modeling Networks (upper right). The Knowledge Network is not currently funded, although provision will be made to extend the CI to incorporate it. The purposes of the five resource networks are:

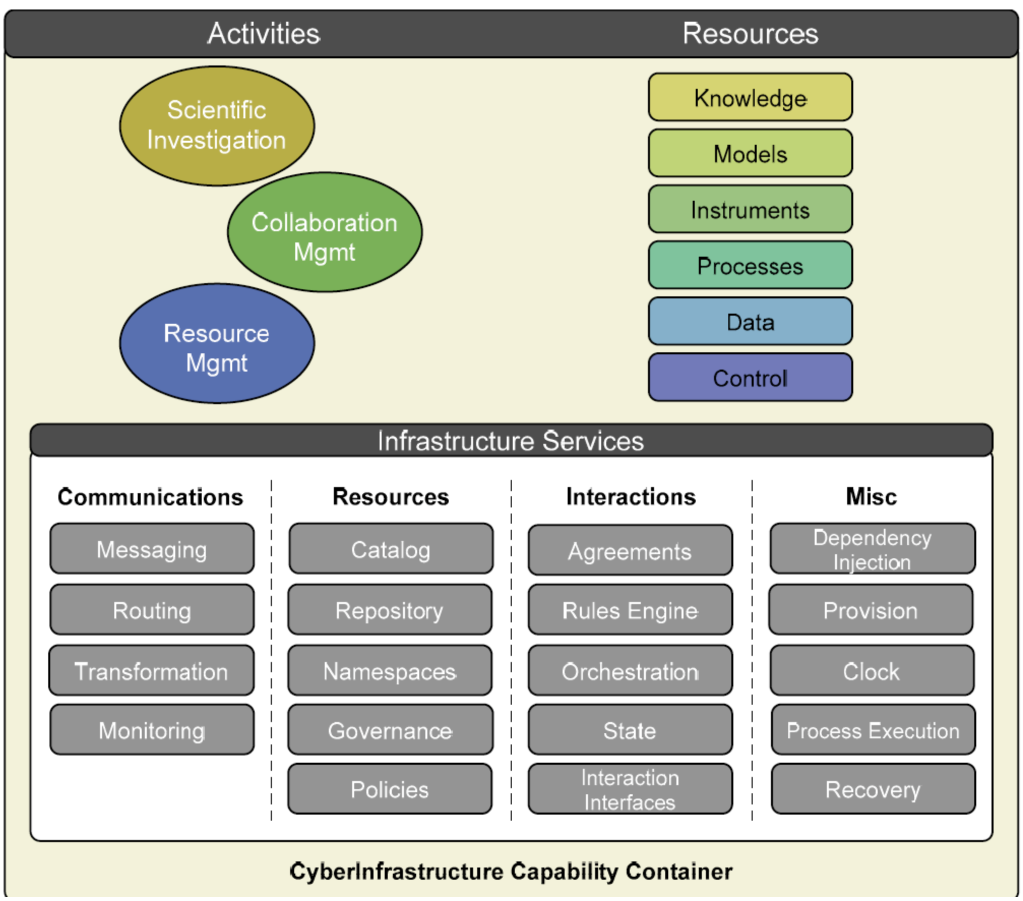

Figure 1. The core CI architecture.

- The Control Network establishes standard models for the management of stateful and taskable resources. It provides the semantics to monitor and control the operating state of an active resource as well as to initiate, monitor, and amend the tasks of a taskable resource.
- The Data Network provides an automated data distribution and preservation network with pervasive and universal access subject to OOI Data Policy. It provisions a federated system of data streams, repositories, and catalogs that supports the distributed organization of resources.
- The Process Network provides immediate-mode scheduling of processes at specified locations within the integrated network based on explicit time requirements and/or event triggers.
- The Instrument Network provides interactive and coordinated relations with real and/or synthetic environments through the use of transducers (sensors or actuators). It ensures the safe and secure operation of individual sensing platforms, and provides reliable delivery of acquired data with their associated metadata. These capabilities must be integrated with network-wide resource allocation and observation planning.
- The Modeling Network establishes baseline processes and tools comprising a coherent framework for the analysis and assimilation of data. The capability to network and interact with multiple community-based numerical ocean models for parameter estimation/optimization and data assimilation is integrated into the framework.

A key innovation of the architecture lies in a comprehensive strategy to support technology integration without depending on any particular software or hardware component. The CI is being built around existing and proven "best of breed" technologies. The heart of this integrative strategy lies in a core infrastructure that frames provisioning, interoperation, and management of activities and resource elements circumscribing the CI. The infrastructure is decomposed into two main elements: the Common Operating Infrastructure (COI) and the Common Execution Infrastructure (CEI). The COI focuses on operation of the system, while the CEI supports provisioning and management of the system.

The Common Operating Infrastructure provides a crosscutting common platform that supports collaboration between individuals, groups, and institutions through the controlled sharing and aggregation of their capabilities. The COI is the core set of frameworks that establish consistent semantics for communication, state, execution, governance, resources, services, and presentation across the OOI enterprise. Each framework implements the domain model for one of these operational aspects of the system. The frameworks establish the interaction interfaces for and between their elements of the system. Using the capabilities of dependency injection and process orchestration provided by an Enterprise Service Bus, new technology solutions can be incorporated by implementing an interaction pattern without impacting the remaining infrastructure. This approach is the basis for all modern operating systems, and is a core principle of object-oriented programming. It is being extended to a distributed resource and authority environment by the CIIO. The same integrative capability is provided to applications through the Service framework.

The Common Execution Infrastructure provides configuration management and demand-driven provisioning of capability at selected locations (CyberPoPs) within the CI network. The CEI is elastic, having the ability to expand and contract the configuration of computing resources as the need rises and falls. This subsystem extends the Amazon® service model for the "Elastic Computing Cloud®" by incorporating a security framework and a demand-based scheduler to enable dynamic configuration.

3.3 System Breakdown Structure

The system breakdown structure (SBS) is included in the SSEP because the WBS derives its structure from it. A product-based WBS will include a Project Office branch, a system engineering branch, a system integration and test branch, and an operations branch in addition to the elements in the SBS.

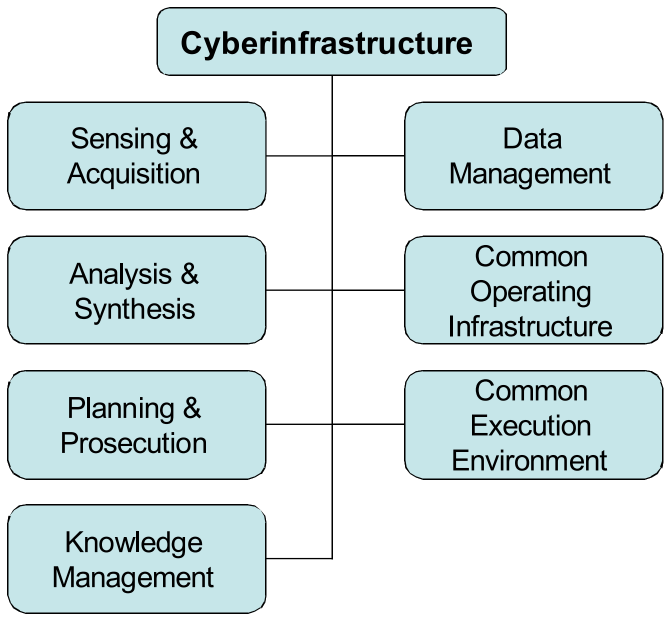

Figure 2. System breakdown structure.

The six elements (subsystems) in the funded SBS (Figure 2) are:

- Sensing and Acquisition: the subsystem responsible for providing the life cycle and operational management of sensor network environments, as well as observing activities (i.e., scheduling, collecting, processing) associated with sensor data acquisition;
- Analysis and Synthesis: the subsystem responsible for providing the life cycle and operational management of community models, ensembles of models and the virtual ocean simulator, as well as modeling activities (i.e., assimilation, analysis, evaluation) using observed and derived data products;
- Planning and Prosecution: the subsystem responsible for providing the mission and campaign planning and prosecution (execution through completion) activities to carry out simultaneous coordinated multi-objective observations across the resources of the observatory;

- Data Management: the subsystem responsible for providing life cycle management, federation, preservation and presentaiton of OOI data holdings and associated metadata via data streams, repositories and catalogs;
- Common Operating Infrastructure: provides the services and distributed infrastructure to build a secure, scalable, fault-tolerant federated system of independently operated observatory components;
- Common Execution Infrastructure: provides the services to manage the distributed, immediate mode execution of processes

4 System Life Cycle

4.1 Relation to Project Life Cycle

The project life cycle is described in Section 2.2 of the PEP, and is based on the spiral development management model that is widely used for software intensive systems. Each spiral (Figure 3) is a four to six-month-long inception phase, a four-month-long elaboration phase, a six-month-long construction phase, and a two-month-long transition phase. Each phase terminates in a milestone review as depicted in the figure. The details and impact of the risks (whether technical, management, operational, or stakeholder) drive the number of spirals, the level of detail, and effort within each phase of a spiral. The riskiest elements are brought forward as early in the development process as possible. Each spiral includes management, engineering, and support activities in proportion to the risks. Each spiral expands system definition and results in a deployed representation of the CI.

