NEW LOOK!
Our Name has Changed!
FORMERLY LinuxWorld MAGAZINE

ENTERPRISE
OpenSource
MAGAZINE

Visit us at **opensource.sys-con.com**

THE LEADING MAGAZINE FOR ENTERPRISE AND IT MANAGEMENT   **JULY 2006** VOLUME 4 ISSUE 7

Linux Technology Leadership and the
Forking Issue

+ PLUS

AN
ARGUMENT
FOR
LINUX VARIANTS

What's Inside...

Net Neutrality

Is the freedom of the internet in jeopardy?

By Mark R. Hinkle

First, Internet access will be the core telecom service of the future. Not only will essentially everyone in the developed world be connected to the Internet, but more and more applications will migrate to the Internet. In our homes we will use a variety of devices which communicate via the Internet. We will use telephone-like devices to have voice conversations over the Internet. We will use one-way and two-way video devices that communicate through the Internet. We will use many different web-capable and email-capable devices. And certainly we will use important new Internet applications that we cannot yet imagine.

— **Charles Brewer**
MindSpring's Founder, Chairman and CEO
As Testified to at the Senate Committee on Commerce, Science, and Transportation
April 13, 1999

Why did I choose to quote my friend Charles Brewer? Perhaps I thought it relevant that during his campaign for "choice in access providers" seven years ago, he accurately summed up today's Internet (his complete testimony is linked at the end of this article and well worth the read). His efforts helped to ensure that you had choice in who serviced your broadband Internet service. Today I have the choice to buy my broadband service from Time Warner, Bell South, or EarthLink. Actually, I can even buy a wireless broadband access card from Verizon (I love my EV-DO card). Thanks Charles.

The fight for competitive broadband access was the predecessor of many fights that will take place over liberties related to the Internet. Charles correctly noted that we will be using the Internet for things like telephone conversations and television (for example, Skype and Slingbox). This has happened because of the competition, collaboration, and innovation that have thrived on the Internet. Even in Africa and undeveloped nations, the Linux-powered One Laptop per Child initiative (http://www.laptop.org) is going to offer children all over the world, despite their geography or economic status, access to the Internet much more quickly than imagined. It's our responsibility to be caretakers of the Internet and offer them the same freedoms that we enjoy when they join the global conversation. The threat we are faced with is the potential loss of Net Neutrality.

Net Neutrality (short for network neutrality) is the name given to the principle that all users of the Internet are treated the same. In a neutral network, data is delivered with the same priority, but that level playing field is seriously being threatened. Lawmakers are debating legislation that may affect your options on how you access content and how that content gets delivered. It's up to us to exercise our opinion and to lobby our legislators to protect our interests. If you don't

About the Author
Mark R. Hinkle is the vice president of strategy and corporate development for Emu Software as well as the editor-in-chief of **Enterprise OpenSource Magazine***. Mark served on the Formation Board of The Desktop Linux Consortium and is also the author of* Linux Business Desktop Migration for Windows Users *from Charles River Press.*
mrhinkle@encoreopus.com

Reclaim lost time

The world's first Linux management appliance

The Intrepid M™ appliance performs the most important Linux management tasks in a fraction of the time you now spend. You'll save both time and money with a single point of control.

The Intrepid M does these tasks and more:

- **Quickly provision servers** — in minutes—even from bare metal

- **Easily deploy software and patches** – eliminating lengthy installations

- **Efficiently migrate software** — from one piece of hardware to another

- **Instantly rollback** — undo changes in minutes, even to the distant past

"Levanta reduced an eight-hour process down to five minutes."

-Arty Ecock, Manager of VM Enterprise Systems, City University of New York (CUNY), Computing and Information Systems

Levanta Intrepid M™

Download the Illuminata Analyst Report

Levanta Marries Provisioning with Data Virtualization

www.levanta.com/lwm0406

Find out why the Intrepid M is the cutting edge in Linux server management.

LEVANTA®
www.levanta.com
1.877.LEVANTA

WINNER
LINUXWORLD
Most Innovative
Hardware Solution

from the editor

take the initiative to tell them how you feel, this issue could be decided based on the efforts of high-paid lobbyists backed with dollars from parties whose interest is in their own profits, not in maintaining a free and competitive global network. Then again, maybe there's nothing to worry about, but it is definitely worth your time to be somewhat well versed on this very complex issue.

The reason I call the issue of network neutrality to your attention is because it has benefited people like Linus Torvalds, who used the neutral Internet to share his Linux kernel and collaborate with other users to make an operating system (Linux) that rivals that of the one made by multi-billion dollar Sun Microsystems. It's the thing that has made global collaboration among open source developers possible. It's what allows people all over the world to communicate using e-mail, instant messenger, and Voice over IP. Whether you are a single person with a blog or a large multi-national company with deep pockets, you get the same treatment. Imagine if network providers such as AT&T, Comcast, and Verizon were legally allowed to decide at what rate and priority content was delivered. What would that mean to us as end users? Would it open the door for infringement of the freedoms of the Internet that many of us take for granted? Would it stifle competition and freedom of expression? If we don't take action to prevent this, I believe there is potential for the Internet to turn into a pay-to-play market, dominated by those who have the most money to spend to fast-track their products, services, and ideas to you.

Net Neutrality is of global importance, not just a U.S. concern. However, it's coming to light in the U.S. because of our position as one of the largest consumers of Internet services. Congress is currently considering a rewrite of the Telecommunication Act of 1996, which was intended to deregulate and foster competition by giving consumers choice in local phone service. It also had the interesting consequence of allowing many mergers to occur in the telecom industry, including some of the formerly broken apart Bell companies to consolidate, but more on that later.

If you notice I haven't used the word "rights," because freedom to use the Internet without discrimination was not spelled out in the Bill of Rights or any document for that matter. It was probably something that our founding fathers could have never imagined when they penned those words, specifically: "Congress shall make no law respecting an establishment of religion, or prohibiting the free exercise thereof; or abridging the freedom of speech, or of the press; or the right of the people peaceably to assemble, and to petition the government for a redress of grievances." Right now the Internet provides us all a place to voice our thoughts and opinions with equality; what happens when the ability to share opinions goes to the highest bidder. This is the point where you can help set the direction of the freedom of the Internet by contacting your legislators and voicing your opinions. There are already advocacy groups that will help you understand and articulate these points to your legislator; they include the Save the Internet (www.savetheinternet.com) and It's Our Net (www.itsournet.org).

Let me step back, because perhaps I am not being fair; perchance I should advocate a laissez-faire approach to this issue, urging you to tell your legislator not do anything and to leave the telecom providers to their own devices. I would, in some situations, say that sounds like a good idea. Though I'll use an historic example of what happens when telecom providers are left unchecked – specifically in 1984 when the Bell System was broken apart into the regional Bell operating companies known as the RBOCs. This was the result of an anti-trust lawsuit brought by the U.S. Department of Justice to break up the telephone monopoly. I believe that left unchecked and without minimal regulatory oversight, the new monopolies and practices that stifle competition will arise. I offer the Internet Freedom Act of 2006 (S. 2917) submitted by Senators Byron Dorgan (D –North Dakota) and Olympia Snowe (R – Maine) as a way to make sure that providers of Internet access don't mess with the way our bits and bytes are delivered.

Network neutrality is a high stakes proposition. On the one side there is the potential for stifling competition and innovation, the tenants of success for the Internet. The counterpart is that without a way to increase their revenues, bandwidth providers won't be able to subsidize the build-out of an increasingly wired world. I also don't oppose different pricing and other changes in today's model; the value the Internet provides me is immeasurable. What is critical is that the Internet remains free from bias and that any governance is there only to provide neutrality. ●

Additional Resources
- *Save the Internet: http://www.savetheinternet.com*
- *It's Our Net: http://www.itsournet.org*
- *Testimony of Charles M. Brewer to the Senate Committee, April 13, 1999:*
  http://commerce.senate.gov/hearings/0413bre.pdf
- *S. 2917 The Internet Freedom Preservation Act:*
  http://thomas.loc.gov/cgi-bin/query/D?c109:8:./temp/~c109BAFr0t::

Opening Up the Enterprise

The emergence of true enterprise-class applications

by Tom Manos

A quick Google search of "enterprise Open Source" returns over 1.5 million results. There are Web sites devoted to the topic; there are dedicated conferences and trade shows, and even a fairly large number of books. There's even a magazine — the one now in your hands! — devoted entirely to the subject of enterprise Open Source. Most of these sources discuss using Open Source applications and infrastructure in one's business, and tacitly assume that this defines enterprise Open Source. This implied definition falls short of what I think of when envisioning an "Open Source enterprise application" however.

About the Author

Tom Manos is the CTO and co-founder of Centric CRM. A nationally respected Internet expert, Tom has founded and held senior technology posts at several other notable ventures including North Wall LLC and NetworkTwo Communications Group. In 1993, he co-founded InfiNet, one of the first national ISPs and then the largest "private label" ISP in the world. Infi-Net was subsequently sold to Landmark Communications. Before becoming an entrepreneur, Tom had a distinguished career as a U.S. naval officer, including a successful stint as director of computer operations for the World Wide Military Command and Control System (WWMCCS) for the Navy's Atlantic Command. Tom is a long-time member of the executive advisory board of the International Engineering Consortium, holds seats on several other boards and frequently speaks at telecom industry conferences. He has a BS in electrical engineering and his MS in computer science. He lives with his wife Gail and their two children in Chesapeake, VA.

tmanos@centriccrm.com

This article investigates a more rigorous rationale for the term and describes what's necessary for a particular piece of software to call itself "enterprise Open Source." It also describes a growing movement in Open Source: the enterprise Open Source application.

