Creation Review
July 2007

Table of Contents

1. Introduction

OSEE provides a tightly integrated environment that supports lean engineering across the full product life-cycle. Since OSEE integrates all engineering areas, the full life-cycle data for a product is managed by a common platform allowing this data to be seamlessly combined to form a coherent, accurate view of a project in real-time. At the heart of the OSEE architecture are the OSEE core services and application framework which are available to all the OSEE applications and enable their tight integration.

The vision for OSEE finds concrete grounding in daily use by Subject Matter Experts developing next generation mission software for an existing Boeing aircraft. This direct involvement with a full-scale deployment of OSEE allows important user feedback to be rapidly incorporated back into both OSEE itself and as lessons learned for the development team.

Since OSEE has been developed for use by separate groups of end users, careful attention has been given to providing a proper separation of core OSEE capabilities and specific extensions made for various groups. The requirement to support separate groups has helped ensure that OSEE is extensible through standard Eclipse mechanisms. The OSEE architecture is designed to provide an integrated, extensible engineering environment that can be adopted by any group engaged in engineering.

Descriptive Name: Open System Engineering Environment
Acronym: OSEE
Root Package: org.eclipse.osee
Project Proposal: www.eclipse.org/proposals/osee/
Creation Review Comments/Votes: https://bugs.eclipse.org/bugs/show_bug.cgi?id=194250
Newsgroup: news://news.eclipse.org/eclipse.technology.osee
Mailing list: osee-dev@eclipse.org

2. Scope and Objectives

2.1 Objectives

- Provide an extensible application framework and core services which enables a rich and diverse set of applications to share a common persistence layer providing access control, advanced version control, and dynamic object models through standard Eclipse mechanisms while reducing the heavy burden on applications that such capabilities normally entail.

- Provide exemplary applications which demonstrate how the OSEE application framework and services can be used to create tightly integrated applications.
- Provide an integrated environment for Subject Matter Experts to create and utilize data throughout the engineering lifecycle.
- Allow the engineering lifecycle data to be delivered in an environment that supports sophisticated navigation, visualization, and reporting.
- Expand the Eclipse user community to include the broad spectrum of engineering disciplines.
- Foster an ecosystem involving universities and organizations engaged in engineering of products and the customers of these resulting products.

2.2 Scope of Components with Status

The functional description of OSEE is provided in the following listings of its proposed components along with their key capabilities. The overall maturity of each component is indicated by a label of [Planned], [Designed], [Prototyped], [Developed] or [Deployed]. The scope of the OSEE project includes the aggregate scope of its components.

2.2.1 OSEE Application Framework [Deployed]

2.2.1.1 Description

The Application Framework provides a common persistence storage to all OSEE exemplary tools. Its architecture enables an application to be developed and deployed as an extension to OSEE or in place of existing OSEE exemplary tools.

2.2.1.2 Status

The Application Framework has been deployed as a part of a customized OSEE solution including requirements management, configuration management and real-time testing. In addition, the Application Framework is a key component in a maintenance manual application being developed for delivery to the US Army.

2.2.1.3 Key Capabilities

- Dynamic and strongly typed artifact model (persistence layer)
- Bidirectional traceability through typed relations model
- Advanced version control including multi-level branching
- Subject-based and object-based Access Control
- User management and authentication
- Dynamic searching API
- Indexing and tagging services
- Views and editors for the creation, manipulation, and navigation of artifacts, attributes, and relations
- Remote event service for communication and synchronization of OSEE instances
- Rules framework for text processing
- Utilities for plug-in developers
- Scheduling framework
- API for the extension and use of this framework to build tightly integrated applications
- Encryption Utility
- Database Utilities
- Distributed Computing Utilities (Jini/RMI)
- Scheduling Service
- Datastore Adaptor

2.2.2 OSEE Test Environment (OTE) [Deployed]

2.2.2.1 Description

OTE is a powerful test solution within OSEE that integrates with existing Java, C, and C++ development environments to provide a seamless flow between developing, debugging, executing, and dispositioning of tests for complex hardware and software. The user is provided a common interface to the simulated and real-time environments for both functional and unit testing.

2.2.2.2 Status

OTE has been deployed as a part of a customized OSEE solution including requirements management, configuration management and real-time testing. In addition, OTE was also deployed in a separate solution for unit testing.