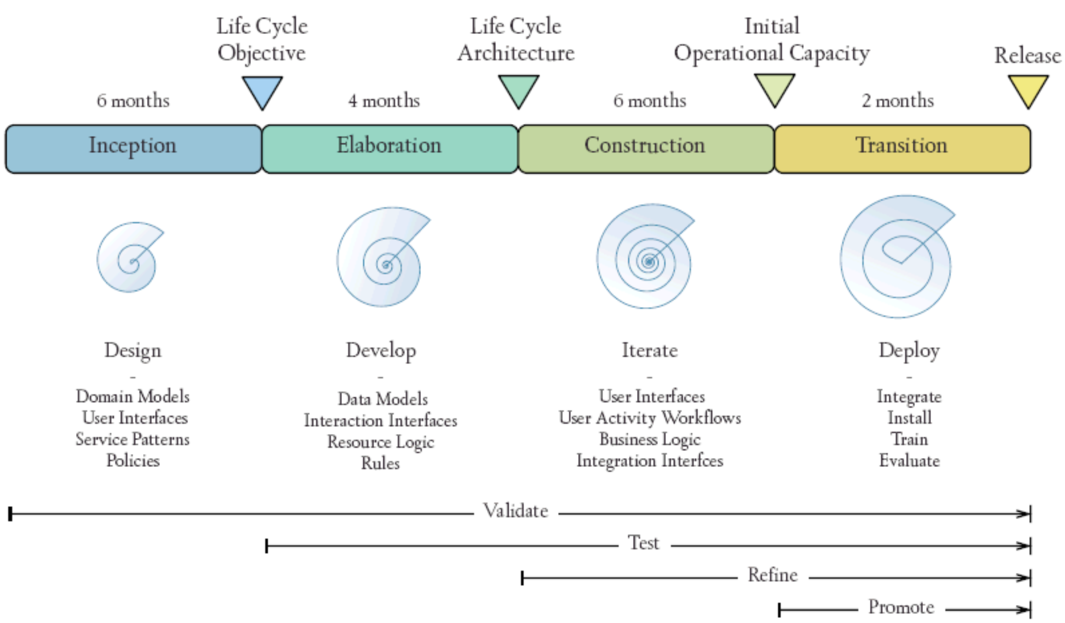

Figure 3. The spiral development project life cycle.

The project master schedule consists of five full development spirals (Figure 4). The entire project consists of an initial six month planning phase, five overlapping sixteen month spiral cycles and ending with a six month transition to operations phase over the six year extent of the project. Beginning with the second cycle, the initiation phase starts four months prior to the end of the preceding construction phase, so that a full development cycle and major software release occurs every twelve months beginning eighteen months after project inception.

The system life cycle consists of a set of planning, requirements definition, and system-level design, prototyping, integration, and verification activities. Under the spiral development model, these activities can occur during any project life cycle phase, although typically they are dominant during particular phases as described throughout this section of the SSEP. In addition, the project is being managed in an agile manner, and in reality, several elements of the system life cycle may occur iteratively during a single phase of a given spiral. For example, requirements and architecture definition are iterative activities that are most productive when carried out over relatively short (order one month) periods. The key system life cycle activities are the responsibility of and under the supervision of the Chief System Engineer, Chief Architect and Software Development Manager, as defined in Section 2.1 of the PEP.

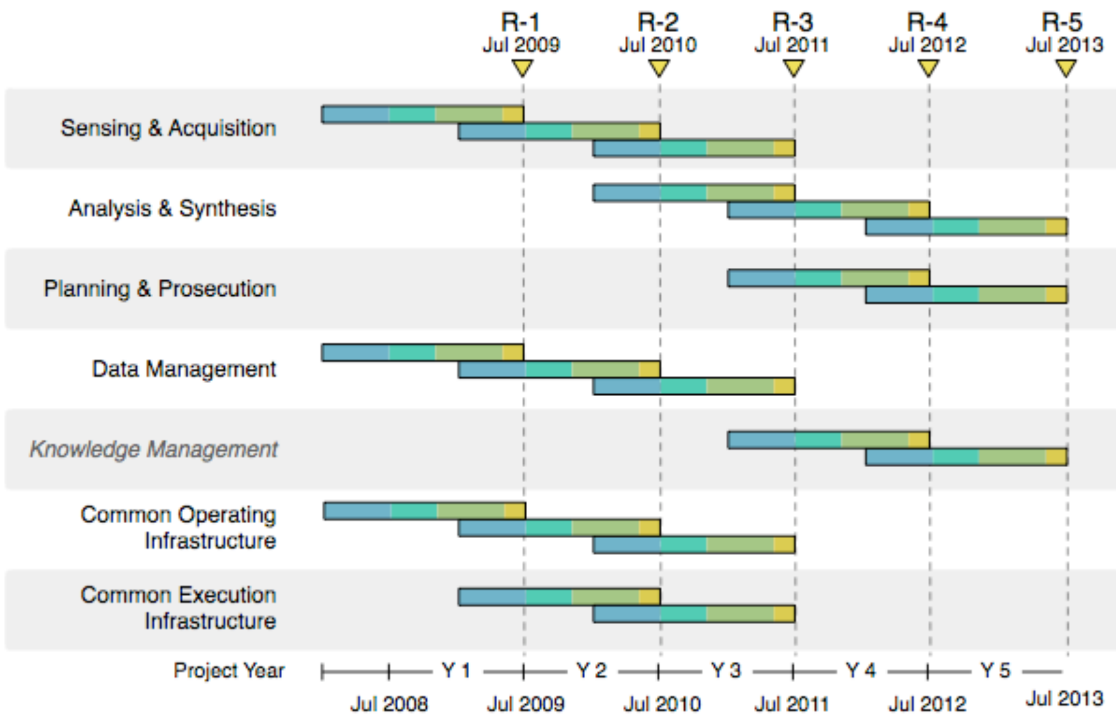

Figure 4. The five overlapping development spirals comprising the project master schedule.

4.2 Role of System Engineering

Use of the spiral project life cycle does not alter the function of system engineering, and in fact it becomes the key activity that binds the cyclically growing system into a coherent whole. The system engineering framework used by the project will be a tailored version of that defined in the System Engineering Handbook Version 3 issued by the International Council on System Engineering (http://www.incose.org).

System engineering is a triumvirate of processes that places equal emphasis on Requirements Analysis, Operational Analysis, and Architectural Analysis. As shown in Figure 5, the three primary processes and their subprocesses are continuously executed in a spiral manner that defines progressively more detailed levels of the system during each loop through the triumvirate. The entire process begins with an examination of the science user requirements Iterations through the triumvirate of processes are executed until the science perspective is thoroughly understood and the Department of Defense Architectural Framework (DoDAF; see Section 6.3) Operational Views have been fully developed. This is followed by additional Iterations through the triumvirate of processes until the technical or system perspective is thoroughly understood and the DoDAF System Views are fully developed. As concepts are exposed or developed, iterations back and forth between the Science and System perspective are executed until the Operational Views and the System Views are balanced.

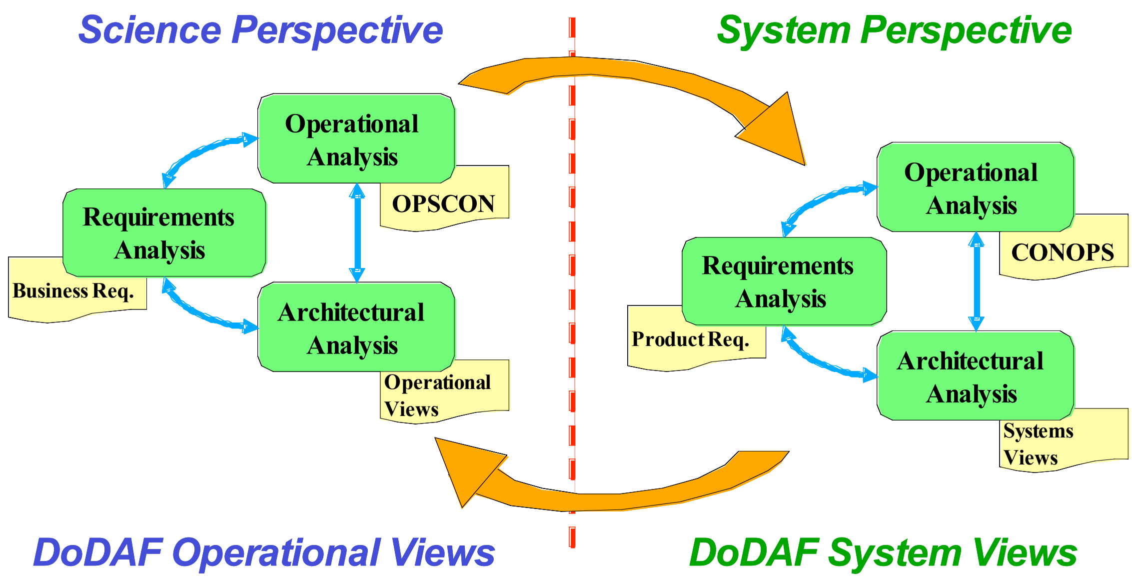

Figure 5. System engineering triumvirate.

