Cyberinfrastructure for the US Ocean Observatories Initiative: Enabling Interactive Observation in the Ocean

A.D. Chave
Woods Hole Oceanographic Institution
Woods Hole, MA 02543 USA
M. Arrott, C. Farcas, E. Farcas, I. Krueger and M. Meisinger
Calit2, University of California at San Diego
La Jolla, CA 92093 USA
J.A. Orcutt, F.L. Vernon and C. Peach
Scripps Institution of Oceanography
La Jolla, CA 92093 USA
O. Schofield
COOL, Rutgers University
New Brunswick, NJ 08901 USA
J.E. Kleinert
Raytheon Intelligence and Information Systems
Aurora, CO 80011 USA

Abstract- **The Ocean Observatories Initiative (OOI) is an environmental observatory covering a diversity of oceanic environments, ranging from the coastal to the deep ocean. Construction is planned to begin in mid-2010 with deployment phased over five years. The key integrating element of the OOI is a comprehensive cyberinfrastructure whose design is based on loosely coupled distributed services, and whose elements are expected to reside throughout the OOI observatories, from seafloor instruments to deep sea moorings to shore facilities to computing and archiving infrastructure. There are six main components to the design comprising the core capability container, consisting of four elements providing services for users and distributed resources and two infrastructural elements providing core services. The Sensing and Acquisition component provides capabilities to acquire data from and manage distributed seafloor instrument resources, including their interactions with the infrastructure power, communication and time distribution networks. The Data Management component provides capabilities to distribute and archive data, including cataloging, versioning, metadata management, and attribution and association services. The Analysis and Synthesis element provides a wide range of services to users, including control and archival of models, event detection, quality control services and collaboration capabilities to create virtual laboratories and classrooms. The Planning and Prosecution element gives the ability to plan, simulate and execute observation missions using taskable instruments, and turns the OOI into an interactive observatory. The remaining elements are the Common Operating Infrastructure that provides core services to manage distributed, shared resources in a policy-based framework. It includes capabilities for efficient and scalable communication, to manage identity and policy, manage the resource life cycle, and catalog/repository services for observatory resources. The Common Execution Infrastructure provides an elastic computing framework to initiate, manage and store processes that may range from initial operations on data at a shore station to the execution of a complex numerical model on the national computing infrastructure, and on compute clouds.**

I. Introduction

Ocean science has long been the franchise of individuals or small groups of scientists working to solve problems within a single science domain at a time. However, the broad scientific and civil demands for multidisciplinary and interdisciplinary research coupled with exponential growth in information technology are irrevocably changing oceanography.

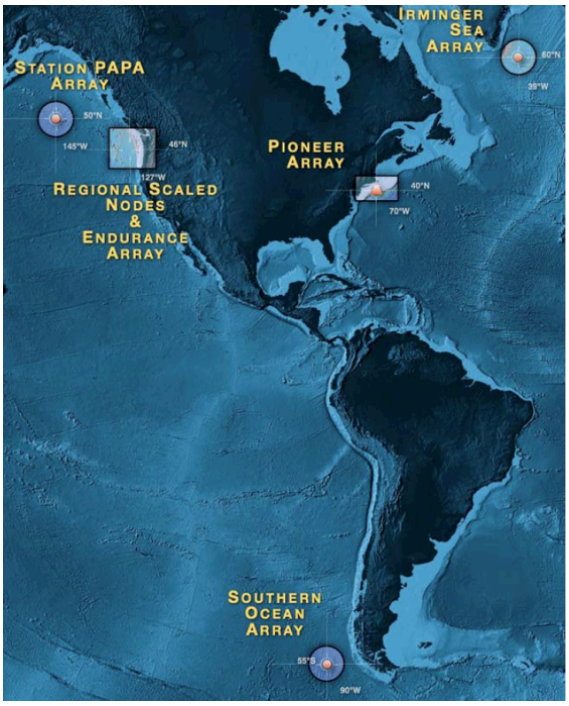


Fig. 1. Location map for the OOI marine infrastructure

The US National Science Foundation is initiating a transformation of ocean science with the Ocean Observatories Initiative (OOI) [1]. The OOI is designed to provide new, persistent, interactive capabilities for ocean science, and has a global physical observatory footprint (Fig. 1). Two implementing organizations (IOs) have been constituted to construct the Regional Scale Nodes (RSN) and Coastal/Global Scale Nodes (CGSN) comprising cabled and multiple buoyed observatories, respectively.

The OOI CyberInfrastructure (CI) constitutes the integrating element that links and binds the physical infrastructure into a coherent system-of-systems, and is being constructed by a third IO. The core capabilities and the principal objectives of the OOI are collecting real-time data, analyzing data and modeling the ocean on multiple scales and enabling adaptive experimentation within the ocean. A traditional data-centric CI, in which a central data management system ingests data and serves them to users on a query basis, is not sufficient to accomplish the range of tasks ocean scientists will engage in when the OOI is implemented. Instead, a highly distributed set of capabilities are required that facilitate:

- End-to-end data preservation and access,
- End-to-end, human-to-machine and machine-to-machine control of how data are collected and analyzed,
- Direct, closed loop interaction of models with the data acquisition process,
- Virtual collaborations created on demand to drive data-model coupling and share ocean observatory resources (e.g., instruments, networks, computing, storage and workflows),
- End-to-end preservation of the ocean observatory process and its outcomes, and
- Automation of the planning and prosecution of observational programs.

In addition to these features, the CI must provide the background messaging, governance and service frameworks that facilitate interaction in a shared environment, similar to the role of the operating system on a computer.

II. Scientific Process Model

From the outset, the OOI CI was designed to implement a set of activities derived from a process model that structures their interaction and compartmentalization (Fig. 2). The model supports the real-time coupling and aggregation of the main *observe, model* and *exploit* activities as well as their respective sub-activities.

The *observe* activity constitutes the data collection strategy that lies at the core of scientific investigation. The outcome is a set of observational products such as data or quantities derived from them that serve as the input to a *model* activity. The result of modeling is a set of derived products that yield an interpretation of the data and the processes that determine them. Based on this improved understanding of the underlying physics, chemistry and biology, the *exploit* activity is used to plan, simulate and execute additional observation activities. The entire scientific process operates as a closed rather than an open loop, with or without a human in the loop.