Defining Enterprise Software

Over the years, the term enterprise software has taken on many meanings, ranging from heavy-duty operating systems, such as Sun's Solaris Enterprise System and Red Hat Enterprise Linux to more classical definitions like Martin Fowler's:

Enterprise application is the name I give to a certain class of software systems: the data-intensive software systems on which so many businesses run. Another, and perhaps better, name for them is Information Systems since these are systems that process and manipulate information.

For purposes of this article, I'll use a definition similar to Martin's: Enterprise Software is software that solves problems across an enterprise, rather than specific problems within an enterprise. Thus, most desktop applications wouldn't be considered enterprise applications and, in this case, I won't include operating systems or database systems since in and of themselves they don't solve problems or process information.

Examples of true enterprise applications include Customer Relationship Management (CRM), Enterprise Resource Planning (ERP), Content Management, and corporate financial applications. Does this mean that every ERP system is an example of enterprise software, somehow deserving of the title? No, not necessarily. I submit that there are some additional requirements that must be met before calling an application an enterprise application. In no particular order, some additional requirements might include:

- **High availability**. If operations or revenue stop when the application stops, then the application must be continuously available.
- **Security**. The application will presumably be processing the enterprise's most precious possession: its information. As has been so clearly reported recently in the media, when dealing with competitive information, financial information, or customer information, strong security is a must.
- **High performance**. The application must perform adequately under the load it's given to support its users.
- **Scalability**. The application must scale to whatever size is required of it. While there are those who would claim that even an application built solely for very small enterprises might be called enterprise applications, I lean more to a description of software suitable for the larger corporation. So I wouldn't normally consider QuickBooks to be an enterprise application even though it's arguably the world's leading financial application.
- **Support**. The application must be fully and formally supported. While I suppose a wonderful piece of software that fulfilled all other requirements might be called an enterprise application, no real company would use it. In the enterprise application world, support from the Open Source community alone is not enough.

While I'm sure there're many more requirements that could be added, in my view these are the most important and are certainly enough to illustrate the case.

Building Enterprise Open Source Applications

So what does it take to design, build, market, and sell enterprise Open Source applications? After many years in and around both the commercial and Open Source worlds, I've learned many lessons about what pleases and motivates customers, partners, and communities. Great and relevant software is obviously at the top of the list, but control is most certainly on the list. Customers want and need the ultimate in control and flexibility, and the greatest protection against vendors whose interests are not, in the end, perfectly aligned with their own. After all, many software companies see their business first as selling software and only secondarily as pleasing their customers and partners. Open Source software helps shield the customer from these worries. Because the actual application code is in their hands, they are no longer at the mercy of an uncooperative or even adversarial vendor.

It's no surprise then that ever-larger enterprises are beginning to evaluate Open Source solutions seriously to determine whether such products provide a legitimate alternative to traditional proprietary applications. "We are making a strategic move to embrace open source technologies," says Brian Shield, CIO of The Weather Channel. "Finding enterprise-ready products like Centric CRM have allowed us to mothball several expensive proprietary systems already."

The challenge for Open Source providers, therefore, is to build an application that simultaneously embodies the spirit of Open Source, while also fulfilling all the requirements necessary to be seriously considered ready for the enterprise, including the very large enterprise. This means solving a number of potentially conflicting requirements. On the one hand, it must appeal to the Open Source community, be accessible, understandable, inexpensive or even free, and run entirely on an Open Source infrastructure. On the other hand, it must be built on an enterprise architecture, support very large installations, and be capable of running on any application stack the customer specifies with little or no change.

An Example of an Enterprise Open Source Application

Customer Relationship Management (CRM) is one of the most critical activities successful companies engage in. CRM is first and foremost a business strategy. Successive waves of technologies — starting with index cards! — have emerged to help businesses do a better job of managing their customer relationships.

Most CRM solutions available to date have involved expensive proprietary software. Some six years ago, my company, Centric CRM, set out to change all that. I'd like to use our long (at least by Open Source standards) experience to illustrate some important points about enterprise Open Source development.

We set out to design Centric CRM first and foremost using only Open Source or freely available components. We chose the following:

- **Platform** – We wanted our application platform to be enterprise-class and totally cross-platform and portable. The only available platform that solved all our problems was J2EE. While the Java language itself is not Open Source (although I think it will be soon), it's freely available and has a variety of free development tools and libraries. There are also compelling Open Source Java application servers such as Apache Tomcat and JBoss. We passed on scripting languages such as PHP and Python because, while they make great prototyping and rapid development platforms, they lack the enterprise security and scalability features we believed our enterprise customers require. They're also not well suited to building very large and complex systems. We didn't think the CIO of a typical Fortune 500 company would be receptive to a critical business application built on such a platform. And although Microsoft's .NET platform would have been a good candidate in terms of performance, scalability, and power, it certainly fell short in the Open Source category!
- **Database** – For our reference database, we wanted a system that we could distribute with Centric CRM that was very standards-compliant and provided very high performance and scalability. The only database in this class at the time was PostgreSQL. Another choice might have been MySQL, but seven years ago it didn't fully support the relevant SQL standards, and wasn't freely distributable.
- **Operating System** – There was really only one choice here: Linux. For our reference system, we chose Red Hat Linux.

Java as the Platform for Enterprise Open Source Development

The J2EE stack is at once reasonably well understood and accepted by the Open Source community (look at the recent breathtaking acquisition of JBoss by Red Hat), and one that with thoughtful and correct design and architecture will allow an application to fit nicely into the large enterprise.

Customers want and need the ultimate in control and flexibility, and the greatest protection against vendors whose interests are not, in the end, perfectly aligned with their own

Choosing Java as the development platform lets enterprise Open Source application developers be completely operating system-independent. If the application is designed with portability in mind, J2EE will let it run identically on Linux, BSD, Solaris, Windows, and just about any other operating system you or a CIO would be likely to specify. If the database interface uses only JDBC, standard SQL 92/99 queries, and no special or non-portable database capabilities, such as triggers and stored procedures, you have the beginnings of database portability. And if all the application's business logic is written in Java, the application will run identically and quite happily on PostgreSQL, Microsoft SQL Server, Firebird, Daffodil DB, DB2, and a host of other standards-compliant databases. (There's certainly a tradeoff in performance and convenience using this database architecture, but as long as adequate performance across a range of installation sizes is possible, I believe that the customer should be the final arbiter in the choice of database.)

Choosing Apache Tomcat as a reference application server lets one easily move to JBoss, WebLogic, or WebSphere when customers need the power and scalability provided by a full-blown J2EE application server environment. Taking advantage of the advanced features of enterprise middleware then becomes a matter of writing code that is "vanilla" enough to run on all of the available choices. Sometimes this involves a tricky design, but I've always found it possible. Furthermore, the development discipline such an approach enforces leads to the creation of very well structured code. (Download our source code as an example.)

The Emergence of an Enterprise Open Source Ecosystem

Centric CRM is by no means the only Open Source application developer that has thought through the issues involved in building true enterprise-class Open Source solutions. Indeed, one of the most exciting trends in the IT industry today is the emergence of a recognizable handful of serious enterprise Open Source products. Without the guiding hand of a single organization, there is now available a suite of complementary enterprise-class products that solve a wide range of corporate IT problems.

For example, in the content and document management space, you have Alfresco and Magnolia. For reporting and business intelligence you have Jaspersoft. For system and application monitoring, you have Hyperic HQ. For project management, there's Project.net. "It may take some getting used to, but Open Source is the way of the future for enterprise software," says Peter Winston, CEO of Project.net. "With the product source code, customers have the ability to customize applications to their own specifications in much less time and at a much lower cost of financial and human resources."

The emergence of this suite of enterprise-class applications is also creating tremendous opportunities for VARs and systems integrators with Open Source expertise. "We are now able to offer our corporate customers cross-departmental Open Source business functionality," says Ron Bongo, CEO of CorraTech, a large East Coast Open Source integrator. "By delivering solutions using Open Source code within a Service Oriented Architecture (SOA), we can provide deep integration that's maintainable."

An Enterprise Open Source License

Despite the lack of a guiding hand, all of these vendors have made substantially similar choices in the architecture they've chosen. In particular, they have all chosen J2EE as a development platform. They've also made substantially the same choices about how they license their products. Consensus is building around the key elements an *enterprise Open Source* license must embrace.

In particular, the following elements are important to an effective enterprise license. First, the source code must be truly open. Virtually all of the vendors mentioned above have turned away from the dual license in which a stripped-down community version is offered under an open license, while a separate proprietary license governs the use of the complete version of the product. Instead, most of these vendors — Centric CRM included — make their full product source code available for download under a single open license.