Under the triumvirate approach as implemented for the OOI/CI project, the Chief System Engineer is responsible for Requirements Analysis, including the System Requirement Document, and for system perspective Operational Analysis. The Chief Architect, assisted by the System Architecture IPT, is responsible for the architectural design of the system, including production of the design documentation. The Project Scientist is responsible for science perspective Operational Analysis, including devising use case scenarios, and for validation that the user requirements have been met at the time of deployment.

Figure 6 maps the system life cycle onto the spiral model shown in Figure 3, and describes it through a series of activities in each spiral phase that produces a set of artifacts, ending in a milestone review with a defined set of principal assessors. The Chief System Engineer drives the entire process through management of the system life cycle. During the inception phase at the start of each spiral, the main activities are highly iterative requirements gathering and architectural design activities. These terminate with the LCO milestone review in which the assessors are the Software Development and System Engineering IPTs. The elaboration phase is dominantly carried out by the System

Architecture and Software Development IPTs, and involves rapid prototyping to minimize risk, It terminates in the LCA review for which the assessors are the stakeholders and Project Scientist. The construction phase is primarily carried out by the Software Development IPTs. and has the key goal of building and integrating a functional system that meets the requirements. It ends with the IOC milestone review during which the evaluation activities are verification and validation, with the Chief System Engineer and Quality Assurance IPT assessing the former and the Project Scientist and stakeholders carrying out the latter. During the transition phase, the activity is integration of the CI and the marine IO components authored by the System Integration IPT with the goal of producing a deployed system. It terminates with a release and is evaluated through the acceptance process carried out by the OOI Program Office and stakeholders. Under the alignment of successive spirals shown in Figure 4, the LCO and release milestones are coincident.

The following sections describe the system life cycle activities in more detail.

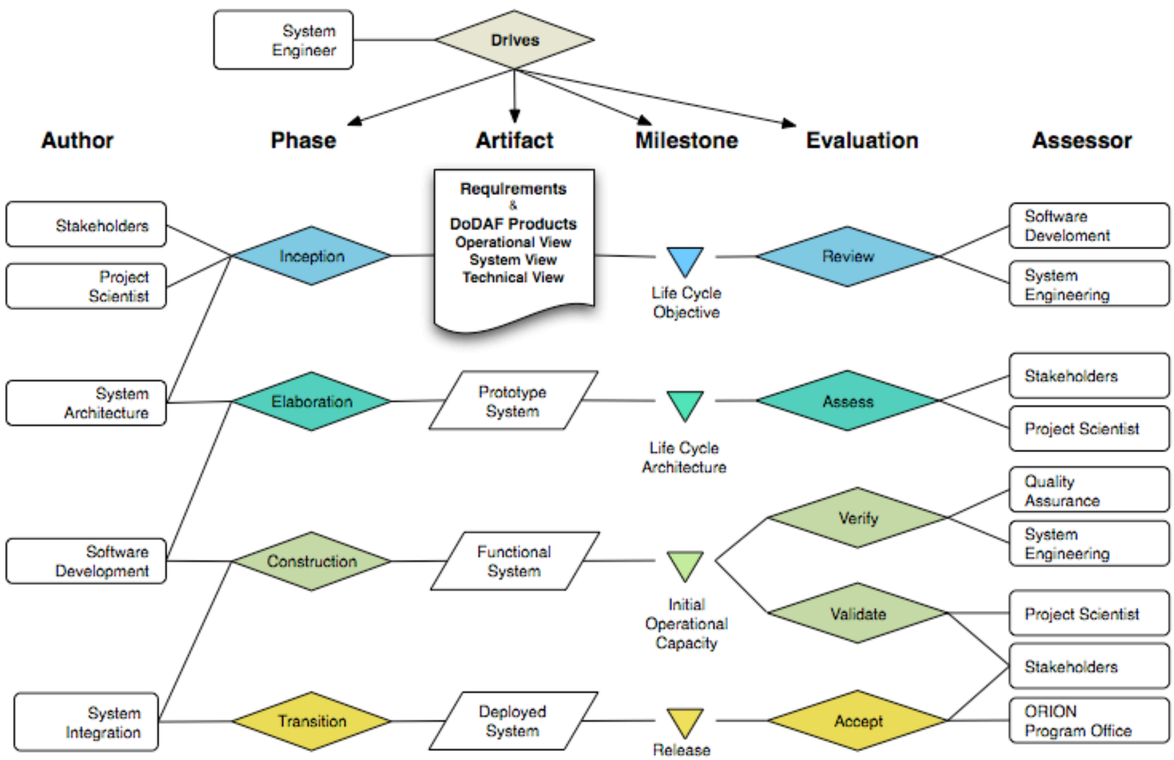

Figure 6. Engineering Process Model

4.3 Science User Requirements

Design of the CI architecture must be driven by the needs of the diverse domain stakeholders who will use it. The elicitation of user requirements is the initial phase of an ongoing stakeholder oversight process that must continue through the deployment and acceptance phases of the system life cycle. Extracting requirements from diverse stakeholder communities presents a challenge, as the typical user may not be familiar with many of the relevant information technologies, and will not be able to readily quantify present and future needs in a manner that will lead to a formal design. The Project

Scientist, E&O Manager, Chief System Engineer and System Architecture IPT will work with domain scientists and educators to overcome this issue.

The requirements IPT must construct a wide range of use scenarios (i.e., operational concepts) and concepts of operations incorporating representative suites of sensors and platforms in close collaboration with a representative group of domain users. A complementary process involves preparation of a Concept of Operations Document by the Chief System Engineer. The ConOps document describes what the system will do and why it will do it from the user perspective. It serves the important function of seeding stakeholder thinking with feasible capabilities and applications of the CI, and serves to bootstrap the use case scenario development effort.

4.4 System Requirements

System and subsystem requirements definition is the dominant activity in each inception phase (Figure 3), but continues at reduced levels throughout each development spiral. The Chief System Engineer and System Architecture IPT extracts system requirements from the science user requirements and other sources. Requirements definition will involve the stakeholder communities organized through the Project Scientist and E&O Manager, the entire CI development team, and the Program Office. Additional elements of the system requirements are the policies that govern use of observatory resources, which must be negotiated with the marine IOs, OOI Program Office and stakeholders.

4.5 System Development

System architecture definition is the dominant activity during each elaboration phase, although it continues at a reduced level in the other spiral phases. With input from the Subsystem Development IPTs, the System Architecture IPT defines and refines the system architecture with guidance from the evolving requirements. Architecture specification activities may include prototyping and trade studies as needed. Quality assurance is a cross-cutting activity throughout system development.

4.6 System Integration and Verification

During the construction phase of each development spiral, the newly created architectural elements are integrated into any existing CI release and verified for compliance with the system requirements by the Software Development Manager. Upon completion, the integrated and verified CI is deployed into the marine IOs, and then undergoes a second verification process by the Chief System Engineer. The System Integration IPT carries out both activities to provide continuity, and the Operations Manager oversees deployment to give a seamless transition to the operations and maintenance phase.

4.7 System Validation

Near the end of each construction phase, the newly integrated and verified CI will be validated to show that the result meets stakeholder needs through compliance with the science user requirements. The Project Scientist is responsible for validation with assistance from the Chief System Engineer and selected stakeholders.

4.8 System Deployment and Acceptance

During each transition phase, the integrated, verified, and validated system is delivered to the Operations Manager for deployment. As part of this activity, the system is accepted by the OOI Program Office using specified criteria. After deployment, the CI is available for operations and use by the scientific and educational communities.

5 System Life Cycle Management

5.1 Process Performance Management

This task involves the collection and analysis of information regarding the status of the project and its comparison against the relevant plans to assess progress. Assessments are scheduled at regular intervals as determined by the Project Manager, at all milestone reviews, and at other decision gates as required. The primary goal of Process Performance Management is keeping open and thorough communication within the project team and with the stakeholders.

Process Performance Management consists of two elements: Self Assessment Process and System Engineering Measures. The former will be carried out on a monthly basis in conjunction with earned value management reporting to the Project Office, and consists of evaluation by the Chief System Engineer and Chief Architect of how system level processes are functioning, identification of problem areas, and communication of lessons learned. These will be communicated to the Project Manager for possible action or wider distribution.

The principal system engineering measure will be earned value, as described in the PEP and as handled by the Project Manager in the CI Project Office. The Cost Account Managers (CAMs) are responsible for reporting cost and progress information to the Project Office on a monthly basis for incorporation into the CIIO Earned Value Report to the OOI Program Office. The Project Manager will determine the format and content of these reports.

