Close Window

Print Story

A Look at the Eclipse Callisto Release

Callisto is the simultaneous release of 10 major Eclipse projects at the same time. An important thing to note about Callisto is that even though it's the simultaneous release of 10 projects, it doesn't mean these projects are unified. Each one remains a separate Open Source project operating with its own project leadership, its own committers, and its own development plan. In the end, Callisto is about improving the productivity of developers working on top of Eclipse projects by providing a more transparent and predictable development cycle.

A Quick Tour of Callisto's Projects

In this article, we 'll go through each of the Callisto components. We'll give a brief overview of each and quote an Eclipse committer about what's exciting about his component in the Callisto release. Then we'll discuss some of the challenges that faced Callisto and conclude with the advantages gained by adopting Callisto. As you soak in what the committers have to say, remember that they are from the various corporations working together to make Callisto a reality.

Platform

The Eclipse Platform component (www.eclipse.org/platform) is the heart of Eclipse and has three main pieces:

- ***Java Development Tools (JDT)*** - www.eclipse.org/jdt - When most people think of Eclipse, this is the first component they think of. Eclipse provides a world-class Java development environment.
- ***User Interface/Core Tooling*** - This piece encompasses many smaller components in the Platform. It's responsible for all the visuals you see in Eclipse and features like team integration and Ant support.
- ***Plug-in Development Environment (PDE)*** - www.eclipse.org/pde - Have you ever used a wizard in Eclipse to create an Eclipse plug-in? If you have, you used the PDE. It's responsible for all the tooling in plug-in development.

Since it's hard to track down all the committers for each of the small Platform projects, we'll focus on what PDE has to offer Callisto:

"For the Callisto release, PDE provides comprehensive OSGi tooling, which would

make it an ideal development environment for component programming, not just Eclipse plug-in development. Other noteworthy features include quick fixes in plug-in manifest files, NLS tooling, and tighter integration with JDT via participation in search and refactoring."
- Wassim Melhem, PDE lead, IBM

C/C++ Development Tools (CDT)
www.eclipse.org/cdt
Did you know Eclipse isn't just for Java development? The CDT project aims to bring a fully functional C and C++ development environment to the Eclipse Platform. One should note that CDT can scale. A famous CDT demo is to import the Mozilla code base and use CDT to develop it.

"The CDT brings Callisto a development environment for writing C and C++ programs. The JDT sets a high bar as far as Eclipse IDEs go and we are constantly working in catch-up mode. For Callisto, the CDT provides an editor with all your regular text editor features such as language-specific keyword highlighting and content assist. It also provides an index of the user's code to provide search and code navigation features. There's also a framework for integrating build tools and debuggers to complete the edit-build-debug cycle. In this release, we've focused on a faster, more scalable indexing framework as well as a flexible build system that allows for per-resource builds as well as a new experimental internal builder that eliminates the need for MAKE files. We also have the beginnings of a framework for supporting additional compiled languages such as Fortran by the Photran project and hopefully more such as C# and Ada in the future.
- Doug Schaefer, CDT lead, QNX Software Systems

Business Intelligence & Reporting Tools (BIRT
www.eclipse.org/birt
The BIRT project strives to bring a Eclipse-based reporting system that integrates with your application to produce compelling reports for both Web and PDF. BIRT provides core reporting features such as a graphical report designer, data access, and scripting support. BIRT reminds me of Crystal Reports or JasperReports, but tightly integrated with Eclipse.

"With the Callisto release, BIRT expands on the themes of scaling, broader appeal, and simplicity. Some of the new features include Re-portlet support, which allows elements of a BIRT report to be rendered as partial HTML pages for better integration into dash boarding-type applications, joined datasets for combining disperse data sources into a single table, improved DTP integration, parameterized XML data sources, the ability to template an existing report design, and several chart enhancements. BIRT 2.1 will also provide better tooling to promote developed reports and ancillary files between environments.
- Jason Weatherby, BIRT evangelist, Actuate Corporation

Data Tools Platform (DTP)
www.eclipse.org/dtp
DTP project includes extensible frameworks and exemplary tools around data-centric technologies. DTP provides data management frameworks and tools not biased toward any vendor. If you plan to work with databases and use Eclipse, this should be your first stop for database tooling.