Second, the license must discourage "forking" the code. As long as large enterprises have to worry about multiple variants of a product emerging, they won't embrace Open Source solutions for mission-critical applications. Instead they need a unitary code base backed by a real vendor offering professional service and support. An enterprise Open Source license creates this condition by restricting redistribution of the source code. So while the full source code is truly open (i.e., freely available to download and inspect), it can't be resold (presumably with modifications) without the vendor's permission. Finally, any modifications that a customer makes to the application for its own internal use must belong to them. Customers who undertake meaningful development to create significant competitive advantage for themselves can't be forced to submit their enhancements back to the core product involuntarily. That said we've found that in practice all of our customers want their enhancements merged into the core code so that they're relieved of the burden of maintaining those enhancements in the future.

True Enterprise-Class Open Source

After seven years of dedicated design and development, we now have *exactly* the same code, governed by a single license, running in a five-person shop under Linux, Tomcat, and PostgreSQL *and* in a Fortune 500 company running HP-UX, DB2, and WebSphere in an installation that will soon have thousands of simultaneous users in a fully clustered and replicated infrastructure. (There are, of course, many other customers between these two extremes.) Our goal all along has been that the customer should have ultimate control of his application and the infrastructure it runs on. I am proud to say that we've achieved that goal.

In summary, by choosing the correct operating environment, using sound and sensible design, and keeping one's eye on the goal, it's possible to craft an enterprise Open Source application that's accessible to customers and the community, is portable across operating systems, databases, and application servers, and runs on a 100% Open Source stack. I've provided a high-level view of how a single *identical* code base, governed by a single Open Source license, can be as appealing and accessible to a shop running it on an old desktop machine with 256MB of RAM as to a large data center using all the power of an enterprise-class infrastructure and run by a battle-hardened CIO.

True enterprise Open Source applications have arrived. ⊙

The Power of Being There.®
Times two.

Discover how the combined power of Avocent and Cyclades IT infrastructure management solutions can take you and your data center to the next level. KVM, serial and power – all over IP. Plus, Intelligent Platform Management Interface (IPMI) and embedded KVM. The Power of Being There®. **Times two.**

Avocent®

cyclades
An Avocent® Company

Visit www.avocent.com/powerx2

Linux Technology Leadership and the Forking Issue

An argument for Linux variants

by Kevin D. Morgan

About the Author

Kevin Morgan has 20 years of experience developing embedded and real-time computer systems for Hewlett-Packard. Experienced in operating systems and their development, Kevin was a member of the HP 1000 computer software design team. While at HP, he worked as an engineer, project manager, and section manager spanning the development of five operating systems. As HP-UX operating system laboratory manager, he was responsible for overall HP-UX release planning, execution, and delivery for HP servers. Kevin has been leading the MontaVista Software engineering team as VP of engineering since joining the company in November 1999. He got his BS in computer science from the University of California, Santa Barbara and his MS in computer science at the University of California, Berkeley.

Linux is the fastest-growing embedded operating environment in the world today. It's quickly becoming the single largest operating system platform for embedded computing. As a result, many technology managers must come to grips with the complexity and the dynamics of Open Source software in general and Linux evolution in particular. Particular questions and concerns arise in the areas of compatibility, the role and nature of different versions of Linux (the "forking issue"), and the technology advancement process itself. The history of incompatible proprietary versions of Unix contributes significantly to these concerns. It's critical for the continued successful adoption of Linux that these concerns be assessed and understood. The nature of Linux as Open Source dramatically changes not just the specifics of these concerns, but relative to the history of Unix, changes even the nature of the concerns themselves.

The Nature of Linux Innovation

Linux is Open Source. What does this mean (in terms that are significant to compatibility and forking)? There are several critical facts:

- Linux is formally a trademark, owned by Linus Torvalds. The trademark is associated with a body of source code hosted at the http://www.kernel.org/ Web site.
- Being Open Source, anyone is free to procure a copy of Linux source code from kernel.org, and modify it.
- Those who do so are encouraged to submit their modifications back to the maintainers of the Linux source code. These submissions may be accepted quickly, may enter a short or long period of discussion and possible evolution before being accepted, or may be rejected.
- Through this process, Linux advances as a technology.
- Independent bodies of source code derived from the kernel.org source code base are routinely called Linux, even when modified. Mr. Torvalds has not campaigned to restrict the use of the Linux trademark to the formal code that resides at kernel.org.

Without this "external to the kernel.org" source code modification process, Linux could not and would not advance. As with any software development program, the "master copy" is only modified when the changes, made separately, are to a greater or lesser degree "complete" and "validated." Unlike proprietary software where an exclusive few are allowed to participate in the innovation cycle, with Linux all software engineers are encouraged to get involved, innovate, and contribute back. This independent innovation activity is happening continuously, everyday, by people working in an individual ("amateur") capacity, and by technology companies motivated by business needs and opportunities.

Market Requirements

As with all vibrant system software technologies, Linux continues to rapidly advance in capabilities to meet new and increased requirements. New requirements arise from multiple sources. One key source is hardware advances. As new features are developed and made available in commercial silicon, Linux must advance to enable and take advantage of that hardware feature. A few recent examples of this include 64 bit processors, on-board security features, multi-core processors with various operating modes, and, of course, extensive new and enhanced I/O capabilities.

A second key source of new requirements for Linux is the Linux's encroaching into new markets or application domains. The general adoption of Linux into embedded computing environments since the late 1990s has created a tremendous number of technical challenges for Linux. These include advanced high-availability requirements for the telecommunications market, improvements in flash memory utilization and real-time performance for multiple markets, boot time and power management improvements for the consumer products and mobile handset markets, and many more.

Linux Latency and Linux Stability

The source code base at kernel.org is managed in a way that strives to balance stability and rapid innovation. The particular mechanics used have evolved over recent years, and no doubt will continue to evolve. What has been constant is that, overall, innovation is the primary focus rather than stability. As a result, those familiar with the daily and weekly level of evolution of the kernel.org software are aware that the leading-edge source base can't always be expected to be in a highly reliable and robust state relative to typical commercial requirements for stability.

There is also a natural latency in any code modification submitted to the kernel.org process. Small "obvious" corrections may only take a few days to move through the process and become integrated into the Open Source code base. Larger corrections and significant incremental value components can take weeks, months, or even years of assessment, debate, refinement, usage feedback and, in general, understanding, before being formally accepted and integrated into the kernel.org source base.

There are several critical pragmatic outcomes of these facts. First, basing a commercial embedded system product development program on the kernel.org source base as "the" source base for the project (i.e., literally utilizing no local copy) is somewhere between extremely risky and outright impossible. A single, small, required change in code would generate the need to "re-source" from kernel.org, and in doing that potentially large amounts of changes would be picked up. Doing this late in a development program could have dramatically deleterious effects on the product program. For this reason, everyone doing serious product development uses a private copy of the Linux source base. A stable, minimally changing set of product source code that's under the absolute change control

of the product team is a fundamental requirement for commercial software development, even when major components of that product are based on Open Source code.

Similarly, anytime an advanced feature is required that isn't already extant in the kernel.org software, it's necessary to procure or develop a derivative version of Linux oneself that includes the required feature. Often such features exist as Open Source component software (typically structured as patches for the current kernel.org source code). Such features are frequently included in commercial products, such as MontaVista Linux. Sometimes, the features require new development. It can be argued that submitting such features back to the kernel.org source base is in the best interests of the developing organization (but that's the subject of a different article). What's critical to understand here is that any such submission and re-sourcing of the kernel.org software with the advanced feature included is rarely a viable development path for a commercial product development program. The delay and instability risks dictate that development, stabilization, and deployment using a privately derived set of source code is the only viable approach.

What Is a Fork?

We've now covered enough of the basics of Open Source Linux and commercial product development requirements to discuss the risks of a fork of the Linux kernel.

First, what is the definition of a "fork" in software?

In its most distilled form, a fork is any version of source code that is different (modified) from the master copy. By this definition, every time anyone, anywhere, makes a copy of Linux and begins to make modifications, they're creating a fork. In this sense, Linux is forking literally thousands of times a day.

Pragmatically however, this is not the common interpretation of the word "fork" as it applies to software in general and to operating system software in particular. A fork is generally construed as:

- A long-lived derivative of the master version of source code, and
- A derivative that's not intended to "resynchronize" with the master source code, and
- A derivative of some significance in the marketplace.

Hence, short-lived derivatives that precede the submission of the changes back to kernel.org aren't forks. Commercial companies embedding Linux inside a product that
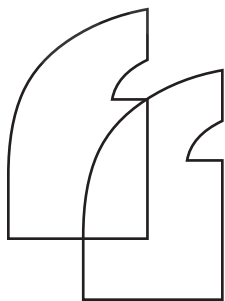
> All significant Linux distributions are Linux variants, and none of them are or have become Linux forks and any viable Linux distributor, **by the very nature of the product and services it is offering, MUST** offer a Linux variant

have developed a derived version with code corrections and/or enhancements that the company doesn't submit back to kernel.org is not a fork. Such a derivative version doesn't have significance to the broader operating systems marketplace; it's a private version of no broader significance.

Similarly, a derived version of Linux that's frequently resynchronized with the evolving kernel.org source code base is not a fork. Almost every commercial distribution of Linux falls into this category. Some new entrants into the Linux distribution market are striving to label such distributions "a fork of Linux." Such claims are just inaccurate, they're disingenuous for the reasons cited above: it's only through such approaches that serious commercial distributions targeting specific markets can be developed and delivered...and those making such assertions know this. The claims are useful for purposes of creating fear, confusion, and doubt, and, in general, slowing down market adoption while such companies strive to catch up with their own distribution products.