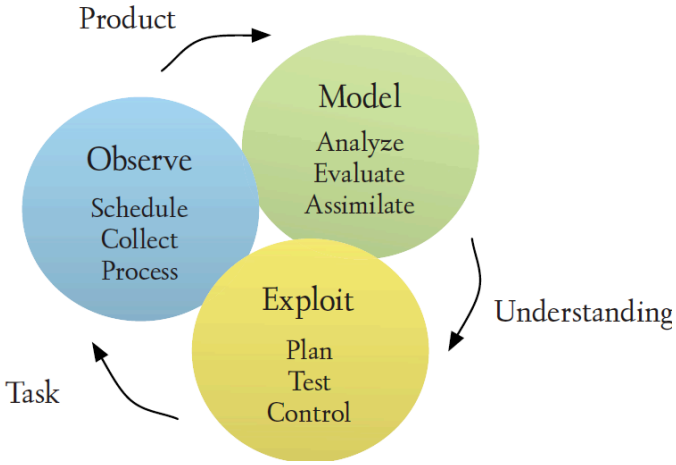


Fig. 2. Scientific process model for ocean observatory activities

III. Fundamental Strategies

The CI integration strategy is based on two core principles: messaging and service-orientation. A high-performance message exchange provides a communication conduit with dynamic routing and interception capabilities for all interacting elements of the system-of-systems. The message interface is isolated from any implementation technologies, and provides scalability, reliability and fault-tolerance. Service-orientation is the key to managing and maintaining applications in a heterogeneous distributed system. All functional capabilities and resources represent themselves as services to the observatory network, with precisely defined service access protocols based on message exchange. Services are also defined independently of implementation technologies. The functional capabilities of the CI are provided by assembling and integrating proven technologies and tools using the integration infrastructure. The integration strategy is an application of the Rich Services pattern introduced in [2].

The CI deployment strategy is based on the concept of a capability container (Fig. 3) that provides all essential infrastructure elements and selected deployment-specific application support. Capability containers can be deployed wherever CI-integrated resources are required across the observatory network, and can be adapted to available resources and their environment. This includes (for example) platform controllers on remote, intermittently connected global moorings, compute units placed in the payload bay of AUVs and gliders and the full range of terrestrial CI deployments (Cyber Points of Presence or CyberPoPs).

The CI multi-facility strategy supports the collaboration of multiple independent domains of authority (Fig. 3) bringing

together their own resources, with no central governance and policy authority, based on agreements and contracts. This enables the sharing and use of observatory resources across the network, governed by consistent policy.

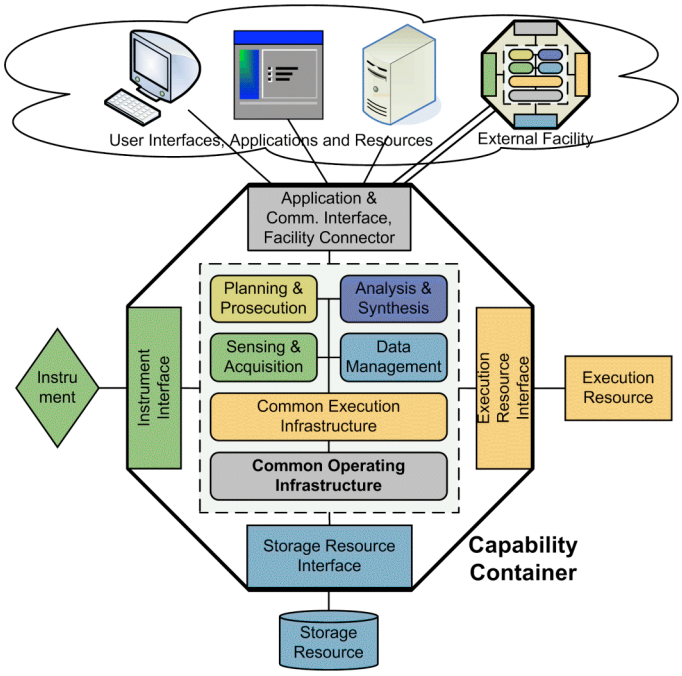


Fig. 3. CI Capability Container showing key external interfaces

IV. SERVICES NETWORKS AND SUBSYSTEMS

The CI's functional capabilities are structured into six services networks (SNs) that participate in operational activities, resulting in services that support user applications. The application-supporting services networks, described in [3], are Sensing and Acquisition, Data Management, Analysis and Synthesis, and Planning and Prosecution. The infrastructure services networks, described in [4], are Data Management, Common Execution Infrastructure (CEI), and Common Operating Infrastructure (COI). Data Management provides both application and infrastructure services.

The services networks and their services are implemented and integrated into subsystems of the same name. Subsystems are the primary units of work breakdown and will be delivered by individual subsystem construction projects.

The Sensing and Acquisition SN (Fig. 3) performs instrument management, mission execution, and data acquisition tasks. The *Instrument* device model consists of one or more physical sensors or actuators, and is represented in the CI by a logical device, the *Instrument Agent*. The physical device provides sensor data and status information to the Instrument Agent, and receives configuration information and commands from it. Each Instrument Agent has an *Instrument Supervisor* that is responsible for monitoring the state of health of the Instrument. Each platform has a *Platform Agent* that is responsible for controlling all devices on that platform. *Observatory Management* is responsible for coordinating all observatory resources to ensure their safe operation. The *Exchange* service is a projection of COI capabilities, with the main purpose of establishing a publish-subscribe model of communication that ensures distributed data delivery to all end-points. It appears in all SNs.

The Instrument Agent is responsible for translating commands from the Platform Agent to the Instrument, event handling, acquiring data from the Instrument, managing the state of the Instrument, providing direct access (e.g., ssh) to the Instrument, and updating the Instrument clock from an external source. The Instrument Supervisor receives status information