5.2 Technical Performance Management

Technical performance management (TPM) is the process of prioritizing the system requirements, objectives, operational constraints and maintenance concepts. These disparate issues must be traded off to provide a system that delivers what the stakeholders need within project and program constraints such as cost caps, delivery schedules, the need for phased development and the deployment schedules for other elements of the OOI.

A key tool for technical performance management will be the quality function deployment (QFD) method to establish ongoing connection between the stakeholders and the designers. QFD is a team approach that ensures that the voices of the stakeholders are involved in the evolving system design and deployment cycles. The purpose is establishment of the requirements and their translation into a technical solution. Stakeholder requirements and preferences are defined and categorized as attributes that are weighted according to their importance. QFD provides the design team with a thorough understanding of stakeholder wishes, forces the stakeholders to prioritize those wishes, and hence allows trades between different solutions to be evaluated. A standard QFD process is construction of a house of quality matrix as shown in Figure 7.

The Chief System Engineer is responsible for oversight of the TPM process, with the Project Scientist and E&O Manager providing key links into the stakeholder community and the entire design team acting as participants. Selected representatives from the stakeholder community will be integrated into the Subsystem Design IPTs as appropriate, and stakeholders will be involved in all milestone reviews.

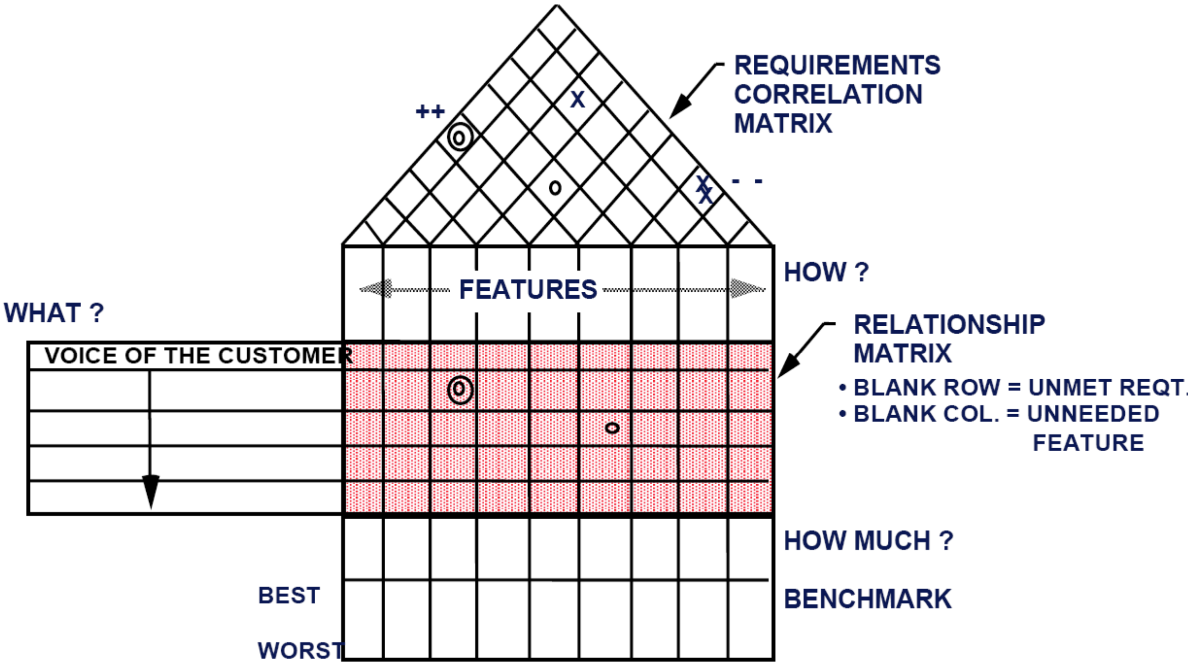

Figure 5. The house of quality matrix.

5.3 System Engineering Metrics

A set of system engineering metrics will be implemented and integrated with the project-level metrics outlines in Section 2.3.1 of the PEP. These will consist of:

- The complete set of earned value metrics that encompass cost and technical areas;.
- A set of TPMs that are derived directly from the performance requirements, as described in the previous section;
- The following additional metrics
  - Total number of requirements
  - Number of requirements associated with an approved baseline change
  - Change in number of requirements
  - Total number of design elements
  - Change in number of design elements
  - Total software size, either source lines of code (SLOC) or function point analysis (FPA)
  - Change in software size (SLOC or FPA)
  - Total number of test cases & test procedures
  - Change in number of test cases & test procedures
  - Total number of defects

Number of defects detected per test case
Defects open during a reporting period
Defects closed during a reporting period

5.4 Requirements Management

Under the spiral model, requirements definition constantly evolves throughout the project life cycle, and hence is an ongoing activity. Requirements and their provenance will be captured using the Telelogic DOORS tool. The requirements definition process results in three documents. The first is the Science User Requirements Document (SURD) that includes use case scenarios and is under configuration control with the Project Scientist as document custodian. The second is the Concept of Operations Document (ConOps) that is under configuration control with the Chief System Engineer as document custodian. The final document is the System Requirements Document (SRD) that is under configuration control with the Chief System Engineer as document custodian. The System Requirements Document (SRD) serves as the top-level description of the desired CI capabilities. A key element of the SRD is the Requirements Traceability Matrix that links the system to the user requirements. The Chief System Engineer is responsible for ensuring that all requirements are stated in an atomic, clear, verifiable manner with source traceability. The Chief System Engineer and Chief Architect are responsible for trade studies, constraint evaluation, and cost-benefit analyses that may be required to refine them. Requirements review is a key purpose of each Life Cycle Objectives milestone review, which is analogous to the waterfall model Concept Design Review.

The SURD and ConOps Documents are subject to review by the stakeholder communities and approval by the OOI Program Office. It is the responsibility of the Project Scientist and E&O Manager to communicate user-initiated proposed changes to the Chief System Engineer and Chief Architect on an ongoing basis. It is also the responsibility of the Project Scientist and E&O Manager to communicate changes in CI capabilities to the stakeholder communities.

For a detailed description of the roles, responsibilities, artifacts, and associated requirements elicitation process, see Section 6.2.

5.5 Interoperability Management

The Interoperability Plan describes procedures that will ensure the interoperability of the CI both internally and with the hardware and software elements produced by the marine IOs and key external entities, notably IOOS. It includes Interface Control Documents (ICDs) that describe the interfaces between all system and subsystem elements internal to the CI, between the CI and the OOI marine observatories, and between the CI and external entities. ICDs internal to the CI are negotiated between the Software Development Manager, subsystem IPT leads and the Chief System Engineer, who has approval authority. The Interoperability Plan also includes Interface Agreements (IAs) negotiated between the project and the marine IOs that establish interface requirements, constraints, and milestones. The IAs incorporate relevant ICDs, and are negotiated between, and signed by, the cognizant IO System Engineers subject to approval by the cognizant IO Project Managers. The OOI Program Office System Engineer has final approval authority, and resolves any conflicts that may arise. Finally, IAs may be negotiated between the project and external entities under similar conditions. The Chief

System Engineer is responsible for developing and implementing the Interoperability Plan, which is under configuration control.

5.6 Risk and Opportunity/Change Control Management

Changes to the baseline design must adhere to the procedures defined in the Configuration and Change Control Management Plan (CCMP), and must be evaluated for risk according to the Risk and Opportunity Management Plan (ROMP). Both of these plans are the responsibility of the Project Manager.The Project Manager serves as risk manager for the project, and chairs the Risk and Opportunity Management Board defined in section 2.5 of the PEP. The Chief System Engineer is a member of the board.

The CCMP charters an Engineering Review Board (ERB) that ensures and controls technical performance of the system; reviews decisions, issues and trades that could have technical, cost or schedule impacts; and reviews and closes discrepancy reports. The ERB consists of the the Chief System Engineer (chair), the Chief Architect, the subsystem IPT leads, the Quality Assurance Engineer, and the Operations Manager. An Architecture Review Board serves as an arm of the ERB, and has parallel functions to it for changes that have architectural impact. The ARB consists of the Chief Architect (chair), Chief System Engineer, System Architecture IPT, and subsystem IPT members as required.

The CCMP also creates the Project Control Board (PCB) that approves or rejects all changes to the project requirements, cost and schedule baselines, as well as changes that have an impact between subsystem IPTs. The ERB refers all matters that meet this description to the PCB for a decision. The PCB consists of the Project Manager (chair), the Deputy Project Director, the Project Scientist, the E&O Manager, the Chief System Engineer, the Chief Architect, the Software Development Manager, and the Operations Manager. The ERB refers all matters that meet this description to the PCB for a decision. The Project Manager will send approved changes on to the OOI Program Office for final approval as required.

5.7 Deficiency Management

The Chief System Engineer is responsible for oversight of system-level activities to detect problems and for the maintenance of the Deficiency List. The Deficiency List is conveyed to the Project Manager together with a monthly report on the system for inclusion in the corresponding report to the Program Office.