Commercial Linux Technology Leadership

Let's explore the details of commercial Linux distribution in more detail. We particularly want to consider the situation of a company using Linux as a commercial operating system platform embedded in products in a market with significant requirements that extend beyond what the Linux from kernel.org is capable of.

MontaVista's approach to this situation is relatively simple. Here are its key strategies:
- Identify the key market (or application) requirements that are underserved by standard kernel.org Linux.
- Identify any extant Open Source technologies already developed or in development but not yet integrated into kernel.org Linux, and if available, assess their maturity, their state of ongoing development and support, their state of adoption by other industry players, and their potential for adoption into kernel.org Linux.

- Either adopt an extant technology or begin developing a new technology. When developing new technology, use Open Source development models to (1) maximize community leverage, and (2) maximize the potential for both broader adoption and use and, ultimately, integration into kernel.org source code.
- Aggressively commercialize the feature in MontaVista Linux products. Provide this capability as a high-value, fully supported and maintained component in the overall set of value components of the complete MontaVista Linux product.

Obviously MontaVista maintains private source code copies of Linux. These are either a new product version in development, an actively sold and supported version undergoing ongoing maintenance and minor enhancement (primarily new hardware support), or an obsolete version that may still be under special support to customers requiring "long lifecycle" support.

MontaVista aggressively integrates advanced capabilities not yet integrated into kernel.org into the MontaVista Linux product. In many cases these are capabilities developed primarily by MontaVista, using Open Source development methods. Sometimes these are capabilities developed by others, with or without active MontaVista participation or contribution. In all cases, the following are true:
- The technologies are developed and maintained as independent Open Source components.
- The technologies are promoted for inclusion back into kernel.org source.
- MontaVista continues to move its own product line forward with new versions of MontaVista Linux, each of which moves forward to a more recent baseline of source code from kernel.org (and many other critical components, such as gcc, gdb, glibc, etc.).

To clearly distinguish how such Linux variants form a true fork of the Linux kernel,

let's describe a hypothetical "real fork" scenario.

Linux is developed with a first-order focus on server and desktop computing. Requirements from more specialized areas of computing aren't given the highest levels of priority. Such areas include consumer products, telecommunications equipment, mobile handsets, and mil-aero. When implementation or optimization choices conflict with requirements from different application segments, servers and desktops typically (and arguably, appropriately) win out.

Given this backdrop, it's possible that an individual or a group (for example, an industry consortium) could build, market, and maintain a separate, specialized version of Linux. Let's say, for example, several large mil-aero firms decided to cooperate in developing and maintaining a "mil-aero Linux." To do this they'd choose a starting point set of Linux code from kernel.org and publish this code independent of kernel.org (say, www.mi-aero-linux.org). They'd then begin the process of evolving this code base to meet better the requirements of the mil-aero equipment market. The stated intention would be to make long-term design, content, and management decisions to optimize the code for those requirements. In doing so, the decision would be to diverge henceforth from the kernel.org source base due to the very different design objectives and specific code evolution being planned.

If such a development occurred, and the project moved forward successfully and the resulting software gathered significant adoption and use...THAT would be a fork of the Linux kernel.

It would have all the attributes of a true fork: a long-term variation; no intention to resynchronize with Linux proper; no attempt to get the capabilities in the fork back into Linux proper; and fielding some significance in the overall operating system market.

Legitimate Concerns

To date, no true Linux fork has arisen. So with respect to the Linux variants that aren't

DEFINE **YOUR** OPEN ENTERPRISE™

In my Open Enterprise, productivity is up because workgroup solutions actually work the way groups want them to.

People are more productive when they have the tools and support they need to work more efficiently. Workgroup solutions from Novell® unite infrastructure, services and tools with unmatched security and reliability. Regardless of location or device. Our full suite of networking, communication and collaboration services support more users on a single server, simplifying administration and significantly reducing costs. So you can communicate and collaborate the way you want, wherever you want—for a lot less than you'd think.
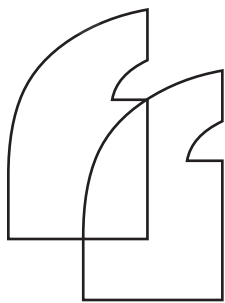
Workgroup solutions from Novell.
This is the way to connect your Open Enterprise.

Novell®

This is Your Open Enterprise.™
www.novell.com/connect

> ## Some new entrants into the Linux distribution market label competitive distributions 'a fork.' **Such claims are inaccurate and disingenuous**

forks of the Linux kernel, are there legitimate concerns about them having "differences" from the kernel.org source code?

The answer is both yes and no. The first area of concern is "compatibility." By definition, if there are new functions (via new or different APIs) in a Linux variant then use of those functions creates "incompatible" software relative to the current kernel.org version of Linux. Before concluding that this is a concern, however, we have to answer a couple of fundamental questions: Why did a commercial vendor add this feature or API to the system? And why is a commercial customer of that product considering using it?

The answers should be obvious by now: the features and API were added so Linux could be used in an application where it otherwise it couldn't be used. And the only legitimate reason for a customer of such a product to use that extended API would be because its requirements can't be serviced without it. So while it's true that the resulting application software won't be compatible with the Linux available from kernel.org, consider the alternative: a product development team (and possibly an entire application or market segment) denied the benefits of Linux because it won't meet their requirements.

A second concern is vendor lock-in: Does the inclusion of extensions and their use by customers re-create a proprietary lock-in situation similar to the "bad old days" of proprietary operating systems? The answer is definitely no.

The reason is that the source code for all the API extensions is included in the distribution; the customer that requires the APIs and uses them has the full implementation of those APIs and can look to other alternatives for maintenance and support of that code in the future if the current vendor fails to satisfy.

A second reason is that (for MontaVista's kernel technology and in general for all exten-sions of market significance) the extensions are independently available as Open Source technology. Typically, this means that the enabling source code is available as patches to the standard kernel.org kernel. So not only is there no lock-in, the switching costs can be very low from the perspective of getting the required APIs integrated into new or different versions of Linux.

Many of the extensions for a particular market segment are becoming de facto standards and are available from multiple commercial vendors. Examples include XFS for streaming media applications, OpenAIS and OpenIPMI for telecommunications equipment, and dynamic power management for mobile handsets.

Summary

The concerns about commercial vendors selling products that "fork the Linux kernel" are not just overblown; analysis shows them to be a red herring. All significant Linux distributions are Linux variants, and none of them are or have become Linux forks. We have shown how even basic commercial support activities require an independently maintained copy (a "variant") of the Linux kernel. Those value-add components (features, internal "-ability" enhancements, and quality improvements) are frequently the critical factor in the purchasing decision and without those values a Linux-based product may not be feasible. So if and when a vendor comes calling, describing competitors' products as "a fork of Linux," ask these hard questions:
- Is YOUR Linux distribution a 100% copy of kernel.org without any changes in source code whatsoever?
- If so, why should I buy it from you?
- If so, how will you maintain it and support me? How can you provide me with integrated defect corrections, without changing the code?
- If code changes to fix defects must first be submitted, accepted, and published by kernel.org, what am I to do in the meantime? That process can be lengthy.
- If your Linux distribution is not a 100% copy of kernel.org, why is it NOT a fork of the Linux kernel and yet the competitor's product IS a fork?

These questions will quickly reveal the truth: any viable Linux distributor, by the very nature of the product and services it is offering, MUST offer a Linux variant. The issue isn't who is or isn't "forking Linux." In fact, it can be argued that any vendor who suggests that a competitor's product is a fork of the Linux kernel has such a low understanding of Linux and system software in general that he should be thrown out immediately in any serious vendor evaluation. These kinds of statements indicate some combination of deep ignorance or deep misinformation under the assumption that the customer is deeply ignorant. Either is a legitimate reason to keep searching for vendors of quality.

Linux leadership requires aggressive innovation and value provision. It requires investment, but not traditional IP investment – it requires Open Source investment. Key elements of any evaluation criteria are the following questions: who is driving and leading advanced Linux Open Source technology for the markets and applications of interest? Who is actively enabling Linux on the hardware critical to that market? Who is innovating the key features needed so Linux can be used in those applications?

These questions are easy to answer through assessment of Open Source projects and their leadership, through source code counts of different vendors' copyrights and check-in log data in the kernel.org source base, through participation and the leadership history at any number of key industry forums, such as OSDL, SAF, and CELF. It's at this level of assessment that leadership, value provision, and true product differentiation become clear. ⊙

Getting the Most Out of Eclipse

Scaling the DOM

by Doug Schaefer

Eclipse has emerged as the platform that finally integrates development tools from multiple vendors into a common, extensible, and widely supported framework. The Eclipse CDT development environment brings this rich offering of integrated tools to the world of C and C++ programming, enabling C/C++ developers to improve their productivity and enhance the quality of the software they produce.

The CDT project began in the summer of 2002 with source code donated by QNX Software Systems. Since then, developers from a number of organizations have contributed to the CDT, helping it evolve into a mature commercial-quality development environment. This diverse participation, which lets the CDT address the needs of a wide audience of developers, brings together the best minds in the C/C++ IDE industry.