2.2.2.3 Key Capabilities

- Supports the execution of multiple simultaneous batches within a single workspace
- Built-in help system extended with test manger user guide
- Message system supporting MIL-STD-1553 MUX, serial, wire, Ethernet, and Data Distribution Service (DDS)
- Message GUIs provide monitoring, manipulation, and recording of messaging data
- Utilizes OSEE application framework to provide traceability to software requirements
- Automatic generation of tests and testing support classes, directly from requirements
- Leverages off Java Development Toolkit (JDT) and C/C++ Development Toolkit (CDT)
- Provides remote execution of test programs against target hardware and operating system
- Test results are logged in XML
- Transforms test results via built-in or user supplied XSL Transformations
- Built-in XSL Transformations produce interactive HTML result reports
- A test environment service that provides both soft real-time and simulated capabilities, schedules the periodic execution of simulation components, and manages the I/O and testing resources.

2.2.3 Action Tracking System (ATS) [Deployed]

2.2.3.1 Description

ATS is a tightly integrated change tracking system that manages changes throughout a product's life-cycle. ATS provides integrated change management to all OSEE applications through user customizable workflows. ATS is built on top of a workflow management framework that can provide the same workflow capabilities to other domains such as reviews.

2.2.3.2 Status

ATS has been deployed as follows:

1. To track all changes to OSEE source code and track all user support for OSEE customers.
2. To track over 20 separate tools used in the development of avionics software.
3. To track requirements, flight code and qualification tasks needed to support the development and deployment of avionics software.

2.2.3.3 Key Capabilities

- Built on same OSEE application framework as requirements, code, test development

- Common Workflow Framework that provides for the creation of any number of simple to complex workflow state machine configurations that can work together during the engineering life-cycle
- Workflows are configured through graphical diagrams that ATS uses at runtime
- Configuration of ATS performed through OSEE's common application framework enabling workflows to be created and modified without separate OSEE releases
- Advanced project planning capabilities and release management
- Duplication errors are minimized as items are automatically linked and data is shared
- Menus, Views and Editors give access to ATS while working in any other aspect in OSEE
- ATS is used to track changes and support issues for the development of OSEE itself
- Bug Icon allows quick Action creation against any OSEE integrated tool

2.2.4 Define [Deployed]

2.2.4.1 Description

Define provides requirements and process management with tightly integrated change management using the Action Tracking System (ATS). OSEE provides publishing capabilities that enable the creation of documents from smaller internal documents to the complex documents needed to meet military requirements for contract deliverables such as the Software Requirements Specification (SRS), System Performance Specification (SPS), Prime Item Development Specification (PIDS), and all the required traceability between them.

2.2.4.2 Status

OSEE Define has been deployed as a part of a customized OSEE solution including requirements management, configuration management and real-time testing and is currently being evaluated by the same program to manage processes.

2.2.4.3 Key Capabilities

- Enterprise support for concurrent, distributed requirements development
- Integrated process and workflow
- Programmatic, bidirectional traceability
- End user navigation and search capabilities
- Capture accurate, meaningful review metrics

- Tight integration with lifecycle tools
- Automated change detection capabilities

2.2.5 BLAM [Developed]

2.2.5.1 Description

BLAM Lightweight Artifact Manipulation (BLAM) allows non-programmers to graphically construct workflows to automate repetitive tasks. A given workflow can be used for variety of similar tasks by using customizable controls to specify workflow parameters.

2.2.5.2 Status

BLAM is being developed as a common part of OSEE and is to be deployed as part of a customized OSEE solution including requirements management, configuration management and real-time testing.

2.2.6 Program/Project Management [Planned]

2.2.6.1 Description

Program and project management tightly integrated with the Action Tracking System and other OSEE components to provide services necessary for estimation, planning, execution, and delivery of products managed within OSEE.

2.2.6.2 Key Capabilities

- Integrated management of charge/cost accounting
- Build planning and execution
- Reporting services
- Rules framework for requiring/alerting certain conditions
- Scheduling services for automating reoccurring tasks

2.2.7 Discovery and Learning [Planned]

2.2.7.1 Description

Services provided for allowing advanced learning and discovery using OSEE's abundant and inherent life-cycle data and metrics.

2.2.7.2 Key Capabilities

- Discovery of inefficiencies in life-cycle processes
- Advanced data mining and data fusion
- Advanced estimating
- Advanced export/import of product capabilities between programs including applicable design, requirements, code, and test
- Advanced data visualization
- Prediction of future risks
- Simulation of recommended process and life-cycle changes

2.2.8 Application Development [Planned]

2.2.8.1 Description

Provide capabilities needed for external software application development plug-ins, like JDT, to utilize the OSEE persistence layer and integrate with other OSEE-based applications.