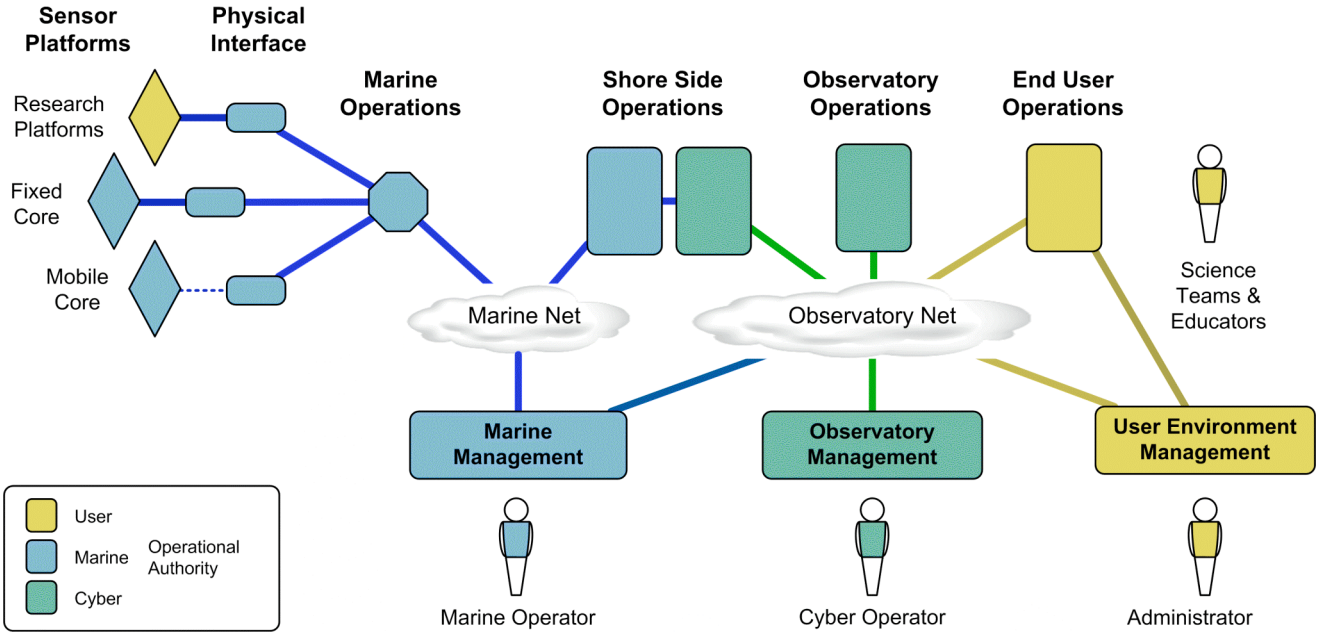
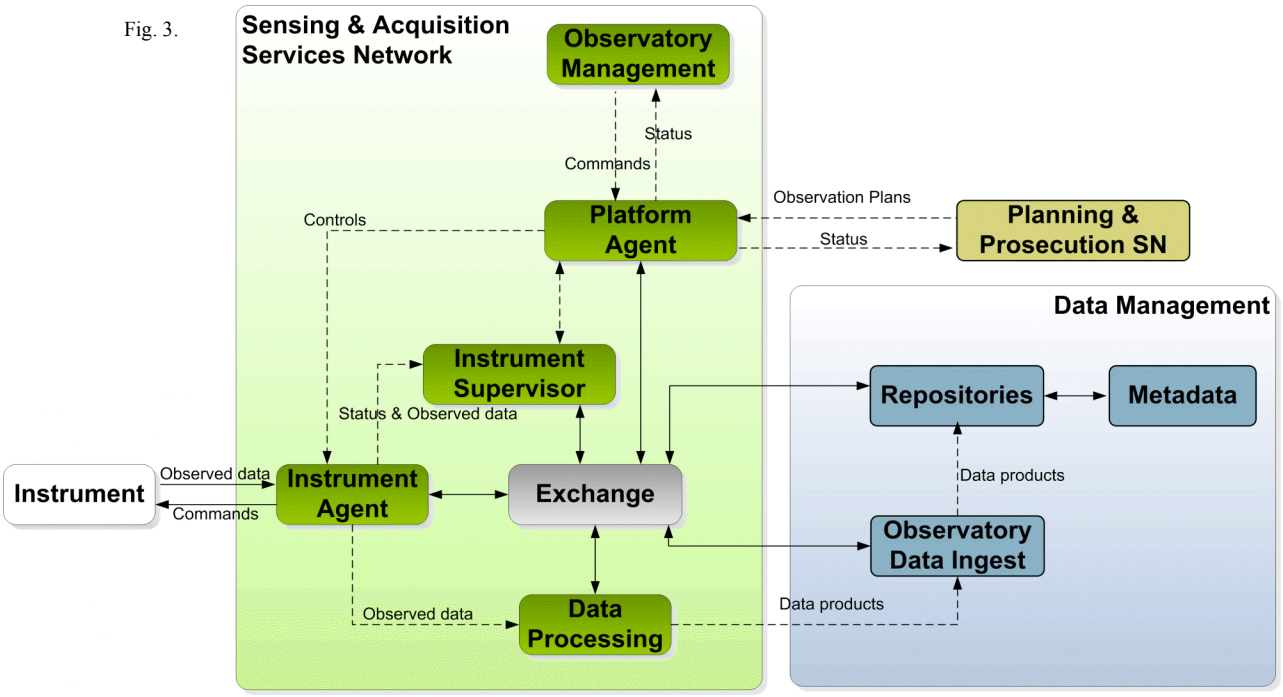


Fig. 3. OOI integrated observatory operational domains

Fig. 3.



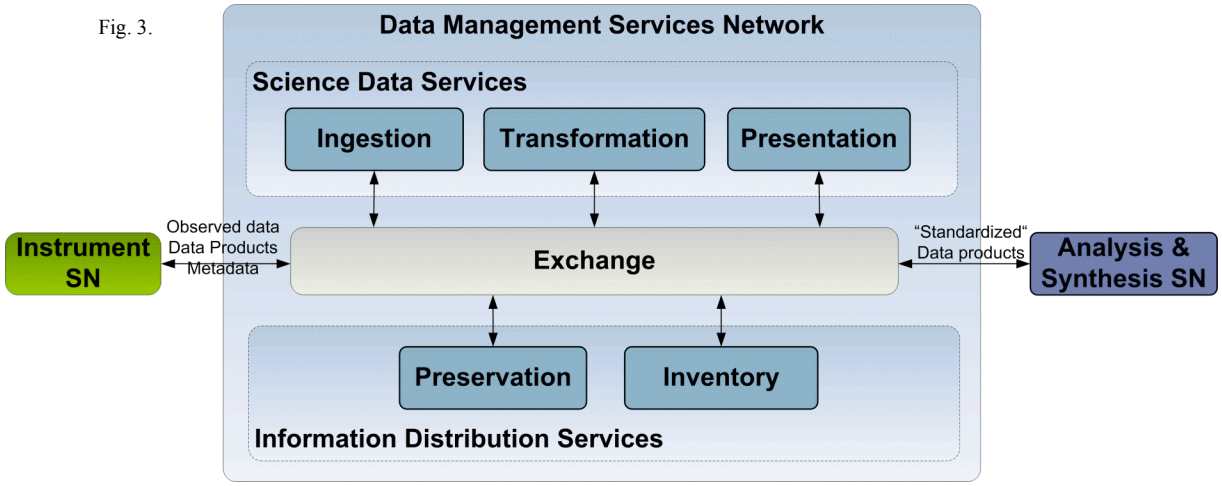
and observed data from the Instrument Agent and performs diagnostics and instrument state estimation. It is responsible for instrument fault detection and recovery. Information about instruments is kept in an *Instrument Repository*; instruments have associated metadata that can be invariant (e.g., model, manufacturer, specs) or variant (e.g., location). Observatory Management is responsible for registering the invariant instrument metadata, whereas the Instrument Agent is responsible for publishing the variant instrument metadata.