In the summer of 2005, the CDT project team released CDT 3.1, which offers a feature set that closely matches the capabilities of other commercial and Open Source development environments. When designing version 3.1, the CDT team paid special attention to scalability for large development projects and laid the foundations that will allow the CDT to grow beyond the edit/compile/debug cycle of development.

This article walks through the CDT, with a special focus on features introduced in CDT 3.1. We'll also preview upcoming features that will make the CDT even more responsive and scalable than it is today. We'll then discuss the growing adoption of the CDT and how you can contribute to it.

CDT Overview

The CDT has three major components:
- **CDT Core** — helps developers create, edit, and navigate projects
- **CDT Build** — integrates with build tools to create executables for projects
- **CDT Debug** — integrates with native debuggers to help developers debug those executables

CDT Core

The CDT Core comprises several components, including project wizards, a source code editor, and a source code navigation system. Project wizards

The project wizards help simplify the task of creating CDT projects. There is a wizard for each language (C and C++) and for each supported build system. Using the wizards, developers can quickly set the properties and build settings for a project, as well as enable or disable optional CDT features.

Source Code Editor

The CDT source code editor provides a modern feature set for browsing and editing C/C++ source. For instance, the editor:
- Colorizes keywords and literals in the source code
- Displays the location of matching brackets and braces
- Offers code-formatting features for shifting code left and right and for commenting and uncommenting out lines of code
- Provides code completion — the user can type the prefix of an identifier and then select from a list of possible completions for that identifier

With its built-in parser, the CDT provides a very accurate list of code completions based on the context of the code that the user is writing. This feature saves the time required to type in the completions and provides API assistance that frees the developer from having to rely on printed manuals.

Source Navigation

The CDT provides several features to help the developer navigate through the code in a project. These features are particularly useful when the developer is new to a project and doesn't necessarily know where certain pieces of code are defined or referenced.

The Document Object Model, or DOM, is at the core of the CDT's source-navigation features. The DOM provides a database of syntax

About the Author
Doug Schaefer is project lead of the Eclipse CDT project and senior software engineer at QNX Software Systems, which donated the source code that kicked off the CDT project.
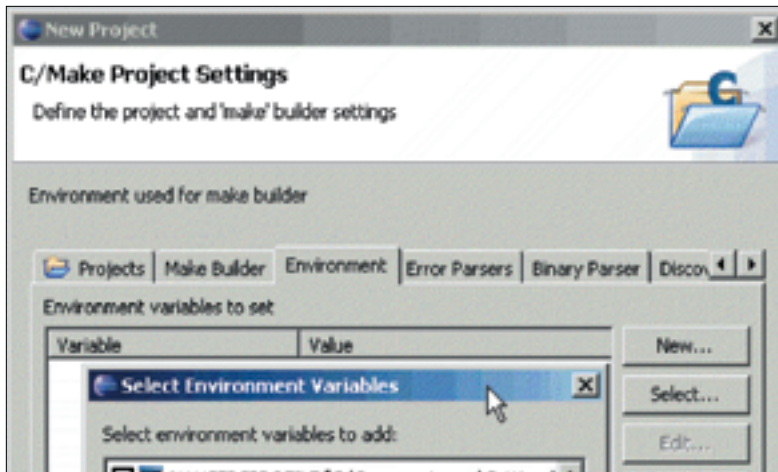DSchaefer@qnx.com
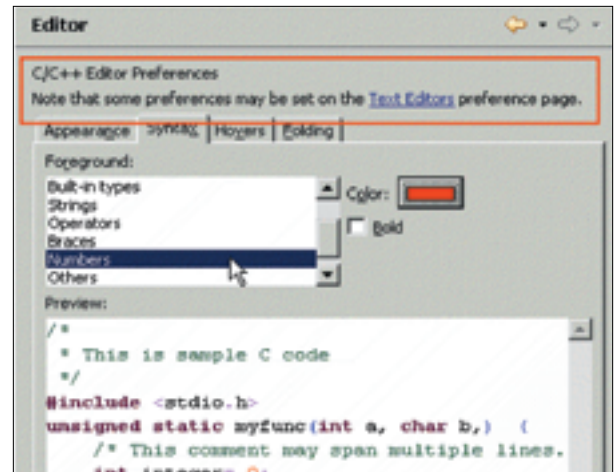
Figure 1: A view of the C/C++ project wizard

Figure 2: Adjusting user preferences for the C/C++ source-code editor

and semantic information that's captured by the CDT's built-in parsers for C and C++. This database persists between sessions in an index that the CDT builds as files are saved in the file system.

The DOM supports several source navigation features, including:
- *C/C++ Projects View* — Helps the developer navigate projects by providing a C/C++-centric view of source code, elements, and libraries.
- *Outline View* — Shows declarations from source files. If the developer clicks on a declaration, the source-code editor will "jump" directly to the source line where the declaration occurs.
- *Search features* — For looking up declarations, definitions, and references to identifiers defined in the source code or entered through dialogs. The developer can browse the search results. If the developer clicks on a result, the source code editor will jump to the corresponding point in the source file.
- *Source-code refactoring* — Lets developers change the structure of their code without changing its behavior. The most common form is rename refactoring, which propagates an identifier name change to all uses of that identifier. By using the DOM, the refactoring engine makes intelligent decisions to ensure that the changes don't cause compile errors.

The New Fast Indexer

For CDT version 3.1, a major restructuring has taken place surrounding the CDT's source indexing functionality that feeds information to many of the Source Navigation features. The new "Fast" indexer uses the CDT's internal parser to populate the index database. However, instead of doing full parsing as is done by

CDT's "Full" indexer, the "Fast" indexer takes a number of shortcuts to reduce the amount of work that needs to be done to generate the index information. The most important short cut is to reuse parse information between files, thus eliminating the need to parser header files every time they were included in other files. With these heuristics, the "Fast" indexer is able to parser Mozilla in about a quarter the time than the Full indexer. In addition, index updates when file changes occur also benefit from this reuse of information, resulting in update times of under a second in most cases. In all, this leads to much better UI response times while working on C/C++ projects..

Integrating Third-Party Tool Chains

Using the managed build system, vendors can provide descriptions of their tool chains, including option settings and the file extensions of the files that the tools will use as input and generate as output. The CDT uses this information to adapt the property dialogs, allowing options to be set per project and per file. The CDT also uses this information, along with user settings, to generate the build files, usually makefiles, that are used for the build. Out-of-the-box the CDT provides integration with the prolific gcc tool chain for host development.

To support the DOM, the CDT needs build information to properly parse the source files in a project. The main information required consists of the header-file include paths and the command-line macro settings used by the preprocessor. For managed build projects, the CDT gathers this information from the user settings on the tool options. The managed build system will also determine and supply the built-in settings for the compiler being used for the build.

For standard build projects, determining build information is more difficult since in this case the CDT doesn't manage the build settings for the project. Thus, the project properties dialogs allow the developer to manually enter the include paths and macros. The CDT also provides a mechanism to automatically discover these settings; this mechanism will scan the build output, try to determine the build commands that are used, and extract the build information from there. CDT 3.1 introduces the option of importing a previously run build from which to extract the information.

To support multiple tool chains, both the standard and managed build systems support the setting of environment variables that are
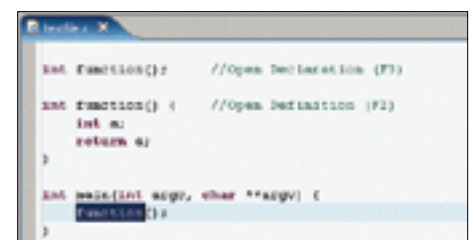

Figure 3: Source navigation in CDT 3.1 includes a new Open Definition feature that takes the developer directly to the definition of a given class or type.
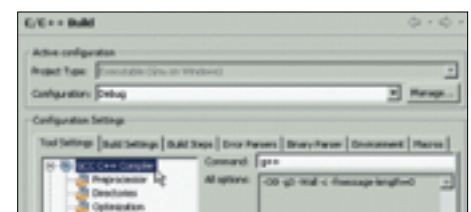

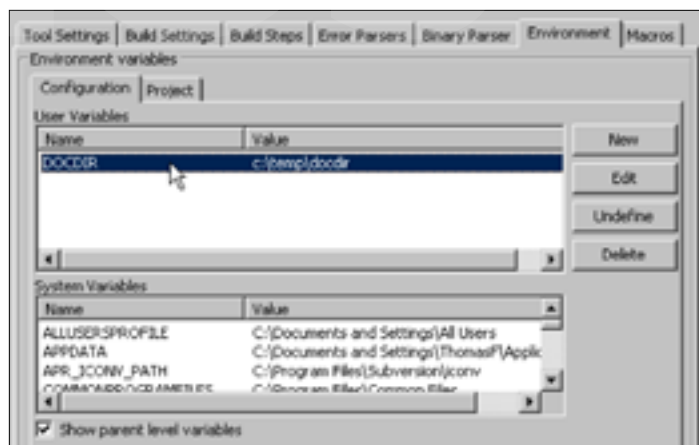Figure 4: The new C/C++ Build project preference dialog

Figure 5: Environment variables can be set per project and per configuration.
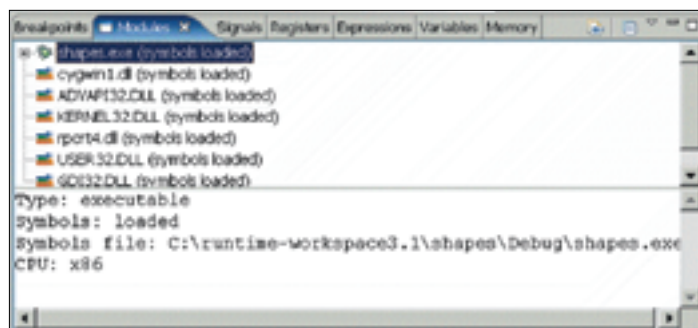

Figure 6: The new debugger modules view

exported to the environment before running the build. Using this facility, the developer can change the PATH used to find compilers and linkers and set other environment variables that the build tools may require.