2.2.9 Design and Modeling [Planned]

2.2.9.1 Description

Provide capabilities needed for external design and modeling plug-ins to utilize the OSEE persistence layer and integrate with other OSEE-based applications.

3. Mentors

- **Mark VandenBrink** – Chief Architect for systems software at Motorola
- **Richard Gronback** – Chief Scientist at Borland Software Corporation and GMF Project Lead

4. Architecture

4.1 OSEE Application Framework

The Application Framework layer provides common services, utilities, and capabilities for all OSEE exemplary applications.

4.2 OSEE Exemplary Applications

As a full engineering environment, the goal of OSEE is to provide exemplary applications to support the full life-cycle of engineering. Although categorized in the below architecture diagram as Application Development, Action Tracking System, Systems Engineering, Requirements Management and Testing Environment, these applications and future applications can span any number of these or provide new capabilities such as Discovery and Learning.

4.3 OSEE Extensibility

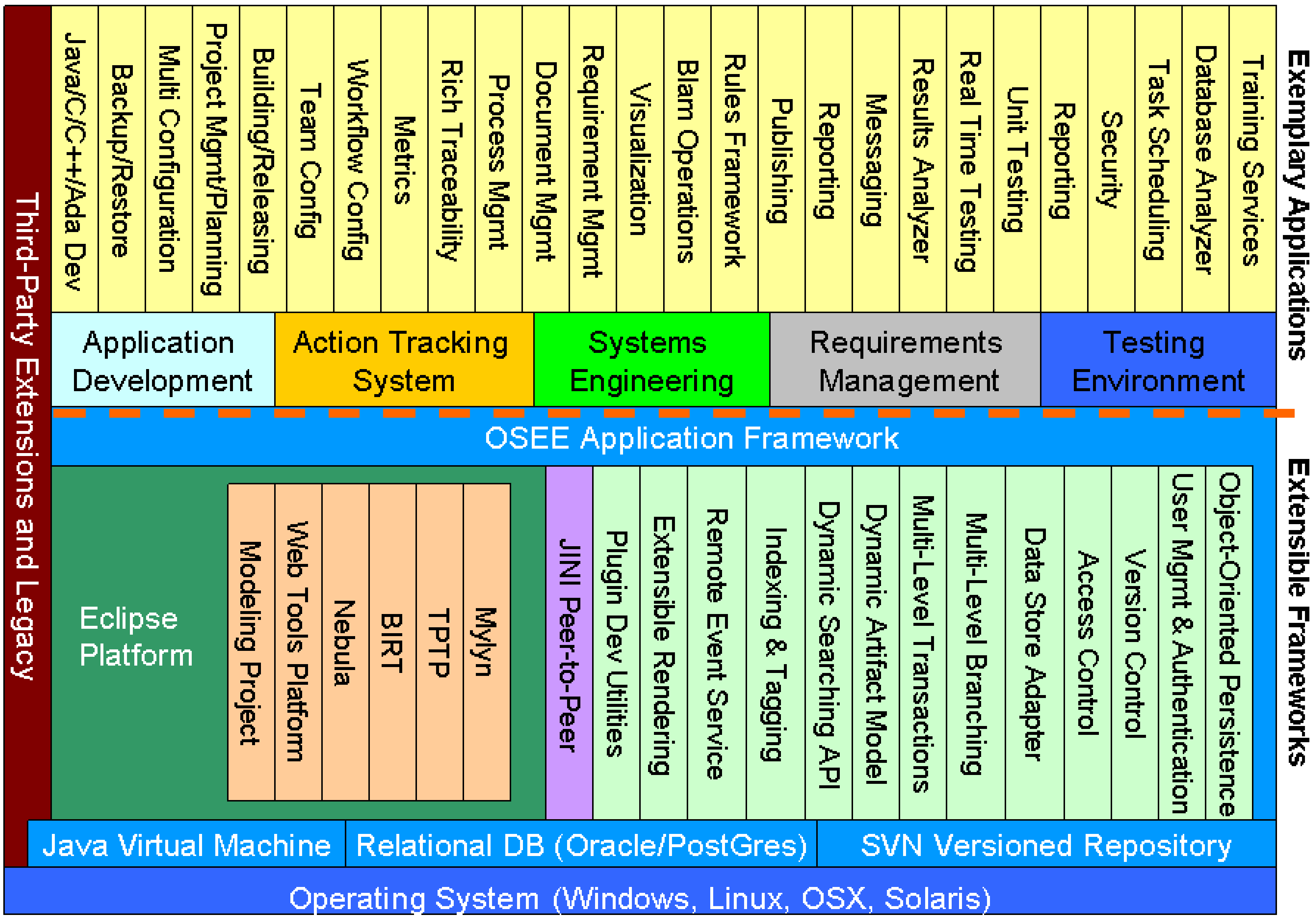
OSEE's Exemplary Applications, as is the entire OSEE framework, are designed with extensibility in mind. Since all possible needs of engineering can not be foreseen and all possible exemplary applications have not been integrated on OSEE's Application Framework, OSEE provides the capability to add other 3rd party plugins through Eclipse's standard plugin framework.