5.8 Integration and Verification Management

The Integration and Verification Plan (IVP) establishes sequences and schedules for integration of the subsystems with each other, with the existing CI, and with the marine and external elements at successive development spirals. It also establishes criteria to verify the system by asking “was the system built right?” through establishing that the system requirements have been met. ISO 9126 serves as a framework of verification attributes and criteria. Preparation and custody of the IVP is the responsibility of the Chief System Engineer. The Software Development Manager is responsible for delivery of a quality integrated CI to the Chief System Engineer. The Chief Architect is responsible for verification of the integrated CI, while the Chief System Engineer is responsible for integration of each CI release with the marine and external observatory elements and verification of the result. At the end of each integration and verification phase, the Chief

System Engineer must submit an Integration and Verification Report to the Project Manager that includes a Requirements Verification Compliance Matrix

5.9 Validation Management

The Validation Plan (VP) establishes the criteria and process for evaluating the completed system at each release against the science user requirements by asking "was the right system built?". Preparation and custody of the VP are the responsibility of the Project Scientist, who will carry out the validation process with the assistance of the Chief System Engineer and selected stakeholders. Upon completion, the Project Scientists will prepare a Validation Report for submission to the Deputy Project Director, who has approval authority.

5.10 Deployment and Acceptance Management

The Chief System Engineer is responsible for developing a Deployment and Acceptance Plan that defines the process for CI deployment at the end of each development spiral and the criteria for its acceptance by the COL Program Office. The Plan will also describe the documentation that allows the system to make the transition to operations and specify the training required for operations personnel. The acceptance process will comply with requirements imposed by, and will be overseen by, the COL Program Office, which has ultimate responsibility for accepting the CI. The deployment and acceptance process follows on the integration, verification, and validation processes described in the Integration and Verification Plan and Validation Plan, respectively. Deployment will be carried out by the Operations IPT and the System Integration IPT, with oversight by the Operations Manager and Chief System Engineer. The Operations Manager will prepare a Deployment and Acceptance Report after each development spiral.

5.11 Integrated Logistics Support Management

Integrated Logistics Support (ILS) defines all of the elements required to support the CI system throughout its life cycle. It is usually divided into ten components:

- Maintenance planning
- Supply support
- Test equipment/equipment support
- Manpower and personnel
- Training and training support
- Technical data
- Computer resources support
- Facilities
- Packaging, handling, storage, and transportation (PHS&T)
- Design interface.

The ILS process and products are described in the Integrated Logistics Support Plan (ILSP) that is the responsibility of the Chief System Engineer. The objective of the ILS process is the design, development, and delivery of a supportable, reliable, sustainable, maintainable and affordable OOI/CI that meets the required operational performance, availability and Total Ownership Cost (TOC) goals. The ILS process will:

- Identify, define and incorporate appropriate supportability characteristics into the OOI/CI system design
- Develop and deliver the OOI/CI focused product support to sustain TOC goals

The Chief System Engineer, Chief Architect, System Architecture IPT and subsystem IPT leads serve as the Supportability IPT. Each of the remaining IPTs take the lead for the ILS aspects of their respective OOI/CI hardware/software. The major functions of the OOI/CI Supportability IPT are:

Preliminary Candidate Review Both hardware and software candidates for inclusion in the OOI/CI system are subjected to a continuous review process during the configuration refinement process. Typically, the reviews consist of vendor contact to evaluate technical documentation, equipment reliability, long-term viability, and the willingness/ability to provide logistics support that meets specified requirements. This is a best value approach to eliminate non-conforming candidates, while minimizing resource expenditures.

In-Depth Review Candidates surviving the initial evaluation are subjected to an expanded analysis. Failure analysis and mean time between failures (MTBF) are assessed. Maintainability aspects such as built-in-test (BIT)/built-in-test-equipment (BITE) capability, required maintenance labor hours, skill levels, test and support equipment, and training are quantified to the level required for trade studies, associated life cycle cost/cost as an independent variable (LCC/CAIV) modeling and determining the cost of the OOI/CI mission system baseline configuration.

Trades Typical quantifiable trade elements include LCC data, technical evaluation, quality of service levels achieved, supportability analysis, as well as supportability and cost drivers. These elements provide support data for the design process. The overall effect of this type of analysis early in the design and development phase is one of ensuring system supportability and reduced LCC.

LCC/CAIV Modeling Industry standard models have been identified for manipulation of the maintainability and cost data. An initial collaborative ILS undertaking is the definition of preferred software models, level of modeling required, and output documentation desired to support analysis activities. The project team has trained engineering specialists experienced in utilizing LCC software tools.

Outputs Outputs of the specified models, coupled with other logistics analysis parameters, are formatted to enable trade studies. Evaluations of alternatives are included in this process. Output formats are defined by the development organizations to provide the data necessary for informed, best value decisions.. Accomplishment Criteria (AC) for this process include delivery of the identified contract data requirements list (CDRL) items.

Logistics Support Analysis (LSA) Analysis of the baseline reference architecture is conducted on the approved baseline, providing initial and subsequent recommendations to the development organizations for maintenance planning, OS training, documentation, and other ILS support elements.

Logistics Product Development OOI/CI product supportability packages can now be defined and developed in accordance with the approach identified within the ILSP, meeting the requirements for supportability of the OOI/CI system that is cost effective and provides the necessary infrastructure support to achieve availability requirements. Ongoing iterative coordination between the project IPTs is critical to ensure successful integration of all the supportability packages

OOI/CI Focused Product Support The OOI/CI support packages are now in place and total OOI/CI product support has been established in time to sustain initial operational capabilities.

Results from each process step provide a valuable history of trade results, analysis, and decisions that culminate in the final ILSP. They also provide communication to management during process execution, along with historical records for future program participant reference. The results are captured in informal documentation, sent to project management and are made a part of program records.

5.12 Security Management

The Chief System Engineer and Chief Architect are responsible for preparing and implementing the Security Management Plan that covers all aspects of operational and CI security for the system, including defining the software and hardware “best practices” (e.g., firewalls, one-time passwords, anti-virus software) that will be used to protect against intrusion on a real-time basis and the processes used to define and manage reportable incidents both within OOI and at the federal level. It will also describe the authorization and auditing policies for the CI at different levels of access and the ongoing process for ensuring that repositories remain free from external aggression. Compliance with national security requirements will also be described. The Security Management Plan will incorporate any additional requirements imposed by the Program Office and NSF.

6 Engineering Process

6.1 Development Methodology

6.1.1 Software Development Process and Standards

Software and system engineering activities will follow a highly iterative process pattern with a nominal one month duration. Each iteration defines a number of development products that will be delivered with a prescribed state and content. Some deliverables are external only, and will be subject to review and acceptance. The dates for deliverables must be coordinated and consistent with project milestones.

The Software Development IPT, consisting of the Software Development Manager and Subsystem IPTs, together with the Chief System Engineer and System Architecture IPT, will leverage and further develop existing technologies to ensure both the rapid availability of an initial CI implementation and the systematic, iterative, and incremental implementation of the full set of CI capabilities in accordance with the release schedule. At its core, the OOI software development effort is a system-of-systems integration challenge, with software development primarily focused on subsystem adaptation and integration of best-of-breed, proven technologies.

The conceptual architecture rests on a rigorous service-oriented design approach to project subsystem capabilities and integration into system capabilities. The SDT will leverage service-oriented implementation techniques to yield a seamless software and system engineering project framework. Intuitively, every OOI entity (examples range from instruments to laboratories to data repositories to coastal, regional, and global observatories to the computational Grid) will represent itself as a set of services that can be located and accessed via the CI. Web services and related technologies enable rapid implementation, provisioning, and integration, along with flexible configuration of these services to yield the CI.

6.1.2 Software Development Environment

The Software Development Environment consists of the following tools:

Table 1: Software Development Environment Tools

Process Result Product	**Tool(s)**
Project Planning	Microsoft Project
Requirements Specification	Telelogic DOORS
Software Architecture Design	Telelogic Software Architect
Software Development	Eclipse IDE, based on Java SE 6
Version Control, Configuration	Subversion
Issue Tracking System	Atlassian JIRA/CONFLUENCE
Documentation	Microsoft Word/Excel/Powerpoint

6.1.3 Software Configuration Management

6.1.3.1 Version Control

Version control of development artifacts will be provided through a CollabNet Subversion installation running on a specially protected server and archived on a daily basis. Subversion has strong industry support in the software development domain, and is seen as one of the potential successors of CVS (Concurrent Versions System), the current industry standard for version control.

Each created artifact will be uniquely identified by its file name and location in the hierarchical Subversion file structure that acts as the repository. A revision number is automatically associated with each version of a file. The revision number is automatically created by the Subversion server; it uniquely identifies a file version in combination with the file name and repository location.

There is always one current version of a file called the head revision. Multiple users can get copies of all repository files or parts thereof on their local workstations. These copies can be edited locally and concurrently, and committed to the server when finished. The Subversion server ensures that the current server versions of the files are updated consistently. The Subversion repository provides capabilities to access all versions of a version-controlled object and to compare differences between versions. Sets of files can be combined into new configurations.

