

DRAFT 02 - Getting Started with EclipseRT

Nov 24th, 2009

- 1. **DRAFT 02 - Getting Started with EclipseRT** 1
- 2. **Overview** 1
- 3. **Desktop Applications** 1
- 4. **Web Applications** 2
- 5. **SOA**..... 2
- 6. **Enterprise Applications** 3
- 7. **Platforms** 3
- 8. **Mobile & Embedded Systems** 4
- 9. **Conclusion - Eclipse Runtimes**..... 4

Overview

Eclipse is well known as a source of trusted development tools, but it also provides a wide selection of runtime containers, middleware, and enterprise frameworks used to build many types applications. These Eclipse runtime projects are know as EclipseRT which provides the focal point for Eclipse's application development stack for building rich desktop, web, enterprise, and embedded applications, as well as service-oriented architectures (SOA) and applications platforms.

EclipseRT is based on [the concept of component-oriented development and assembly, or CODA](#). EclipseRT provides a growing portfolio of capabilities and open source projects that can be extended and assembled into different types of applications. In this document we will show how EclipseRT can help you apply the principles of CODA and the EclipseRT capabilities to some of the more popular application architectures: desktop applications, web applications, SOA, enterprise applications, platforms and mobile/embedded systems.

Desktop Applications

Desktop applications provide a rich, rapid-response graphical interface for end users. This type of applications also allows for easier access to the desktop resources and allow for an off-line mode.

Starting with Eclipse Rich Client Platform (RCP), EclipseRT provides a modern desktop framework with the rich and powerful features required for high productivity applications. For example, RCP has been used by companies like IBM to deliver its highly dynamic and network-dependent applications in its [Lotus productivity portfolio](#) used as a core business tool daily in many companies.

Extending the success of RCP, the Riena project provides a complete framework for creating next generation client/server applications. In addition to making GUI development with RCP easier, Riena provides the communications back-end needed to create and run connected desktop applications. The reporting framework Eclipse BIRT is commonly used in desktop applications for enterprise grade reporting. For creating graphical editors, EclipseRT provides the Graphical Editing Framework (GEF) and Eclipse Modeling Framework (EMF), which is invaluable for creating custom, graphic-heavy user interface components like circuit

diagrams.

The p2 provisioning platform in Equinox provides the runtime updating framework needed to support frequent feature delivery. More than just providing a place to click and download an installer for a new version of the application, Equinox-driven desktop applications can update select components with p2 over the network resulting in easier to manage update cycles.

Web Applications

Web applications are clearly one of the favorite delivery models for consumer oriented applications. While web applications are quick and easy to create, as they grow the demands to extend and maintain the application can quickly overwhelm the team. EclipseRT provides the reliable, but flexible frameworks for web user interfaces, back-ends, and the tools that support a healthy software development process.

Eclipse provides a light-weight, high-performance web server, Jetty. Widely used in many domains, [such as Google, HP, Cisco, and Atlassian](#), as a web container in development and production. During development, Jetty's easy configuration management and fast start-up time enables rapid feedback loop code-build-deploy so fast in fact that we eliminate the "build" and "deploy" parts and just run right out of the workspace test cycles, allowing developers to stay focused on the immediate task. In production, Jetty provides a diverse array of performance profiles for different scenarios, providing a complete runtime for web applications. Jetty integrates with popular frameworks such as Spring, JBoss, Glassfish, ActiveMQ, and standard Java web technologies. And, of course, Jetty is tightly integrated with the rest of EclipseRT, such as the Eclipse Rich Ajax Platform, EclipseLink and Equinox.

The Eclipse Rich Ajax Platform (RAP) is not only a runtime for running web applications, but a complete framework and tool-set for developing Ajax applications. RAP is tuned to work with Jetty, Tomcat, or any standard web container. In addition to web applications, RAP builds on SWT and the Eclipse UI framework meaning that you can write the application UI once and have it rendered as web and desktop UI.

A rich UI is vital for any contemporary web application, but access to all types of data and processes is required as well. EclipseLink can be used for integrating data mapping into a web applications and Swordfish for SOA infrastructure integration.

SOA

EclipseRT contains a SOA framework called Swordfish. Built on proven open source components such as [Apache ServiceMix](#) and [Apache CXF](#), Swordfish provides an extensible framework that allows application developers and system integrators alike to build their own ESB that can be tailor-made to their requirements. Other components include a process engine provided by [Apache ODE](#) for BPEL support, monitoring, a service registry, and the configuration stores required to run an ESB-driven SOA. The components in Swordfish come together to provide a extensive platform for creating and then running service oriented architectures.