4.4 OSEE Components

See “Scope of Components with Status” section.

4.5 OSEE Architecture Diagram

See Next Page...


Third-Party Extensions and Legacy
Java/C/C++/Ada Dev
Backup/Restore
Multi Configuration
Project Mgmt/Planning
Building/Releasing
Team Config
Workflow Config
Metrics
Rich Traceability
Process Mgmt
Document Mgmt
Requirement Mgmt
Visualization
Blam Operations
Rules Framework
Publishing
Reporting
Messaging
Results Analyzer
Real Time Testing
Unit Testing
Reporting
Security
Task Scheduling
Database Analyzer
Training Services
Exemplary Applications
Application Development
Action Tracking System
Systems Engineering
Requirements Management
Testing Environment
OSEE Application Framework
Eclipse Platform
Modeling Project
Web Tools Platform
Nebula
BIRT
TPTP
Mylyn
JINI Peer-to-Peer
Plugin Dev Utilities
Extensible Rendering
Remote Event Service
Indexing & Tagging
Dynamic Searching API
Dynamic Artifact Model
Multi-Level Transactions
Multi-Level Branching
Data Store Adapter
Access Control
Version Control
User Mgmt & Authentication
Object-Oriented Persistence
Extensible Frameworks
Java Virtual Machine
Relational DB (Oracle/PostGres)
SVN Versioned Repository
Operating System (Windows, Linux, OSX, Solaris)

5. Initial Committers

Name	Org.	Role	Experience and connection with the OSEE project
Ryan D. Brooks	Boeing	Co-project Lead, Committer	Ryan is the OSEE co-team lead and is the lead and chief architect of the OSEE Application Framework and OSEE Define. He has been an Eclipse plug-in developer for 3 years and Java developer for 10 years.
Donald G. Dunne	Boeing	Co-project Lead, Committer	Don is the OSEE co-team lead and is the lead and chief architect of OSEE ATS. He also develops portions of the OSEE Application Framework. He has been an Eclipse plug-in developer for 4 years and Java developer for 5 years.
Andrew M. Finkbeiner	Boeing	Committer	Andrew is a member of the OSEE Team and is lead and chief architect of the OSEE Test Environment. He also develops portions of the OSEE Application Framework. He has been an Eclipse plug-in developer for 3 years and Java developer for 8 years.
Jeff C. Phillips	Boeing	Committer	Jeff is a member of the OSEE Team and is a developer with a focus on the OSEE Application Framework and OSEE Define. He has been an Eclipse plug-in developer for 3.5 years and Java developer for 4.5 years.
Ken J. Aguilar	Boeing	Committer	Ken is a member of the OSEE Team and is a developer with a focus on OSEE Test Environment and the use of the OSEE framework for the creation and deployment of tailored RCP applications. He has been an Eclipse plug-in developer for 2 years and Java developer for 4 years.
Michael P. Masterson	Boeing	Committer	Michael is a member of the OSEE Team and is a developer with a focus on OSEE Test Environment. He has been an Eclipse plug-in developer for 2 years and Java developer for 6 years.
Robert A. Fisher	Boeing	Committer	Robert is a member of the OSEE Team and is a developer with a focus on the OSEE Application Framework and OSEE Define. He has been an Eclipse plug-in developer for 3 years and Java developer for 4 years.
Roberto E. Escobar	Boeing	Committer	Roberto is a member of the OSEE Team and is a developer with a focus on the OSEE Application Framework. He has been an Eclipse plug-in developer for 2 years and Java developer for 5 years.