6.1.3.2 Configuration Identification

Version numbers for significant project deliverables and artifacts are assigned when needed. Version numbers are different from the revision numbers that are automatically created by the version control system, as they are manually assigned by the configuration manager for artifacts and groups of artifacts (configurations) that have a certain status. Configuration information is maintained by the version control system Subversion.

6.1.3.3 Configuration Control

After release of a software version or document, all changes shall be recorded and associated with the description of the change and cause of the change.

6.1.3.4 Configuration Status

Each configurable object has a status that is one of:

Table 2: Artifact Status

Status	Meaning
Initial	File newly created
Draft	File currently edited targeting a planned release
Candidate	File prepared ("frozen") for external QA
Final	File finalized, incorporating minor addressed QA requests

The status can change in the order given. It can also change back to draft from candidate and final if more significant modifications are required or a new release is targeted.

6.1.4 Software Quality Assurance

Project quality assurance activities are defined in the Quality Assurance Plan (QAP) that the Project Manager is responsible for. The QAP defines the software quality assurance policies and procedures that will be implemented throughout the design/build/integrate/deploy cycles.

The Quality Assurance IPT oversees quality control throughout each development cycle. The key activities include monitoring the implementation of the project plans, controlling the acceptance of project deliverables, and providing input to the Project Manager on risk identification and mitigation. The Quality Assurance Engineer reports to the Project Manager and is responsible for system development quality assurance during each development spiral.

Analytical and constructive quality assurance measures will be defined jointly by the Chief System Engineer and Chief Architect according to the criticality of software components and in compliance with the QAP. Analytical QA measures include software testing, code/documentation reviews and walk-throughs, and run-time monitoring. Constructive QA measures include formal methods specification, design verification and simulation.

6.1.5 Software Risk Management

A software risk list will be created during the early domain analysis and architecture definition phase by the Chief Architect and Chief System Engineer. The list will be maintained through ongoing Engineering and Architecture Control Board activities. In the event that an identified risk has significant cost, schedule or scope impact, it will be referred to the Program Control Board by the ECB using standard procedures.

6.1.6 Primary Artifacts

The primary engineering process products and artifacts are listed in the table below

Table 3: Primary Engineering Artifacts

Artifact	Responsibility	Purpose
Science User Requirements Document	Project Scientist	Documents use case scenarios and dentifies science user requirements
System Requirements Document	Chief System Engineer	Identifies system requirements; includes requirements traceability matrix; enables tracing to infrastructure elements; can be verified
ConOps Document	Chief System Engineer	Describes what the system will do and why it will do it from the user perpective
System Architecture (DoDAF) Documents	Chief Architect	Documents the architecture from the operational and system perspectives. Identifies all subsystems required to fulfill the user requirements, as well as crosscutting infrastructure elements.
System and Software Engineering Plan	Chief System Engineer	Lists system level processes, tasks, responsibilities and authorities, including internal schedules and deliverables as well as external deliverables
Interoperability Plan	Chief System Engineer	Defines internal and external interface control processes and includes interface control documents defining all interfaces
Integration and Verification Plan	Chief System Engineer	Defines the processes for internal and external integration of the CI and criteria for their validation
Subsystem Architecture Documents	Chief Architect	Defines the required subsystem capabilities
Subsystem Implementation Documents	Subsystem IPT Leads	Specifies the deployed form of the subsystems

6.2 Requirements Engineering

Initial system requirements are part of the CI Conceptual Architecture, and were developed from an examination of existing observatory projects (LEAD, SIAM, SSDS, IOOS DMAC, and VENUS/NEPTUNE Canada) along with input from the OOI advisory structure and the OOI Science Plan. Further user and system requirements definition will involve the stakeholder communities organized through the Project Scientist and E&O Manager, and the entire CI development team. Additional elements of the system requirements are the policies that govern use of observatory resources. Their specification will follow a parallel process, but may involve negotiation with the marine IOs, and may require COL Program Office approval. The system requirements will be divided into four major categories (functional requirements, performance requirements, design principles, and interface requirements), and then further sorted into categories that are consistent with the CI subsystems.

Requirements will be captured throughout the project through direct stakeholder involvement. The requirements capture process will be iterative in nature to provide immediate stakeholder feedback and requirements adjustment. The creation of infrastructure prototypes will provide further feedback for stakeholders and validate selected requirements. With early prototypes, it is possible to detect technology risks expeditiously and constrain stakeholder expectations.

6.2.1 Roles and Responsibilities

Figure 6 defines the detailed roles and responsibilities underlying the requirements process.

Chief Architect

- Defines requirements management process in collaboration with Chief System Engineer; creates initial requirements management plan
- Defines requirements document (RD) template
- Defines requirements elicitation and management tool support and establishes tool support environment
- Delineates boundary between requirements and design
- Elicits requirements from source documents and requirements workshops
- Provides requirements elicitation materials as required
- Elaborates requirements to the level of detail necessary for architecture design according to priorities set by the Chief System Engineer (system level) and Project Scientist (user level)
- Supports requirements controlling and prioritization by providing feasibility information and realization effort estimates based on conceptual system architecture and existing/projected technologies and tools. Provides prototypes to validate requirements feasibility
- Iteratively analyzes and documents requirements in RD (both user and system level)
- Prepares RD deliverables (both user and system level)
- Keeps RM and RD artifacts under configuration and version control and performs document archival
- Performs requirements change management and issue tracking
- Submits RD (both user and system) for review to Chief System Engineer and Project Scientist

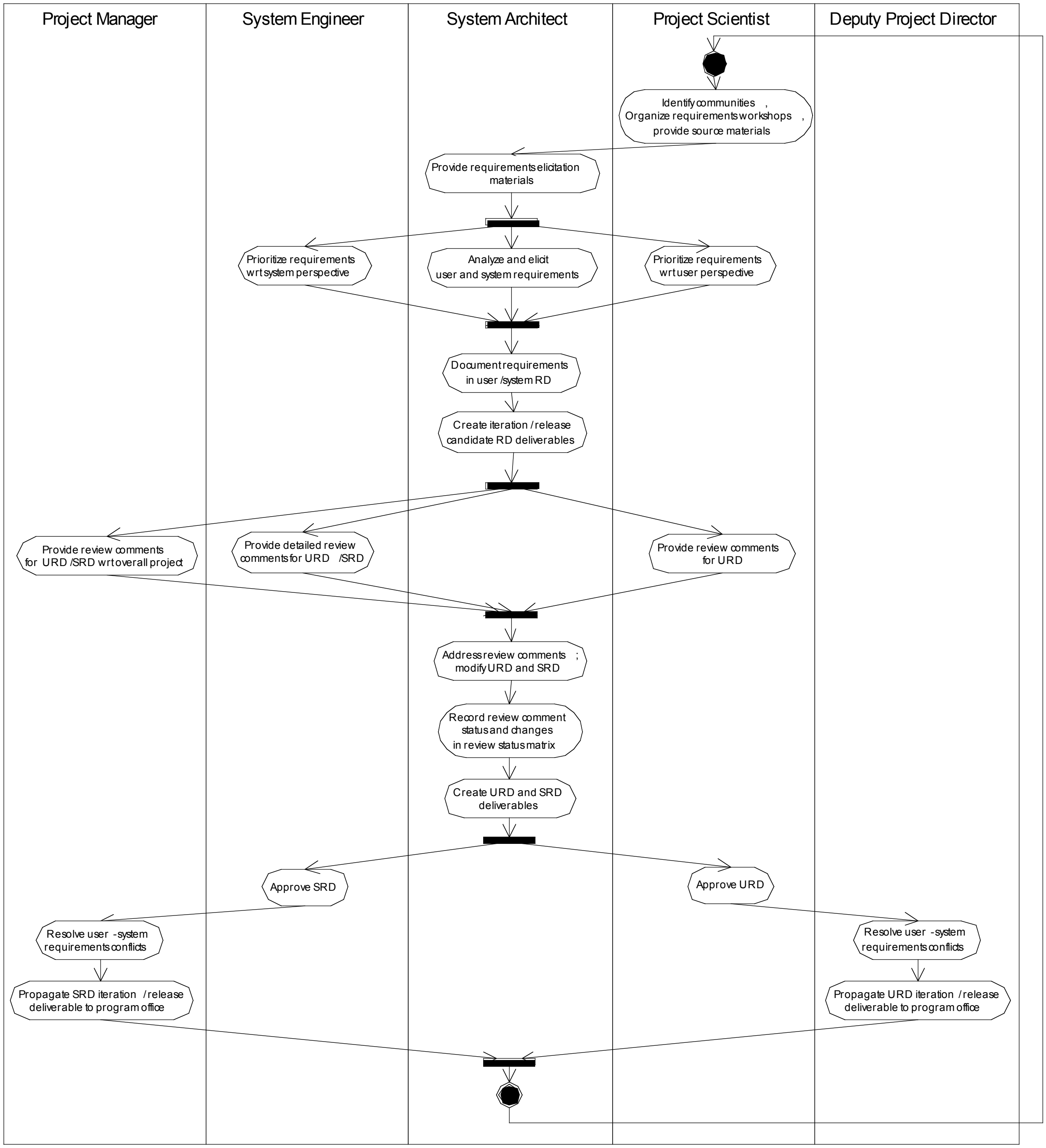

Figure 6 - Requirements Management Process

Chief System Engineer

- Approves requirements management process and plan
- Approves RD template
- Provides source documents and prioritizes their importance/relevance for OOI CI
- Defines system scope
- Provides use case scenarios/workflows at system level
- Supports traceability identification for each requirement