CDT Debug

The debug component of the CDT provides visualization of the debug session. A native debugger such as gdb handles the nuts and bolts of executing the application, setting breakpoints, and extracting variable values. Out-of-the-box, the CDT supports integration with gdb using gdb's MI text-based protocol. The CDT also provides an extensibility mechanism that lets other debuggers be integrated in the same way.

The CDT provides standard Eclipse visualizations for debug, including a list of threads, stack frames for currently paused threads, and variable values. CDT 3.1 introduces a new memory view from the Eclipse platform that supports a number of formats for memory values and enables the developer to browse more than one location at a time.

CDT 3.1 also introduces a new modules view that shows the executable being debugged as well as the shared libraries and objects that the executable has loaded. The modules view can display any available symbol information, letting the developer set breakpoints on functions defined in the modules.

The CDT also provides a register view for C/C++ debugging. Version 3.1 extends this view by letting the developer organize registers into groups — especially useful for processors that have large numbers of registers.

Future Directions

With release 3.1, the CDT has reached a level of maturity that will ensure its wide-

spread adoption by C and C++ developers. Nevertheless, the CDT team has planned some new features that will help the CDT become an even better IDE.

While the new ctags-based indexer can build an index for source navigation in dramatically less time, it suffers from less-than-perfect accuracy and doesn't capture references to identifiers. To address this problem and improve the performance of CDT's parser-based full indexer, the CDT team has developed a new strategy that allows the parser to make use of the results stored in the index. This way the parser can support most CDT features by parsing each file only once after the file changes. This approach will greatly improve the performance and scalability of large projects.

An interesting new Eclipse project is Photran, which is dedicated to building a Fortran IDE based on the CDT. To create the first version, however, the Photran team needed to clone the code in the CDT and then make changes to it. While some parts of the CDT, such as the managed build system, already offer multi-language support, the CDT team will add new extension mechanisms to make it easier for language developers to add new language support. This should allow the Photran team to build directly on the CDT and let them leverage the source navigation features of the CDT.

Joining the CDT Community

The CDT has a large and diverse community that contributes in a variety of ways. Committers, the people who have the power to change code, include members from QNX Software Systems, Intel, Texas Instruments, IBM, and Wind River. Other contributors contribute patches on a smaller scale, though these contributions are still critical to the success of the
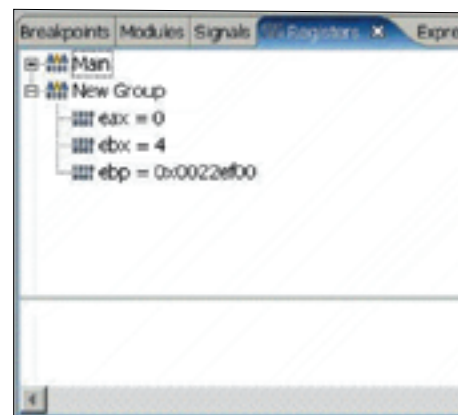
Figure 7: The new debugger registers view

CDT. These contributors include HP, Siemens, and Symbian. A number of other contributors contribute by testing, submitting bug reports, and requesting enhancements.

In October 2005, QNX hosted a CDT Contributors Summit in Ottawa. Members of over 20 major organizations participated, representing the embedded, enterprise, and scientific/research segments of the C/C++ market.

The CDT is used in many places and by many people. Recently, we recorded 130,000 downloads of the CDT in a two-month span, including 50,000 downloads of CDT 3.1.1 in the two weeks following its release. These numbers represent a significant portion of the Eclipse community in general, proving again that the scope of Eclipse extends far beyond Java development.

The CDT community is always open to more contributions. For up-to-date information and instructions on how to contribute, visit http://wiki.eclipse.org, then click on "CDT Project" in the projects list. Contributors also meet on the cdt-dev@eclipse.org mailing list to discuss technical issues and make program decisions.

» **Enterprise Solutions Built on**
OPEN STANDARDS

>> We can help you migrate to Open Source Solutions in no time. We have the Enterprise server and storage solutions you are looking for, at a price you never thought possible. By using open standard components and software, we can pass significant cost savings to you. And with our new state-of the-art, fully automated production facility, we can build and **SHIP** hundreds of machines daily, further saving you money by not having to wait weeks, or even months for your mission critical systems to arrive. All solutions are built to your specifications, loaded with your choice of software, and backed by a 3 year warranty. What are you waiting for? Let Open Source Storage show you what we can do!

www.OpenSourceStorage.com

Open
Source Storage ®

866-664-STOR
Sales@OSShpc.com
OpenSourceStorage.com
1195 Borregas Avenue,Sunnyvale, CA 94089

AMD 64 Opteron
The AMD64 Platform

Movielink/Optaros

Reducing risk

by Dave Gynn

Movielink (www.movielink.com) is an online movie download service offering U.S. broadband customers an extensive selection of recently released films and classic movies covering every genre for rent or purchase. Movielink, LLC was founded in 2002 as a joint venture of five major studios, Metro-Goldwyn-Mayer Studios, Paramount Pictures, Sony Pictures Entertainment, Universal Studios, and Warner Bros. Studios. Movielink content offers are drawn from the current releases and vast libraries of those studios as well as those of Walt Disney Pictures, Twentieth Century Fox, and leading independent studios.

About the Author
Dave Gynn has over 10 years of experience leading and delivering Web and client/server solutions in both start-up and enterprise environments. Dave's clients have included Charles Schwab, Lehman Brothers, NASCAR.com, GTECH, and Compaq. Recently Dave designed and developed a full Open Source software solution to assist in retirement planning for a financial services start-up. The solution included sophisticated financial calculations, dynamic chart generation, Web Service integration with partners, PDF generation, and live chat support. Dave was a Technical Architect of Acumen Sciences, and was an early member of Viant Corporation. He began his career at Seer Technologies.

The Movielink Web site is divided into two "stores," with a common home page. At the "Rental Store" customers can browse Movielink's user-friendly site and view trailers of available titles without charge. Once they're ready to rent a title, they register with Movielink and pay for the rental by credit card. Movielink's Movies in Minutes feature lets customers begin watching titles in two-10 minutes, or store them for up to 30 days and experience unlimited viewing for any 24-hour period. Customers can also use Movielink's MultiPlay feature to re-rent titles for additional 24-hour viewing periods for up to 30 days after the initial rental.

The other storefront, the "Purchase Store," functions just like the Movielink original "Rental Store," but the license to watch a movie gotten from the "Purchase Store" allows for unlimited viewing. The movie can be permanently stored on a hard drive to create a permanent archive or burned to a disc in Windows Media format for backup or playback on up to two additional tethered computers. The movies can also be downloaded to a notebook computer for traveling. Using a Media Center Edition PC, consumers can stream their copy of the movie to a TV set connected to a Media Center extender or Xbox.

Movielink's online catalog is at the center of the consumers' movie selection and shopping process, making it a key software application for Movielink. Since the launch of the Movielink Web site in 2002, the group has used customized packaged software developed specifically to create the online catalog. However, the company wanted to ensure that its processes and systems provide optimal efficiency in keeping its catalog updated.

Conventional Approach

Movielink first explored proprietary catalog, commerce, and content management software packages. However, while there are many packages that support retailers' print and online catalogs, Movielink found they weren't capable of managing its virtual inventory, complex availability windows (when movies can be sold), and pricing models. The company then considered building another custom system.

Optaros' Approach

Optaros initially looked at finding the right content management or e-commerce system that would support the emerging needs. What it discovered was that many business requirements called for both types of systems as well as needs that neither satisfied. Heavily customizing one kind of system or another wasn't going to meet the schedule or the budget. Optaros then looked at meeting the requirements by pulling together multiple Open Source software tools and assembling them to meet the requirements. By taking well-understood and proven Open Source tools and integrating them at standard integration points, Optaros could deliver new functionality without heavy customization. Assembling the application from Open Source components provided new functionality without the kind of investment that large custom development projects require.

Optaros' Solution

Movielink hired Optaros to design and develop its new catalog management system, assembling several Open Source software components. These components included the Apache Tomcat application server on Red Hat Linux, the Spring Framework Java/J2EE ap-

Dynamo Server	Tomcat
WCA	CMS
Nucleus	SpringFramework
Dynamo Repository	Hibernate

Deployment Mechanism

Dynamo Server	Tomcat
WCA	ADF
	Engine

Content Staging Database

Content Production Database

Administration/Staging Environment

Production Environment

plication framework, the Hibernate object/relational mapping framework, and the Acegi Security authentication system.