6. Interested Parties

6.1 Arizona State University

- Huan Liu, Ph.D.
- James S. Collofello, Ph.D.
- Peter Wonka, Ph.D.
- Susan D. Urban, Ph.D.

After face-to-face meetings and other discussions with ASU, the department of computer science and engineering sent Boeing a letter regarding collaborating on OSEE which stated:

“We, professors in the department of computer science and engineering, are enthused to learn your very ambitious vision for the Open System Engineering Environment (OSEE), which has gone beyond a specific use case or project at hand in the very beginning at Boeing and aims to foster collaboration.”

“We share your vision for OSEE and are enthusiastic to collaborate with the counterparts at Boeing in mutually interested research areas. We firmly believe that our collaborations on OSEE will not only advance the state-of-the-art OSEE, but also eventuate palpable results/systems with practical impact.”

6.2 Auburn University

- David Umphress, Ph.D.
- Dean Hendrix, Ph.D.
- James H. Cross II, Ph.D.

Auburn has expressed interest in several areas of collaboration. The most immediate opportunity is for Auburn's 7000-level graduate course, “Software Environments”, to utilize OSEE and develop plug-ins that could be contributed to the OSEE project.

6.3 EADS

- Andreas Keis

EADS has expressed great interest in OSEE especially since many of OSEE current use cases align well with their needs. During a conference call with EADS, the OSEE Team provided an overview of OSEE and EADS provided the OSEE Team with an overview of TOPCASED. Additional communication on possible collaborations are ongoing.

6.4 General Motors

- Srinivas Chande

Multiple groups inside of General Motors including EI&S Powertrain and the Vehicle Group are interested in potentially using OSEE on their project. The OSEE Team has provided a demo of OSEE to members of these groups. Based on this demo and the associated discussions, GM is planning to setup a follow-up meeting with a much wider audience (management level) to further explore possible collaborations.

6.5 Lockheed Martin, Advanced Technology Laboratories

- Alexei Samoylov

Lockheed is in the initial stages of learning about OSEE. Initial discussions have focused on the fact that Lockheed has many Subject Matter Experts that develop complex systems and the OSEE capabilities that support this.

6.6 Rockwell Collins

- Paul Streit

Rockwell Collins' Enterprise Tool Integration department is exploring the possibility of making OSEE part of their enterprise tool strategy. The OSEE Team has provided several demos for Rockwell Collins, and some of the resulting discussions have included how they might contribute to the development of OSEE.

7. Relationship to other Eclipse projects

- The Java Development Tools (JDT) and Plugin Development Environment (PDE) are depended upon for the development of the OSEE Application Framework itself and by adopters extending the OSEE Application Framework.
- OSEE applications utilize custom SWT widgets from the Nebula project.

- Business Intelligence and Reporting Tools (BIRT) are used to view test result summaries from the OSEE Test Environment.
- The Graphical Editor Framework (GEF) is used by a graphical view showing the decomposition of an Action in the Action Tracking System.
- The Zest visualization toolkit from the Mylyn project is used to provide general purpose data visualization capabilities showing artifacts and the relations between them.
- The OSEE team uses Subversive as their SVN Team Client for version control of the OSEE source code.
- C/C++ Development Tooling (CDT) provides the C/C++ IDE to OSEE end-users.

8. Code Contribution

The Boeing Company is offering a suite of Eclipse features (and their associated plug-ins) corresponding to 5 of the proposed components of this project as the initial code-base, namely: OSEE Application Framework, OSEE Test Environment, Action Tracking System, Define, and Program/Project Management.

9. Initial Roadmap

The Open System Engineering Environment (OSEE) Project is proposed as a sub-project under the Eclipse Technology Project. Upon demonstration of the characteristics of a Top-Level Project and the successful completion of a Promotion Review we envision the OSEE project being promoted to a new Top-Level Project.

The project will continue to function under an agile development process with milestone releases every 4-6 weeks.

Date	Version	Description
2007 - Jul		Project Creation Review
2007 - Aug	0.80.x	Sanitize source code and initial contribution of code
2007 - Sep	0.81.x	Demonstration version released with sample database
2007 - Oct	0.82.x	Add additional extension points; Document extension points
2007 - Nov	0.83x	Incorporate first round of improvements requested by Eclipse community
2008 - Jan	0.84.x	Public release for EclipseCon 2008 presentations