- Prioritizes requirements with respect to necessary level of detail from system perspective
- Supports requirements mediation for conflicting system requirements
- Validates RD
    - Completeness with respect to system scope
    - Completeness with respect to use case scenarios/workflows at system level
    - Utility for transition to design
- Provides process, product and quality level review comments for RD

Project Scientist

- Identifies relevant user communities for OOI CI
- Organize requirements workshops with user communities and participation of Chief System Engineer and Chief Architect
- Provides use case scenarios/workflows at user level
- Supports traceability identification for each requirement
- Prioritizes requirements with respect to necessary level of detail from user perspective
- Supports requirements mediation for conflicting user requirements
- Reviews RD
    - Completeness with respect to use cases/scenarios/workflows at user level
    - Traceability to requirements workshops

Project Manager

- Monitors execution of the requirements management process
- Defines and validates scope and level of detail with respect to budget
- Reviews requirements management plan for coherence with overall project management and system engineering processes
- Reviews RD with respect to overall project vision
- Approves system RD
- Oversees requirements related risk management

Deputy Project Director

- Resolves user-system requirements conflicts that cannot be negotiated
- Approves science user RD

6.2.2 Result Artifacts

The following are the result artifacts produced during execution of the requirements management process.

Requirements Management (RM) Plan:

- Describes the requirements elicitation, analysis, documentation and tracing process
- Describes acceptance criteria and level of detail required for RM artifacts
- Describes requirements validation and QA
- Incorporated into SSEP

Science User and System Requirements Documents

- Documents current situation, goals, project context and requirements

Requirements Document Review

- Provides official comments and input for requirements document.
- All comments will be captured in the review status matrix

Requirements Document Review Status Matrix

- Document under version control as official communication: captures all issues and reviewer comments together with their state and how/when the review comment was/will be addressed

Requirements Traceability Matrix

- Part of requirements document providing traceability to requirements sources and identifying requirements, assumptions and constraints
- Enables traceability to the system architecture

Requirements Elicitation Material

- Questionnaires to be circulated among domain users

Presentation Materials

- Support materials used to describe the process, roles & responsibilities, or artifacts generated during execution of the process.

6.3 Software Architecture and Design

6.3.1 Architecture Development Methodology

Architecture development will follow the process suggested for the Department of Defense Architectural Framework (DoDAF) 1.5, as documented in [DoDAF 1.5 Volume 1], tailored for the needs of the CI project. In particular, this entails development of a sequence of architectural products, structured into:

- All Views Products (AVs): Executive Summary & Glossary
- Operational Views (OVs): Logical Architecture
- System Views (SVs): Deployment Architecture & Implementation Considerations
- Technical Views (TVs): Technical Standards, Constraints and Evolution.

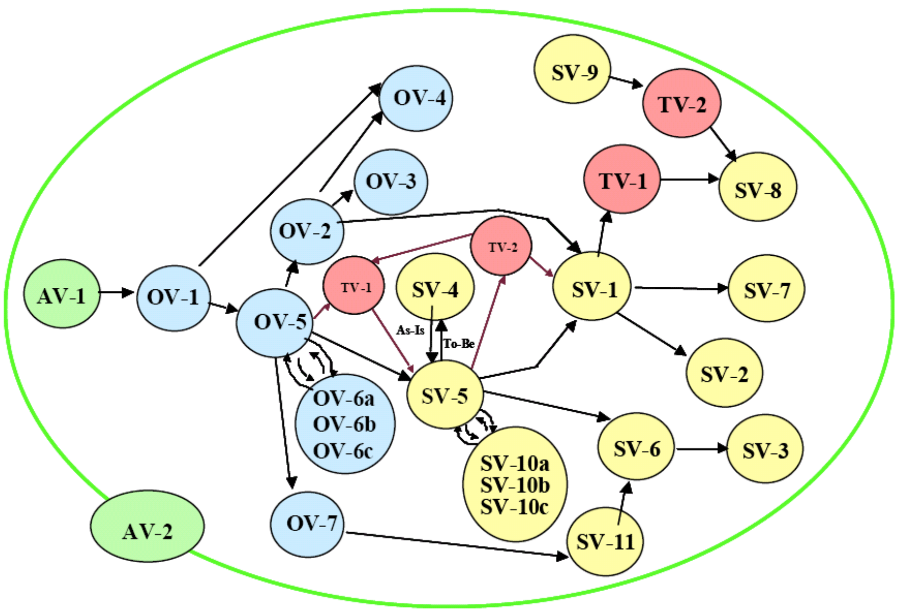

Figure 7: DoDAF Products and Dependencies, DoDAF 1.0 Deskbook, p. 2-5

6.3.2 Architecture Baseline Definition

The ORION CI Conceptual Architecture (CA) forms the baseline for the CI architecture development activity. The pertinent elements from the CI Conceptual Architecture (CA) will be transferred into the CI architecture document and version management system and toolset. This includes, in particular, the CA AVs & OVs, but also the pertinent information contained in the CA requirements document.

6.3.3 Operational and Design Views

Operational and design views are derived from user and system requirements, as well as from the corresponding concepts of operations. The SRD serves as the top level description of desired CI capabilities, and is a key guide for the architecture development process.

The products to be developed for the operational and design views are:

1. High Level Operational Concept Graphic (OV-1)
2. Operational Node Connectivity Description (OV-2)
3. Operational Information Exchange Matrix (OV-3)
4. Organizational Relationships Chart (OV-4)
5. Operational Activity Model (OV-5)
6. Operational Rule Model (OV-6a)
7. Operational State Transition Description (OV-6b)
8. Logical Data Model (OV-7)

The core of the operational views consists of OV-5, OV-6 and OV-7. These products are collectively referred to as the “domain model”. The domain model is the basis for all operational view products (except OV-1) and all system views products.

The architecture team will enrich this list of products to address service-oriented views, including end-to-end policy and governance specifications to the degree they are needed and not explicitly called out in the DoDAF standard.

6.3.4 Deployment and Implementation Views

Deployment and implementation views are covered by the DoDAF systems views. These are derived from user and system requirements, as well as from the corresponding operational and design views.

The products to be developed for the deployment and implementation views are:

1. Systems Interface Description (SV-1)
2. Systems Communication Description (SV-2)
3. Systems-Systems Matrix (SV-3)
4. Systems Functionalities Description (SV-4)
5. Operational Activities to System Functionalities Traceability Matrix (SV-5)
6. Systems Data Exchange Matrix (SV-6)
7. Systems Performance Parameters Matrix (SV-7)
8. Systems Evolution Description (SV-8)
9. Systems Technology Forecasts (SV-9)
10. Systems Rules Model (SV-10a)
11. Systems State Transitions Description (SV-10b)

12. Systems Event-Trace Description (SV-10c)
13. Physical Schema (SV-11)

6.4 Subsystem, Component and Service Breakdown

The CI architecture rests on a rigorous service-oriented design approach to project subsystem capabilities and integration into system capabilities. Service-oriented design and implementation techniques will be leveraged to yield a seamless software and system engineering project framework.

The development effort is structured into seven subsystems (Figure 8) based on the major activities of scientific investigation as well as data and knowledge management and the two infrastructure elements. The subsystems are:

a. Sensing & Acquisition,
b. Analysis & Synthesis,
c. Planning & Prosecution,
d. Data Management,
e. Knowledge Management (not currently funded)
f. Common Operating Infrastructure,
g. Common Execution Environment.

Each subsystem project has a duration of either two or three 16-month development cycles. They are staggered across the five release cycles of the overall proposed project (Figure 4). Their ordering is based on a prioritization of their value to the OOI community and their interdependency. The projects deliver complete subsystems and have a prescribed set of deliverables. These deliverables, starting with the domain models and ending with deployed code, are essential for the long-term viability of the system. Close attention has been paid to the shared architectural elements that can be assembled by one team, enhanced by another and used by most. This strategy of “implement, enhance, use” across teams is employed to ensure sustainability of the interaction interfaces across the duration of the OOI, and drive the quality of the implementations behind them.

In addition to the teams identified for each subsystem, architectural leadership, domain modeling, software design, and user interface resources have been allocated from the System Architecture IPT to work alongside each project. This will ensure coherence of the design and products generated across the projects while giving the proposed work the ability to engage the expertise of the ocean sciences and CI communities in a very targeted and cost effective manner to the OOI vision.