The Platform Agent is responsible for registering and validating instruments and detecting resource conflicts at runtime or at registration. It schedules data acquisition as specified in the observation plan produced by the Planning and Prosecution SN or according to the commands received directly from the operators via the Observatory Management node.

Observatory Management protects the assets of the observatory, and therefore is responsible for fulfillment, assurance, and reconciliation. Fulfillment services include resource activation, configuration, simulation, and testing. Assurance services include state of health monitoring, fault detection and recovery and quality assurance. Reconciliation services include billing and resource conflict mediation. Observatory Management also provides users with the capability to configure the calibration of their instruments and the processing steps to produce calibrated and QC'd data products.

The science services of the Data Management SN (Fig. 3) support observational data ingestion into data repositories managed by the infrastructure along with associated metadata. They support syntactical data and information format transformations as well as ontology-supported semantic mediation, providing access to various information products based on metadata and other search criteria.

Fig. 3.



From the point of view of science applications, the critical services are ingestion and transformation. The *Ingestion* service is responsible for initial data parsing, initial metadata extraction, registration, and versioning of received data products. The *Transformation* service manages the data content format conversion/transformation, mediation between syntax and semantics of data (based on ontologies), basic data calibration and QA/QC, additional metadata extraction, qualification, verification and validation. The *Presentation* service enables data discovery, access, reporting, and branding of data products. For data discovery, it provides the mechanisms to both browse/navigate specific data products and search/query of them based on specific metadata or data content.

The science services in the Data Management SN are complemented by the Data Management information distribution services that provide preservation and data distribution capabilities (Fig. 3). Together, they are implemented in the Data Management subsystem. These services provide information distribution, persistence and access across space and time, making data obtainable across the observatory network. Information includes observational data and derived data products along with other information resources required for the operation of the integrated observatory, such as ancillary instrument information, user identities, workflow definitions, and executable virtual machine images. The *Preservation* service is responsible for data replication, preservation, and archival/backup as defined by OOI policies. It also provides distributed data repository capabilities based on the underlying services of the COI subsystem. The *Inventory* service provides the cataloging, indexing and metadata management capabilities required for data ingestion and retrieval.

The Analysis and Synthesis SN (Fig. 3) provides the core services needed for the analysis of observations and their synthesis into derived data products and visualizations. It provides the principal interface to science and education users. Users access the Analysis and Synthesis subsystem through the Interactive Analysis and Visualization services, and interact with the Event Detection Framework and the Data Assimilation and Model Integration Framework. In each case, the user provides specifications and rules, process definitions, and key decisions to the framework, and receives refined datasets, analysis and visualizations.

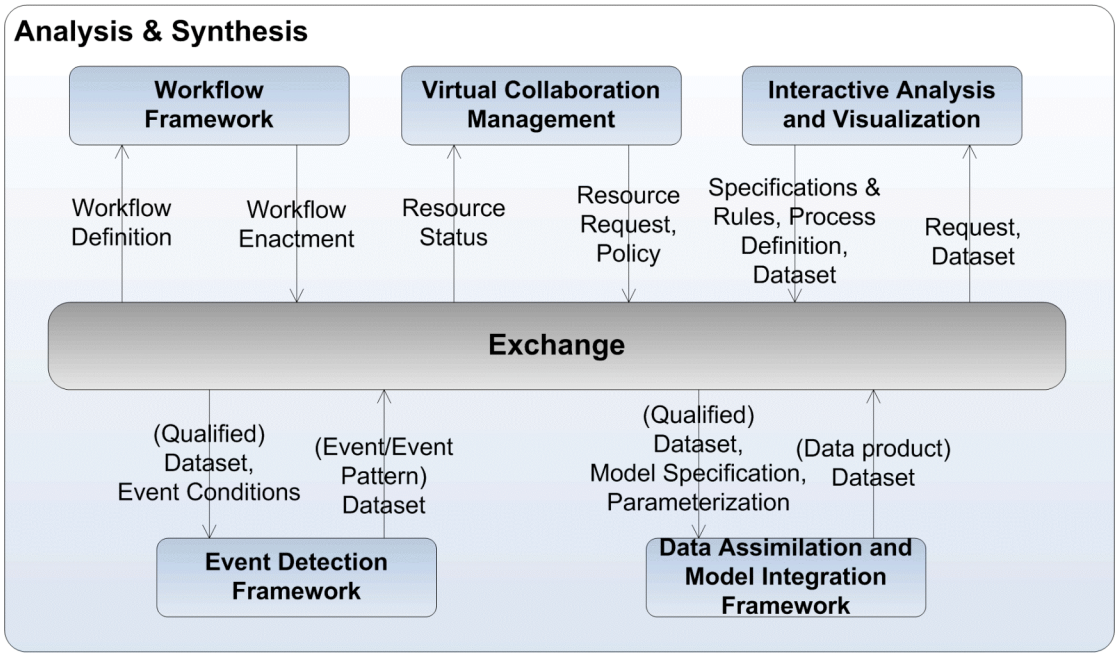
The *Data Assimilation and Model Integration Framework* and the *Event Detection Framework* represent archetypical activities carried out in an analysis and synthesis effort. The Data Assimilation and Model Integration Framework hosts prognostic and retrospective numerical models, usually involving assimilation of real-time or retrospective data from instruments or data repositories, and publishes model products and their descriptors to which interested users can subscribe via the Exchange.

The Event Detection Framework node operates as a filter on real-time or retrospective data to provide detected and classified events as a product. It receives process specifications to establish trigger conditions, definitions and patterns for events, and qualified data or model products via the Exchange.

The *Workflow Framework* provides support for the definition, integration and enactment of user-defined workflows, ranging from human-in-the-loop data QA/QC to event detection and model integration. It provides a workflow engine based on the process execution services of the CEI.

Virtual Collaboration Management provides the foundation for definition of virtual observatories, laboratories and classrooms. This enables a virtual collaboration of multiple individuals in distributed locations in the setting of a defined project using the same set of physical and virtual resources.

Fig. 3.



The capabilities provided include definition of a project by a project lead role, invitation of individuals to collaboration, selection of the resources available to all project members, management of project membership and resource use policy, use of data processing, analysis and visualization tools, the capability to define workflows and the publication of results to the public.

The Planning and Prosecution SN (Fig. 3) leverages the capabilities of the integrated network of sensing, modeling and control resources and supports resource nesting and autonomy. It provides generalized resource planning and control activities that can be applied to plan, schedule, and prosecute multi-objective observational programs.

The *Interactive Observatory Facility* provides the services to design, assemble, and operate configurations of resources from across the OOI into unique systems for planning, testing and prosecuting observation requests, leveraging the nested and autonomous capabilities of the fully integrated network of sensing, modeling, and control resources. It provides experimentalists with services to define, compose, and schedule multi-instrument observations that can execute across the observatory utilizing the services provided by the Analysis and Synthesis SN for virtual collaboration management.