"The Eclipse Data Tools Platform (DTP) brings a number of key data-centric frameworks and tools to the Callisto feature set. Using these DTP frameworks and the examples provided for Apache Derby, the extender community can quickly achieve a high-functionality baseline working with heterogeneous data sources. Once this baseline is attained, specialized offerings for data-centric applications can then be created in the familiar Eclipse Plug-in Development Environment (PDE), allowing developers to leverage existing skills for the data domain."
- John Graham, DTP lead, Sybase Corporation

Eclipse Modeling Framework (EMF)
www.eclipse.org/emf
EMF is a modeling framework and code generation tool for building tools and other applications based on a structured model. To put it simply, EMF lets you build models quickly by taking advantage of EMF facilities. For example, one feature EMF provides is support for persisting models to XML (there are options to persist models to databases too).

"The Eclipse Modeling Framework provides powerful generative and runtime capabilities for applications based on structured data models. From a simple class diagram or XML Schema, you can generate a complete Java implementation of the model, along with an editor for it, and take advantage of EMF's facilities for persistence, notification, validation, and change recording in your application. Callisto includes EMF 2.2, which introduces many exciting new features: a simplified XMLProcessor API for XML persistence; cross-resource containment support; new code generation patterns, allowing, for instance, for all signs of EMF to be suppressed from generated interfaces, or for no interfaces to be generated at all; encryption support in resources; improved XML Schema generation and round-tripping; an extensible model exporter tool; an improved, extensible code generator; and various performance improvements and usability enhancements.
- David Steinberg, EMF committer, IBM

Graphical Editing Framework (GEF)
www.eclipse.org/gef
GEF serves as the base for graphical applications in Eclipse. It includes Draw2D (similar to Java2D), which is a lightweight graphical toolkit built on SWT. GEF itself is a framework that extends the Model-View-Controller paradigm to graphical editors. GEF brings your own model to the framework and provides facilities that take advantage of Draw2D to paint your figures.

"[For the Callisto release] GEF 3.2 is essentially a maintenance release in terms of features and bug fixes. Some minor features that were integrated were for supporting animated layout and general fixes to direct graph layout algorithm..."
- Steven Shaw, GEF/GMF committer, IBM

Graphical Modeling Framework (GMF)
www.eclipse.org/gmf
GMF is a new Eclipse project that aims to bridge EMF and GEF to allow for the generation of graphical editors.

"GMF brings Callisto a more efficient means for Eclipse developers to create graphical editors based on EMF and GEF. Based on model-driven development techniques, GMF leverages a series of models to generate editors targeting the feature-rich GMF diagramming runtime, which can also be used in the absence of the generative framework for the creation of high-quality editors. Follow the GMF Tutorial cheat sheet and online tutorial to get started."
- Richard Gronback, GMF lead, Borland

Test & Performance Tools Platform (TPTP)
www.eclipse.org/tptp
TPTP provides an open platform supplying powerful frameworks and services that allow software developers to build unique test and performance tools, both Open Source and commercial, that can be easily integrated with the platform and with other tools. The platform supports a broad spectrum of computing systems including embedded, standalone, enterprise, and high-performance and will continue to expand support to encompass the widest possible range of systems.

"TPTP provides a rich set of test, profiling, and monitoring tools. However its true value can only be realized by being part of a core typical user use case. By integrating with the WTP project and providing a 'profile on server' action TPTP becomes an easy link to collecting and analyzing your Web application performance characteristics. By further providing the ability to function and load test based on http requests TPTP helps the developer prove the quality of the Web application. Finally by providing customized extended reporting of the rich data TPTP collects with the use of BIRT the user can get the test and performance data they want and need to best manage their own project.
- Harm Sluiman, TPTP committer, IBM

WebTools Platform (WTP)
www.eclipse.org/webtools
The WTP Project extends the Eclipse Platform with tools for developing J2EE Web applications. The WTP project includes source editors for HTML, JavaScript, CSS, JSP, SQL, XML, DTD, XSD, and WSDL; graphical editors for XSD and WSDL; J2EE

project natures, builders, and models, and a J2EE navigator; a Web Service wizard and explorer, and WS-I Test Tools; and database access and query tools and models.