The subsystem IPTs will develop, at a minimum, the following artifacts to prepare integration into the overall systems:

- Requirements & use cases
- DoDAF-style domain models and other pertinent architecture products
- A service framework, consisting of
    - Presentation Interfaces
    - Service patterns
    - Governance logic
- Capability block framework, consisting of
    - Presentation binding

  - Service binding
  - Logic and state
  - Resource binding
- Resource technology descriptions, consisting of
  - System technology
  - Community-supported technologies
  - Externally-provided technologies

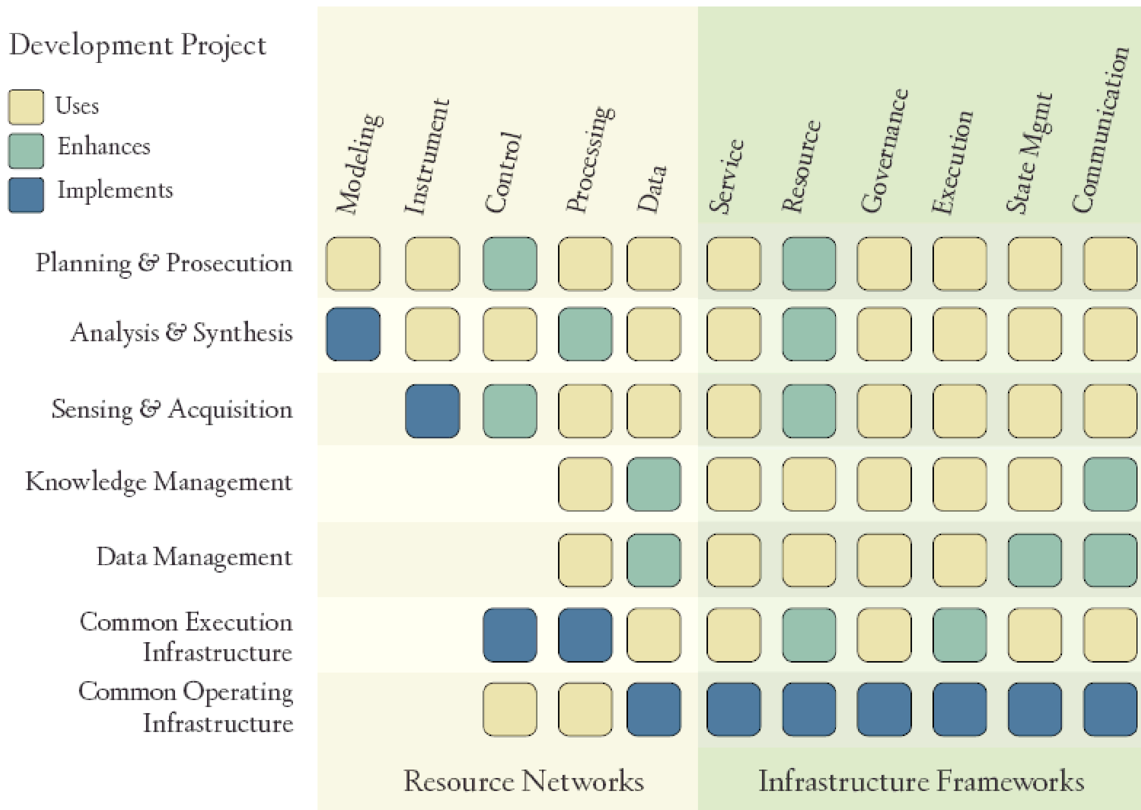

Figure 8: Interrelationship of the subsystems and architectural elements.

Particular emphasis will be placed on the precise specification of Interaction Interfaces (i.e. service interface specifications *including* interaction protocol). This will facilitate the creation of test-plans and inform acceptance decisions to prepare and perform system integration.

6.5 Software Implementation

6.5.1 Methodology

Software Implementation is driven by service-oriented and component-oriented concepts, emphasizing modularization of software units, loose coupling and interface specification.

A given software unit will be developed using a test-first strategy. Tests need to be designed and developed first, according to the software architecture and design.

Rapid prototyping is applied together with frequent requirements/design iterations and software build cycles in order to identify and mitigate implementation risks and obtain stakeholder involvement early in the project.

6.5.2 Tools

All software development occurs under full version control. Software developers can get local copies of the software source code using the version control system, and can build software components as required. Appropriate hardware and middleware installations will be required to deploy and execute the system or parts thereof. Software developers will be able to run automatic unit and integration tests for software units and components on their development workstations.

A testing environment will provide the capabilities for testing individual system components in isolation from the other parts of the system, in order to establish full testability. Test drivers, test harnesses and test containers simulating actual system infrastructure will provide the test environment. Tests are version controlled in a similar manner to software, and need to be automatically executable.

An automatic build environment will be put in place that creates builds of the software on demand and each night. Nightly builds will be subject to automated testing. Build and test results will be published for access by the software development team.

6.5.3 Standards and Technologies

Using the ILS process, the Supportability is responsible for the selection of state-of-the-art industry standards, infrastructure technologies and products as the basis for implementation of the OOI CyberInfrastructure. Such technologies include, but are not limited to, Java SE and EE 6, Enterprise Service Bus (ESB) solutions, ActiveMQ and AMQP messaging middleware, Spring, PicoContainer, Ruby on Rails, and AJAX-enabled Web Frontends.

All source code developed for the OOI CyberInfrastructure must follow industry standard source code format guidelines. In the case of the Java programming language, this will be the Code Conventions for the Java Programming Language, available at http://java.sun.com/docs/codeconv/ .

6.5.4 Unit Testing

Unit testing is required for every software unit and component. Unit tests need to be provided by the software developers using a “test-first” approach, following the software architecture definition and design. Unit tests define the acceptance criteria for each software unit. The Software Development Manager is responsible for checking the existence and quality of the unit tests. Tests should be engineered towards low foot print and automatic execution in order to enable short turn around times. Only software units with appropriate test coverage, passing all tests can be included in software releases.

6.6 Integration, Verification and Validation

6.6.1 Integration

During the construction phase of each development spiral, the newly created architectural elements are integrated into the existing CI release using the roadmap defined in the Integration and Verification Plan. The responsibility for each subsystem lies with its subsystem IPT lead, and the Software Development Manager is responsible for integration of the subsystems into a coherent CI. The subsystem IPT leads are also responsible for any defects pertaining to their subsystems and their timely removal. Integration of subsystems can occur when all tests are passed.

The architecture and design of the subsystems must follow the DoDAF framework and be consistent with the CI overall architecture and design documentation. Particular emphasis must be placed on the design of suitable subsystem interfaces that establish a loose coupling but high coherence.

6.6.2 Verification and Validation

The verification process for the integrated CI is described in the Integration and Verification Plan, and Is carried out by the Chief Architect. Integration of the verified CI with the physical infrastructure produced by the marine IOs is also described in that document, and is carried out by the Chief System Engineer and Operations Manager. Each CI release will be evaluated for correctness, completeness, security, and quality using the criteria established in the Integration and Verification, Security Management, and Quality Assurance Plans, as well as for consistency with the SRD.

6.7 Deployment and Acceptance

The deployment process is described in the Deployment and Acceptance Plan, and is carried out by the Operations and System Integration IPTs with oversight by the Chief System Engineer and Operations Manager. The acceptance process is also described in the plan, and is carried out by the OOI Program Office.

6.8 Operations and Maintenance

The Operations IPT carries out operations and maintenance activities on the CI after deployment and acceptance. The Chief System Engineer and System Architecture IPT, along with the cognizant Subsystem IPTs, will provide necessary support to facilitate the transition to operations during each development cycle transition phase, including the final six month transition phase.

7 Community Participation

7.1 Stakeholder Engagement

An important element of system-level stakeholder engagement is the process of eliciting user requirements from representatives of the science and education user communities at meetings or workshops. The roles and processes for this purpose are described in Section 6.2.

A second key element of stakeholder engagement is the inclusion of selected stakeholders at milestone reviews, especially the annual IOC where the previous software release is evaluated and the requirements and plans for the next one are described. Funds have been allocated to bring 10 members of these communities to the IOC for this purpose, and they will be expected to provide written feedback to the Project Manager.

7.2 User Support and Documentation

User documentation for the CI will be written during development of each subsystem by the cognizant IPT. All user documentation will be freely distributed over a website.

User support will be provided by the Operations Team, and is described in the Operations and Maintenance Plan.

7.3 User Training

User training sessions will be scheduled in conjunction with OOI and general ocean science meetings as demand requires. Training will be carried out by appropriate members of the Operations Team.

Appendix: Abbreviations

Abbreviation	Meaning
AD	Architecture & Design Document
CI	Cyberinfrastructure
CSN	Coastal Sensor Network
DoDAF	Department of Defense Architecture Framework
GSN	Global Sensor Network
OOI	Ocean Observatories Initiative
PD	Project Definition (Phase)
PDR	Preliminary Design Review
QA, QC	Quality Assurance, Quality Control
RD	Requirements Document
RCO	Regional Cabled Observatory
RSN	Regional Sensor Network
TRN	Transition (Phase)
Y1	Year 1