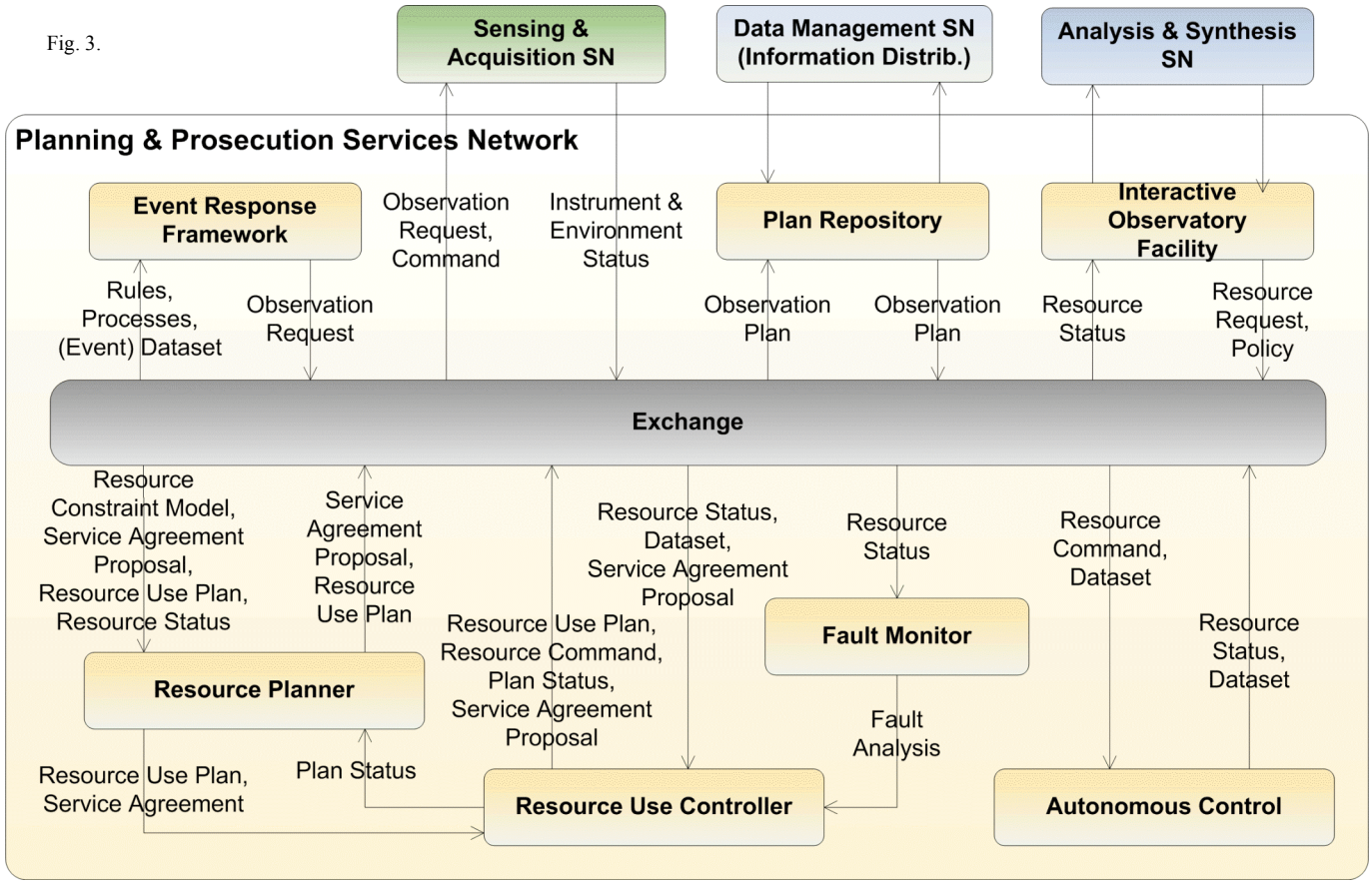
The *Event Response Framework* provides automated and expert (i.e., usually involving user intervention) review of events and processes that may result in responsive tasking or re-tasking of instrument and mobile resources, thus providing observation requests to the *Resource Planner*. It subscribes to rules governing resource usage and information on processes and events from the Data Management SN.

The Resource Planner is a solver based on a resource constraint model. The model abstractly represents resources through their state condition as well as activities that can be performed on the resources. For instance, an AUV's state could consist of the battery charge condition, position, depth, and speed/energy profile. Activities for an AUV could include change position and depth or change to different behavior, such as "return-to-base" from "loitering". The Resource Planner acts as a constraint solver that takes a resource request as input and results in a resource use plan as output.

The *Plan Repository* is a repository instance for storing and managing observation plans and behaviors. Observation plans in the repository can serve as templates that are modified when events response behavior is executed.

The *Resource Use Controller* operates within the framework that the Resource Planner has determined. The Resource Use Controller takes as input a resource use plan and a service agreement, and creates as output resource use plans or specific resource commands to trigger state change and activities in resources. In addition, the Resource Controller can defer the execution of resource use plans within a service agreement to a nested Resource Planner at a lower level.

Fig. 3.



The *Fault Monitor* is a separate component analyzing and overseeing resource status, providing fault analysis input to the Resource Use Controller that in turn can revise plans within their local service agreements or return to the Resource Planner for re-planning.

Autonomous Control provides connection points to local resources, controllers and processes aboard autonomous resource platforms such as AUVs or satellite-connected global moorings. Typically, available bandwidth is severely limited in these cases. Autonomous Control is the gateway to local processes that are not under direct control of the CI.

A broad range of common services is required to bind the CI into a coherent whole. The Common Operating Infrastructure (COI) services integrate the other services networks. They enable data and control distribution among CI services and allow subsystem services to be composed to manage complex interactions. They also implement cross-cutting aspects such as governance and security.

Fig. 3 depicts the COI architecture and services. The *Exchange* service is a fundamental capability of the COI with wide implications for the operation of the CI. It implements the message exchange mechanism between CI services, both within and between SNs. It represents the core integration mechanism for the CI, which is described in more detail in the next section.

The *Identity Management* service provides authentication and supports Policy Management and Governance by implementing authorization. It also participates in the establishment of a federated chain of trust between OOI facilities as well as components of the CI.

The *Governance Framework* enables the implementation of guarded message flows according to specific policies defined by the OOI and its stakeholders. Policies are bound to resources of various kinds. The Policy/Governance and the Identity Management services are tightly coupled through the Exchange service to provide an efficient implementation of policy.

State Management stores and manages all temporary state information about identity management, policy enforcement and ongoing conversations. These services enable the COI to track and enforce the desired behavior of identity management, policy/governance, and conversations (via exchange) through adequate state models.