Key to Swordfish's success has been the tight integration with the Eclipse tool chain and developer environment. Rather than having to learn new tools and then switch between

those and their Eclipse IDE, Swordfish lets developers start and then stay in the environment they know and work in best. The platform nature of EclipseRT means that SOAs built on Swordfish can also use other EclipseRT components such as EclipseLink and BIRT for the usual enterprise application needs.

[Ian to add ECF text here]

Enterprise Applications

EclipseRT's component-driven nature makes it an extremely good fit for delivering and maintaining large, modern enterprise applications. IT departments are expected to deliver applications more frequently, delivering on the promise of agile software development. As business requirements evolve at a rapid clip, software must change and move at the same fast pace. At the same time, development teams must provide the same stability, controls, and forward looking design that comes from good enterprise architecture.

Starting with Equinox's component-driven model, EclipseRT provides the foundation for an enterprise architecture, allowing you to specialize the enterprise stack to your application's needs. This [component-oriented](#) approach allows you to spend less time carving down bulky runtimes, instead focusing on the application at hand.

The EclipseRT portfolio of capabilities allows developers to easily be extended and assembled different frameworks to satisfy the infrastructure needs of the enterprise applications. EclipseRT provides capability that includes:

- UI layers with RCP, RAP, or Rienna.
- Runtime containers such as Jetty and Equinox
- Data persistency with EclipseLink
- Data reporting support from BIRT.
- Data modeling with Eclipse Modeling Framework (EMF)
- Remote communication and distributed OSGi with ECF
- SOA integration with Swordfish
- Identity integration and management with Higgins
- Semantic search capabilities on unstructured data with SMILA

The combination of the EclipseRT projects and Eclipse tooling make it the ideal choice for developing and deploying enterprise applications.

Platforms

Larger organizations and teams working together and become more efficient when they consolidate their architecture, services, and code around a shared platform. These platforms then become a shared set of services and frameworks used by others to build applications and allow developers to focus on their business logic rather than the general purpose underpinnings.

Eclipse itself is an example of a platform for creating tools and runtime frameworks. As such, it's little wonder that many of the early users EclipseRT have built platforms that allow them to rapidly deliver applications to their customers and end-users.

For example, NASA, the US space agency, uses EclipseRT for its Ensemble platform. This platform is used by the different science teams to write the applications needed to control

and run the experiments on their Mars rover operations. Instead of each team writing their software from the ground up, they build on-top of the EclipseRT-based Ensemble platform, to accelerate their software development efforts, create a consistent end-user view of these applications and provide an interchange between applications.

The component-oriented nature of EclipseRT creates a technology base that's designed to be extended, to be a platform. Architects can use the design and policy enforcement aspects available in EclipseRT to architect proper avenues for extension and application development in their organization. Because the platform is built on-top of Eclipse technologies, developers can use their familiar tools and frameworks when developing for the platform instead of being forced to use custom platform tools. The management services available in EclipseRT, such as p2, EclipseLink, and Swordfish gives IT the ability to deploy and then maintain applications built on the platform.

Mobile & Embedded Systems

While EclipseRT is often used for traditional software projects, it works well for mobile and embedded software as well. Embedded software faces a different set of challenges than traditional software: limited resources require high performance, low-profile frameworks and yet embedded devices are increasingly expected to be as dynamic and functional as regular applications.

Once again, the close attention to a high performance, component-oriented architecture lays the foundation for EclipseRT's success in embedded software, helping support everything from [physical security management](#) to [bomb sniffing](#). The embedded Rich Client Platform (eRCP), with a 10MB runtime, provides a small enough footprint for application delivery to many embedded scenarios.

Connections to back-end systems can be integrated with the Eclipse Communications Framework (EC), integration with back-end data stores and orchestration services can be provided with technology such as EclipseLink and Swordfish. The updating and management mechanisms in EclipseRT mean that embedded devices can be updated dynamically, adding new functionality as needed across the life-time of the device, even at runtime.

Developers can use the same tool-chain between embedded, desktop, server, and even mobile development with the familiar Eclipse working environment, making developers more productive by cutting down context-switching costs and helping unify development teams instead of splitting them based on the delivery model.

Conclusion - Eclipse Runtimes

Eclipse has excelled at providing cross-platform tooling for many years, and now with EclipseRT now provides a high quality foundation for all types of applications and services.

[Once main content is nailed down, the conclusion will tie together with a theme like "RT enables agile/rapid application delivery no matter what the delivery mode," or RT as a diverse, general purpose development and production environment." A key part will be emphasizing again that RT is only partly about development, but is also largely about production runtimes.]