"WTP's 1.5 release in the Callisto train will include several new features and a number of stability and performance enhancements. Users of WTP Web Services will appreciate the upgrade to Axis 1.3 and streamlined Web Service and client wizards. XML Schema and WSDL graphical views have also been revamped to make them easier to navigate and read. WTP tackled some major infrastructure work in the Callisto release, moving to the platform's common navigator and undo stacks. The tabbed property support is also transitioning from WTP-only to the platform level in this release. Finally, the Dali and JSF projects are planning to do a technology preview around the Callisto timeframe and will provide some exciting 'first looks' at Java EE 5 tooling support that will preview support in WTP 2.0."
- Tim Wagner, WTP PMC lead, BEA

Visual Editor Project (VE)
www.eclipse.org/ve
Ever wondered if there was a way to create user interfaces visually, using the simple semantics of drag-and-drop? The Eclipse project provides VE, which is a open development platform for supplying frameworks to create GUI builders. VE has two exemplary implementations of Swing/JFC and SWT/RCP.

"The Visual Editor project (www.eclipse.org/vep/) adds the ability to visually develop SWT and Swing user interfaces in Callisto. The main focus of this release was to add initial support for creating Rich Client Platform (RCP) components with the VE. Towards that goal we've added: the ability to develop Views and Editors visually, support for the Forms UI toolkit, and the ability to work with JFace viewers. Other notable new features in this release include significantly enhanced tooling for SWT's GridLayout and support for VE on the Mac OS X platform."
- Jeff Myers, VE committer, IBM

Callisto's Challenges
There are two main challenges with Callisto. The first one and for many people the most obvious one is developing Callisto. Aligning 10 large projects for simultaneous release is very challenging. But once you actually get the release, you have to deliver it and that's a challenge on its own.

The method of choice for delivering Callisto is the Eclipse built-in update mechanism. So you only have to download the Eclipse Platform binary for your system and then you start Eclipse, use the Update Manager to visit the Callisto Update Site, and select the Callisto features you'd like to have installed in your environment. The Eclipse Update Manager will do the rest for you.

You can imagine that this will put a burden on a single update site (in terms of bandwidth use). In Eclipse 3.2, the Update Manager and the Eclipse.org infrastructure were enhanced to deliver Callisto. The goal for the Update Manager was to reduce the volume of data that's transferred and the goal for the Eclipse.org infrastructure was to create a reliable mirroring story for the Callisto Update Site.

Callisto's Advantages
Callisto brings several advantages to users and plug-in developers (adopters) of Callisto projects. Let's start with the user's perspective.

The User's Perspective
From the user's perspective Callisto radically changes the way Eclipse and the participating Eclipse projects get on the desktop. It takes away the need to read through the requirements sections and collect them manually from several download pages. You just download one Platform binary and select your desired projects from the Callisto Update Site after installing and starting the Platform binary.
Q: Which projects does WTP depend on?
A: Who cares. The Eclipse Update Manager will handle this.

Callisto also has another great advantage for Eclipse users. It creates some kind of accountability for all participating projects and their committers. Because Callisto creates a reference platform of Eclipse projects that are intended to work together. And if they don't now it's easier to report cross-project issues because you only need to reference the Callisto platform instead of collecting all dependencies.

The Developer's Perspective
From a developer and adopter's perspective, Callisto introduces stability (in terms of dependencies and investments). Before Callisto, it was up to you to select the projects you'd like to depend on. But often the result was disappointing because of some incompatible dependency conflicts. Now with Callisto the dependencies are clearly defined.

With clearly defined dependencies you get a target platform that will be valid and current for a long time. So Callisto also ensures that the investment you put in your adoptions are well spent in the long term.

Conclusion
On the whole, we hope you enjoyed this quick tour of Callisto and some of the challenges Callisto faced. We think Callisto will make it easier for end users to tailor their Eclipse experience by selecting what they want included in their Eclipse installation. Now, the only logical thing to do is give Callisto a try. See www.eclipse.org/callisto.