Resource Management services establish a base for every resource management network in the CI, and are used by all of the SNs. The *Service Framework* stores resources and associates them with their descriptions and relationships with other resource, allowing their discovery and subscription. The *Presentation Framework* provides core services for user interactions with the CI.

The Common Execution Infrastructure (CEI) provides for the virtualization of computing across the OOI, including execution resource provisioning, remote operational management and process execution. It supports software package functional decomposition and deployment, implementation and integration, as well as execution engines and an environment for specific user-requested purposes.

Fig. 3 shows a decomposition of the CEI. The core element is the *Computation Scheduler* that receives Service Agreement Proposals from other services in the OOI that require processing via the Exchange. These Service Agreement Proposals contain the processing request as well as the exact conditions under which the plan can be executed. The Computation Scheduler initiates a negotiation and agreement process with the *Provisioner*, determining a processing plan and initiating the processing by triggering the *Computation*

Fig. 3.

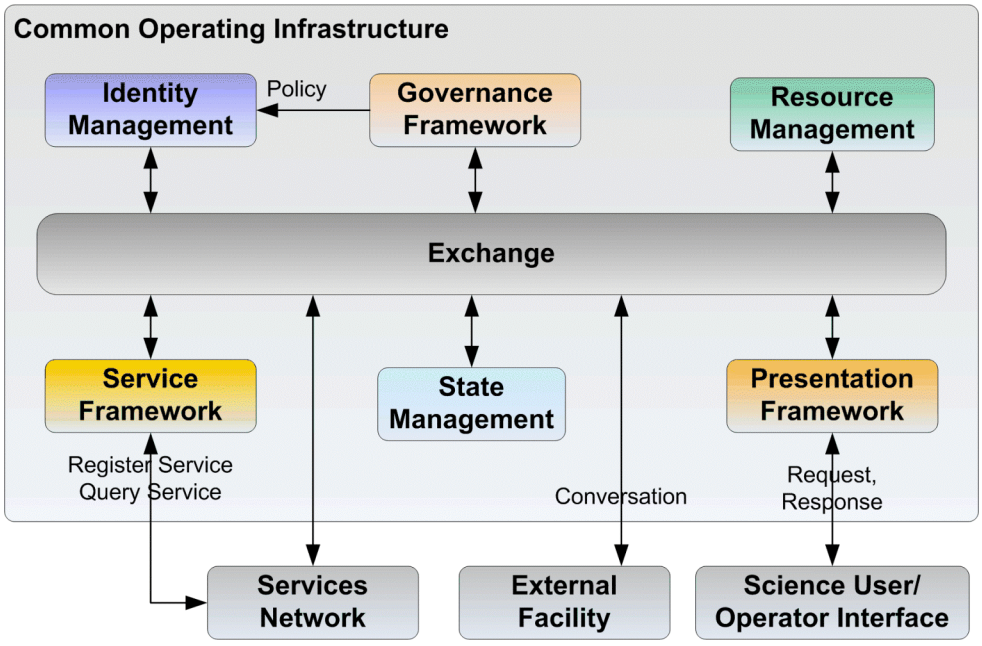
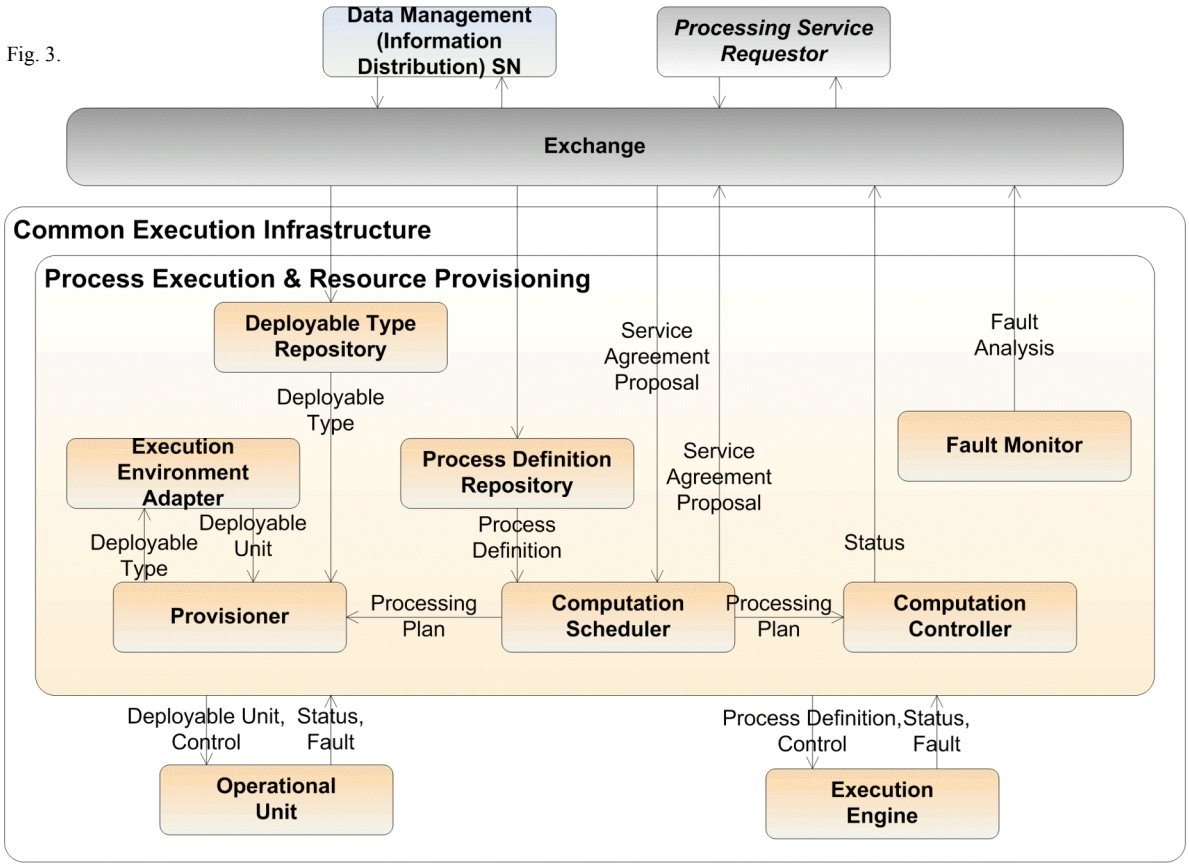


Fig. 3.



Controller. The controller is the entity responsible for enacting the processing plan within the service agreement. It provides status about scheduled and ongoing computations to the Exchange for routing to the service requestor. A *Fault Monitor* is an independent entity monitoring ongoing processes and providing fault analysis information to the service requestor via the Exchange.

There are two distinct entities that the CEI controls. *Operational Units* are independent, self-executing units of processing that can be instantiated on demand by the Provisioner and retired when not needed. Upon receipt of a processing plan, the Provisioner retrieves a machine- and environment-independent representation called a deployable type from the *Deployable Type Repository*. The *Execution Environment Adapter* has knowledge about the characteristics of the intended target execution environment, and automatically performs the adaption, resulting in a deployable unit. The Provisioner can then instantiate it into an operational unit.

Execution Engines are a special kind of operational unit that wait for processes that they can manage. The Computation Scheduler receives process definitions from the *Process Definition Repository* and delegates execution by negotiating with a suitable Execution Engine.

V. SOFTWARE INTEGRATION

The CI integration strategy determines how individual software components integrate into the system-of-systems through a message-broker integration infrastructure. The communication system of the OOI cyberinfrastructure applies messaging as the central paradigm of inter-application information exchange, realizing the Exchange service, the integrating element of all services.

Message-oriented middlewares (MOM) [5, 6] are based on the concept of a message as the exclusive means of information exchange between the distributed components of the system. All information that is passed between two components or services is contained in exchanged messages. Message exchange is asynchronous. The sender of a message does not wait for the message to be delivered or returned; it only waits for the MOM to acknowledge receipt of the message. Addressing messages to recipients utilizes the concept of queues. An application component in a message-oriented architecture only knows the incoming queues that it receives messages from as well as the outgoing queues it delivers messages to, plus the message formats that pertain to these queues. The MOM provides the capability for system integrators to connect these queues to known endpoints in the network; consequently it handles routing, reliable storage and delivery of messages to intended recipients across the network. Standardization is on the way for the underlying message wire transport protocol: the Advanced Message Queuing Protocol (AMQP) [7] defines the interactions of a message broker with

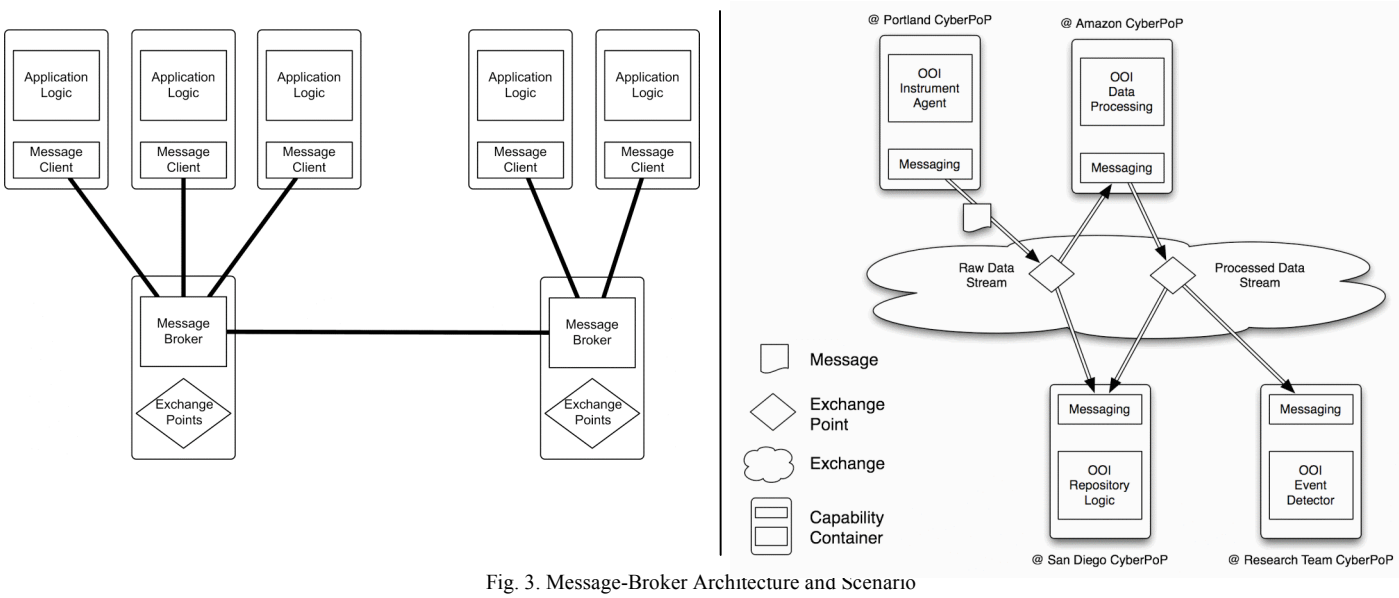


Fig. 3. Message-Broker Architecture and Scenario

its clients, promising interoperability between message brokers of different provenance.

The left part of Fig. 3 depicts the fundamentals of the CI Exchange message-broker architecture. Message brokers realize the central infrastructure elements, represented as Exchange Points to all clients, responsible for the routing and delivery of messages. Message Clients provide the interfaces to the application logic.

The right part of Fig. 3 provides an exemplar application scenario within the OOI CI. Capability containers (CyberPoPs) realize the application logic that interconnects using the message-broker architecture. This is exemplified through an Instrument Agent publishing a raw data stream on an Exchange Point (a queue) via messaging. Any number of consumers may choose to subscribe to such an exchange point. In the example, the data processing application as well as the data repository will receive the published messages. A data stream is a continuous series of related self-contained messages on a given exchange point. There is a second exchange point for another data product containing processed data that is consumed by an event detector process. The physical deployment of all applications is irrelevant. The Exchange realizes all connectivity.

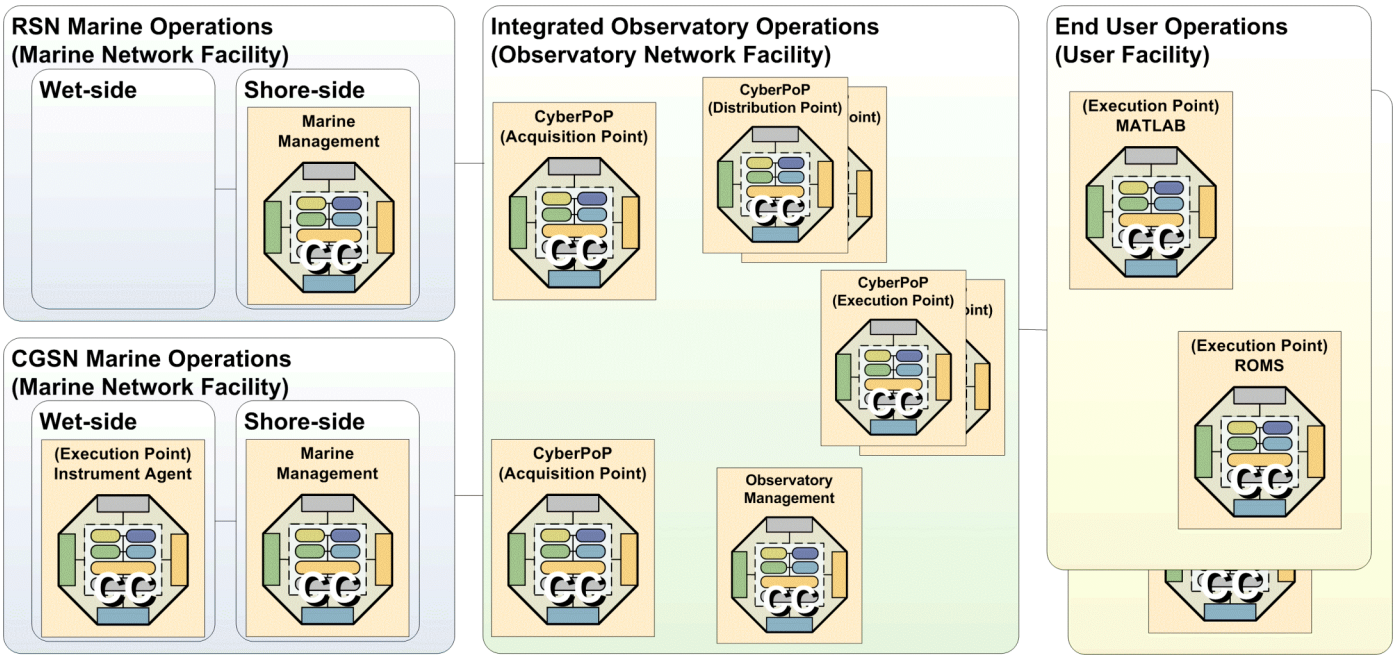
VI. DEPLOYMENT

Error! Reference source not found. depicts a partial deployment scenario for the OOI CI. The primary structured elements represent the CI facilities operated within each of the domains of authority shown in Fig. 3. This includes RSN and CGSN marine operations with wet-side and shore-side components respectively, the integrated observatory facility operated by the CI IO, and further facilities that are connected on behalf of user organizations joining the integrated observatory network based on contractual agreements.

The rectangular shapes show physical deployment nodes within the integrated observatory running partial or full CI software stacks. The octagon shape within each of the deployment nodes indicates that there is an instance of the CI capability container (Fig. 3) deployed with selected infrastructure and application support capabilities, depending on resource availability. Full feature CyberPoPs are deployed on the shore side of each of the marine observatories and within the CI facility. In addition, the CGSN wet-side shows a deployment of a capability container hosting an instrument adapter specific to the instrument platform and their sensors. Capability containers may also be deployed on the RSN wet-side as standalone packages. Within the user facilities, instances of capability containers hosting execution engines for workflows and for executing numerical models are hosted.

All of the above capability containers are connected across the network, and in aggregate present themselves as the OOI integrated observatory. All resources are operated within their respective domains of authority, and policy specific to these facilities applies independent of on whose behalf the resource is used.

The CI applies virtualization of computation and storage in order to realize flexible deployment of capability containers across the OOI network. Implemented and integrated software components are bundled according to specific functional needs into deployable types. The CI manages a repository of such deployable types. In addition, it provides adapters for supported target execution environments. These adapters turn deployable types into deployable units that are specific to one supported target environment. These units can be provisioned as needed and will register themselves in the operational environment where they are deployed (contextualization). Through these steps, the CI achieves independence of software components from specific target environments. In addition, it



can react dynamically to increased resource needs or failures, and bring up new instances (elastic computing).

The described deployment mechanism reaches its full power in combination with cloud computing [8], where virtualized compute and storage resources are provided over the Internet. The above described mechanisms of bundling software components, adapting these bundles to the target environment (e.g. Amazon's EC2 cloud [9]), provisioning such "images" as needed and contextualizing them within an operational context are applied and made available via the CEI. The CI will apply this as an overflow mechanism for user-requested computations. The CEI will then schedule computation and required storage as needed and provision cloud resources that can match user requests. The entire process occurs automatically according to policy defined by the OOI CI operators. Further details are described in [4].

VII. SUMMARY

We have provided an overview of the Ocean Observatories Initiative Cyberinfrastructure (CI) component with its constituent services networks. In total, six services networks support user applications and provide required infrastructure capabilities. In terms of the OOI scientific process model, the Sensing & Acquisition services realize the Observe activity, the Analysis & Synthesis services realize the Model activity and the Planning & Prosecution services realize the Exploit activity. Data Management and infrastructure services support this scientific process that enables interactive ocean observing.

The main strategies that drive the design of the CI have been introduced, and in particular the integration and deployment strategies. Integration is based on high performance messaging and service-orientation utilizing the Rich Services pattern. The deployment strategy makes use of a CI capability container that can be deployed on any computational platform within the OOI network, from deep ocean buoys, to terrestrial data centers to cloud computing networks, The strategy integrates resources and capabilities into the OOI network, thereby forming the OOI Integrated Observatory.

REFERENCES

[1] Ocean Observatories Initiative (OOI). Program website, http://www.oceanleadership.org/ocean_observing

[2] M. Arrott, B. Demchak, V. Ermagan, C. Farcas, E. Farcas, I. H. Krüger, and M. Menarini. Rich Services: The Integration Piece of the SOA Puzzle. In Proc. of the IEEE International Conference on Web Services (ICWS), Salt Lake City, Utah, USA. IEEE, Jul. 2007, pp. 176-183.

[3] OOI CI Integrated Observatory Applications Architecture Document, OOI controlled document 2130-00001, version 1-00, 10/28/2008, available at http://www.oceanobservatories.org/spaces/display/FDR/CI+Technical+File+Repository

[4] OOI CI Integrated Observatory Infrastructure Architecture Document, OOI controlled document 2130-00002, version 1-00, 10/24/2008, available at http://www.oceanobservatories.org/spaces/display/FDR/CI+Technical+File+Repository

[5] G. Banavar, T. Chandra, R. Strom and D. Sturman. A case for message oriented middleware. Proc. of the 13th International Symposium on Distributed Computing, pp. 1–18, 1999.

[6] P.T. Eugster, P. Felber, R. Guerraoui, and A.-M. Kermarrec. The many faces of publish/subscribe. Tech. Rep. DSC ID:2000104, EPFL, January 2001.

[7] Advanced Message Queuing Protocol (AMQP). AMQP Working Group Website http://www.amqp.org/

[8] B. Hayes. Cloud Computing. Comm. ACM, 51(7):9–11, 2008.

[9] Amazon Elastic Compute Cloud (Amazon EC2). http://aws.amazon.com/ec2/.