Many of these components use the Java programming language with which Movielink's internal IT staff is familiar, making it easier to maintain internally. Further, the solution was well integrated with Movielink's ATG-Dynamo-based front-end commerce system.

By having control of this system instead of customizing a packaged application, Optaros was able to utilize the best workflow to fit the client's business. For example, by integrating data onto one input screen at the proper point in Movielink's natural workflow, a higher accuracy rate was possible, and productivity increased. This solution wouldn't have been available using packaged software.

Business Impact

The key benefit is that Movielink could use a system that exactly fit its needs and delivered high productivity to the end users. "Optaros helped us to create a system that exactly matches our business requirements. Tasks that used to take hours now take seconds and with greater accuracy," said Bryan Spaulding, Movielink's chief technology officer. He added, "We've been able to manage a greater volume of movies much faster without adding staff. Our updated system will allow us to take on the projects we have planned for the next year."

The Optaros solution complemented Movielink's existing processes and systems that rapidly release the business functionality that is its competitive differentiator. In addition,

the Open Source-based systems that Optaros designed and delivered have given Movielink new features and more control in key areas. The level of performance increased, supporting more movies and titles.

Movielink has reduced its system risk – i.e., having a critical software system dependent on a single vendor whose future business viability is hard to determine. Optaros and Movielink also contributed back to the Open Source community by supplying certain improvements that they made to the underlying Open Source components.

Movielink has already recognized an added benefit of using Open Source software: the ability to reuse software and frameworks across applications. Code from the catalog management project was quickly reused in other internal application development projects without requiring those projects to purchase additional software licenses, saving time, testing, and software costs. ⭕

About Optaros

Optaros (www.optaros.com) is an international consulting and systems integration firm that provides enterprises with best-fit solutions to IT business challenges, maximizing the benefits of Open Source software. Whether your organization has had little exposure to Open Source software or has Open Source-based solutions in place, Optaros has the business- and IT problem-solving experience to advise on the areas where Open Source and open standards can be effective, and how to manage the use and benefits of Open Source software proactively.

Optaros offers a third alternative to the "build versus buy" decision with our proven assembly methodology (OptAM). In our core practice areas, Optaros ensures successful solution delivery by leveraging our pre-selected Open Source solution sets and customizing to your specific business requirements.

Cross-Browser Client-Side AJAX

Other than using the **XMLHTTP request object,** how does one go about writing an **AJAX** application?

by Edmond Woychowsky

Everyone has one or two words or phrases that people associate with them. For example, I always think of my friend Rob eating squid when I hear the words "What do you mean it's not chicken?" Just thinking about those words is enough to transport me through the years to that moment. And the look on his face still makes me smile.

Of course there can be some painful associations too. Like the words "I could code it in about an hour," which will forever remind me of a former manager who was fond of boasting about his prowess as a developer. The fact that he could barely turn a machine on never entered into it. In his mind he was the greatest developer who ever lived and the fact that some tasks took others more than an hour only served to reinforce his belief.

Somewhere between these two poles are the feelings I get when I hear the words "It doesn't blink." That was what the guy kept saying over and over again as I demonstrated an AJAX application. What this person, who remained true to his mainframe roots, was attempting to say was that the page wasn't going through the normal unload/reload cycle.

Because I'm not fond of being kept in the dark myself I made an effort to explain how the application worked. It went something like this: "Instead of using the browser's standard unload/reload cycle to communicate with the Web server client-side JavaScript uses the XMLHTTP request object to communicate directly with the server. This means that less information is passed back and forth so that this kind of application is inherently faster than a traditional Web application. It doesn't 'blink,' as you so put it, because the client-side JavaScript is only updating the parts of the page that changed instead of the whole page."

This was several years ago, years before Jesse James Garrett's AJAX article, and looking back now I suppose I expected too much of most of the people in the room. For some reason I expected immediate acceptance of the

application I had written by everyone there. My expectations were dashed when the "it doesn't blink" guy latched on a single phrase like a pit bull, the phrase was, client-side JavaScript.

Because of this phrase, the mainframe developer, still reeling from the job market downturn after Y2K, uttered a phrase of his own in the form of a question, "So, it's a mockup?"

"No, it's communicating with the Web server using client-side JavaScript and the XMLHTTP request object. It's a working application," I replied.

"But it doesn't blink!" he said.

This banter continued for about two hours and would probably still be going on had one user not made a suggestion. He wanted the application to look up a specific item of information. So we did. When the information was displayed in the Web browser, he uttered a remark of his own, "It's not a mockup."

The user elaborated by saying that he had entered the information we displayed five minutes before the meeting began. He earned a glare from the mainframe developer and my undying gratitude.

Until recently scenes like this were pretty much par for the course when demonstrating AJAX applications. But now, since Garrett's AJAX article, everyone wants to jump on the AJAX bandwagon. The rapid adoption of AJAX leaves only one question: other than using the XMLHTTP request object how does one go about writing an AJAX application?

Unfortunately, there's no simple answer, but there's a way to make the question simpler by concentrating on one aspect of it. Look at the problem as putting a Web front-end on a Web Service. So, since we're hip-deep in Web Services – or at least books and articles about how to create Web Services – let's look at creating a pretty front-end for a Web Service.

Because most Web Services use SOAP, the Simple Object Access Protocol, we'll start by looking at the SOAP request:

About the Author
Ed Woychowsky is the author of the new book AJAX: Creating Web Pages with Asynchronous JavaScript and XML published by Prentice Hall as part of the Bruce Perens' Open Source Series in August 2006. ISBN 0-13-227267-9. For more information, please visit: www.prenhallprofessional.com

Method/Property	Description
abort()	Terminates the previous outstanding request.
getAllResponseHeaders()	Returns all response headers, labels, and values as a string.
getResponseHeader("label")	Returns the value for the provided label.
open("method","url", asynchronous, "username","password")	Opens/assigns a method; "GET," or "POST" and, optionally, an asynchronous indicator.
send(content)	Sends the request with optional content, which can either be string or DOM data.
setRequestHeader("label","value")	Sets a request header label/value pair.
onreadystatechange	Event handler for asynchronous requests, fires on each change to the readyState property.
readyState	Status of the request as an integer. 0 = uninitialize. 1 = loading. 2 = loaded. 3 = interactive. 4 = complete.
responseText	String returned from the server.
responseXML	XML document returned from the server.
status	HTTP response code returned from the server.
statusText	String message associated with the HTTP response code.

Table 1: XMLHTTP methods and properties

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.
w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
 <soap:Body>
   <getItems xmlns="http://tempuri.org/">
     <guild_item_id>string</guild_item_id>
     <guild_id>string</guild_id>
   </getItems>
 </soap:Body>
</soap:Envelope>
```

Value	Meaning	Description
1	Unintialized	Nothing meaningful has happened with the object yet.
2	Loading	The server's response is being received.
3	Interactive	Enough of the response from the server has been received that the response can be examined. However, it's not complete.
4	Complete	The entire response has been receive and it's complete.

Table 2: XMLHTTP request object ready states

As incomprehensible as the above SOAP request appears to those unfamiliar with the protocol it's actually quite simple. Most of what you see above is a wrapping that identifies it as a SOAP request, essentially any element with the namespace "soap." The stuff that works with the Web Service, getItems, is with the soap:Body tags and it invokes a Web Service named date passing the value "string" in the guild_item_id and guild_id parameters.

All right, now that we know what to send, the next question is how to create an XMLHTTP request object on the client-side to send it to the Web server? This is one of those times when a line in my old resume comes into play. No, not the one about being an escaped mental patient, the one about cross-browser development, which is roughly the same thing. Yes, there's Microsoft's way and then there's everyone else's way.

Why do I feel like I just said a bad word in the pages of LinuxWorld? I feel dirty.

One of the differences between my favorite browser, Firefox, and Internet Explorer is something called ActiveX. Only Microsoft uses it, so it can't be found in Firefox, Flock, or SeaMonkey. And that means that a little more coding is required to consistently create an XMLHTTP request object. What we end up with is code like this:

```
try {
    objXMLHTTP = new ActiveXObject('Microsoft.XMLHTTP');
}
catch(e) {
    objXMLHTTP = new XMLHttpRequest();
}
```

Not, bad, but to quote Michael Kane, "What good is a mace if you don't use it?" But before we use it let's take a look at all of the bells and whistles that go along with it (see Table 1).

I'd like to take a moment before continuing to point out the onready-statechange property. It's there that we provide a handler for when the XMLHTTP request object's readystate changes. Or, in other words, the
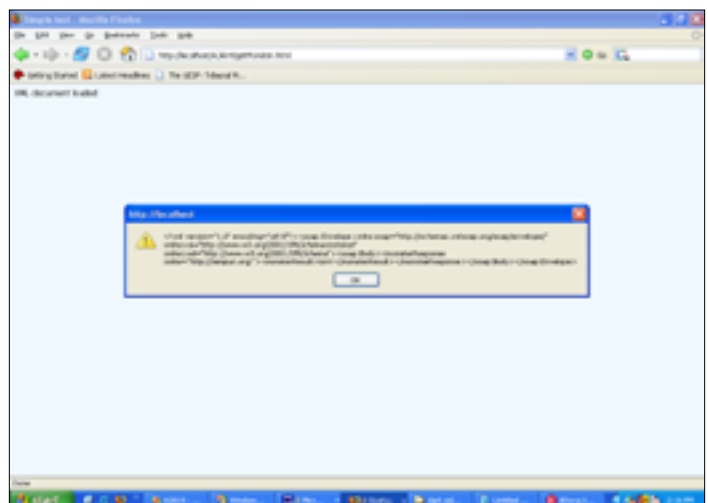

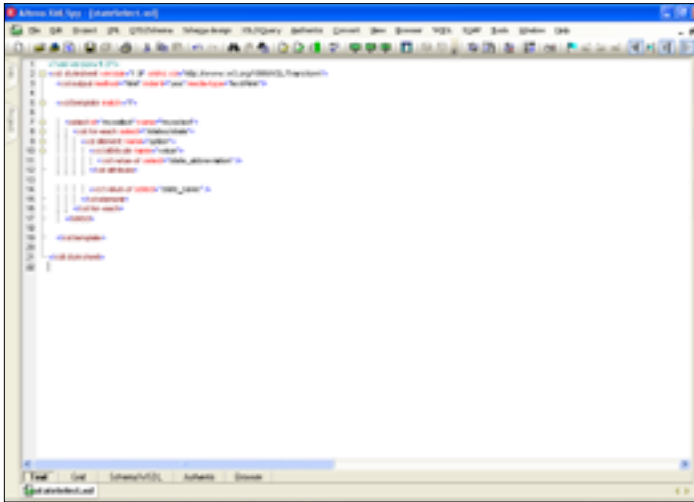**Figure 1:** Listing 1 rendered in Firefox

Figure 2

server did something. The possible server responses, along with their meanings are described in Table 2.

The onreadystatechange property and ready states are only used for asynchronous requests. Since we're busy people and this is supposed to be an article about AJAX we'll concentrate on the asynchronous request as opposed to a synchronous request. The required code is shown in Listing 1 with the results in Figure 1.

Just in case you were wondering, there are only two differences between an asynchronous request and a synchronous request. The first is that the true is false. And the second is that an onreadystatechange event handler isn't needed. The browser will wait for the server's response.

All right, for argument's sake let's say that the onreadystatechange event handler has the server's response and that that the ready state is complete. It doesn't mean that we're done, yes, the information is on the browser, but unless the user has some kind of weird mutant power enabling him to see the invisible, the information isn't in a very useful format. Sitting in an object hidden from view, yeah, that's helpful. Fortunately there are, however, a number of options available that can make the information a wee bit more user friendly.

However, before displaying anything on the browser it might be a good idea to make sure it's what we're expecting. After all, it's within the realm of possibility that we don't get a good response like this one:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.
w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
 <soap:Body>
   <getItemsResponse xmlns="http://tempuri.org/">
     <getItemsResult>xml</getItemsResult>
   </getItemsResponse>
 </soap:Body>
</soap:Envelope>
```

We could, in fact, get a SOAP fault like this:

```
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.
w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
 <soap:Body>
```

Listing 1: Asynchronous Example

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
  <head>
    <title>Simple test</title>
    <link rel="stylesheet" type="text/css" href="common.css"/>
    <script language="javascript">
<!-- <![CDATA[
var objXMLHTTP = createXMLHTTP();
var objXML;
var soap;

function setEvents() {
  objXMLHTTP.onreadystatechange = asyncHandler;

  soap = '<?xml version="1.0" encoding="utf-8"?>';
  soap += '<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-in-
stance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://sche-
mas.xmlsoap.org/soap/envelope/">';
  soap += '<soap:Body>';
  soap += '<monster xmlns="http://tempuri.org/" />';
  soap += '</soap:Body>';
  soap += '</soap:Envelope>';

  objXMLHTTP.open('POST', 'http://localhost/AJAX4/myService.asmx', true);
  objXMLHTTP.setRequestHeader('SOAPAction','http://tempuri.org/monster');
  objXMLHTTP.setRequestHeader('Content-Type','text/xml');
  objXMLHTTP.send(soap);
}

function asyncHandler() {
  if(objXMLHTTP.readyState == 4) {
    document.getElementById('xxx').innerHTML = 'XML document loaded';
    alert(objXMLHTTP.responseText);
  }
}

function createXMLHTTP() {
  try {
    return(new XMLHttpRequest());
  }
  catch(e) {
    return(new ActiveXObject('Microsoft.XMLHTTP'));
  }
}
// ]]> -->
    </script>
  </head>
  <body onload="setEvents()">
    <div id="xxx"></div>
  </body>
</html>
```

Listing 2: A Different Approach to AJAX

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
  <head>
    <title>Simple test</title>
    <style>
xml
{
  display: none;
  font-size: 0px
}
    </style>
    </style>
    <script language="javascript">
<!-- <![CDATA[
var objXMLHTTP;                      // XMLHTTP request object
var objXML;                         // XML DOM document
var objXSL;                         // Stylesheet
var objXSLTProcessor;               // XSL Processor
var xslt = 'stateSelect.xsl';        // Path to style sheet

function setEvents() {
  objXMLHTTP = createXMLHTTP();

  try {
   objXML = new ActiveXObject('MSXML2.FreeThreadedDOMDocument.3.0');
   objXSL = new ActiveXObject('MSXML2.FreeThreadedDOMDocument.3.0');

   objXML.load(document.getElementById('xmlDI').XMLDocument);
  }
  catch(e) {
   var _objParser = new DOMParser();

   objXSLTProcessor = new XSLTProcessor();
   objXML = _objParser.parseFromString(document.getElementById('xmlDI').
innerHTML,"text/xml");
  }

  objXMLHTTP.onreadystatechange = asyncHandler;

  objXMLHTTP.open('GET',xslt,true);
  objXMLHTTP.send(null);
}

function asyncHandler() {
  var strXHTML;

  if(objXMLHTTP.readyState == 4) {
   try {
          var objXSLTemplate = new ActiveXObject('MSXML2.XSLTemplate.3.0');

          objXSL.loadXML(objXMLHTTP.responseText);
          objXSLTemplate.stylesheet = objXSL;

          objXSLTProcessor = objXSLTemplate.createProcessor;
          objXSLTProcessor.input = objXML;
```

```
          objXSLTProcessor.transform();

          strXHTML = objXSLTProcessor.output;
   }
   catch(e) {
          var _objSerializer = new XMLSerializer();

          objXSL = objXMLHTTP.responseXML;

          objXSLTProcessor.importStylesheet(objXSL);

          strXHTML = _objSerializer.serializeToString(objXSLTProcessor.
transformToFragment(objXML, document));
   }

   document.getElementById('xxx').innerHTML = strXHTML;
  }
}

function createXMLHTTP() {
  try {
    return(new XMLHttpRequest());
  }
  catch(e) {
    return(new ActiveXObject('Microsoft.XMLHTTP'));
  }
}
// ]]> -->
    </script>
  </head>
  <body onload="setEvents()">
    <xml id="xmlDI">
      <states>
        <state>
          <state_abbreviation>AB</state_abbreviation>
          <state_name>Alberta</state_name>
        </state>
        <state>
          <state_abbreviation>AK</state_abbreviation>
          <state_name>Alaska</state_name>
        </state>
        <state>
          <state_abbreviation>AL</state_abbreviation>
          <state_name>Alabama</state_name>
        </state>
        <state>
          <state_abbreviation>AR</state_abbreviation>
          <state_name>Arkansas</state_name>
        </state>
      </states>
    </xml>
    <div id="xxx"></div>
  </body>
</html>
```
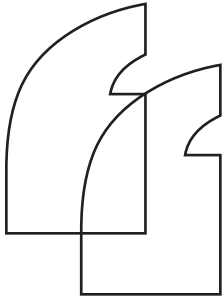
> Creating an XSLT processor is one step in Firefox, while it's multi-step in Internet Explorer; **basically anything to do with XML is different from browser to browser**

```
<soap:Fault>
    <faultcode>soap:MustUnderstand</faultcode>
    <faultstring>Mandatory Header error.</faultstring>
    <faultactor>http://localhost/AJAX4/chapter4.asmx</faultactor>
    <detail>Web Service coffee break.</detail>
</soap:Fault>
</soap:Body>
</soap:Envelope>
```

Again for argument's sake let's assume that the response was success-ful, because handling a SOAP fault actually depends on what we were attempting to accomplish in the first place. With AJAX it's kind of tra-ditional when handling a successful response from the server to show something on the user's browser, so that's what we'll do. Which leads me to the question how?

Well, first it's necessary to extract the part of the XMLHTTP request object's response that we're interested it. While there are several ways to do this extraction ranging from applying a regular expression to the text response to selecting individual nodes from the XML response, I'll use the XML response. Hey, why not, the "X" in AJAX stands for XML, might as well use it, and leave the text response for debugging with alerts.

If you haven't noticed yet, when creating an AJAX application from scratch there always seems to be just one more thing to do. In this case it's display the results in the Web browser, which depending on the re-sult can be done in several ways. For instance, the innerHTML property is a real good possibility. Basically it would look something like this:

```
document.getElementById('tagid').innerHTML = whatever;
```
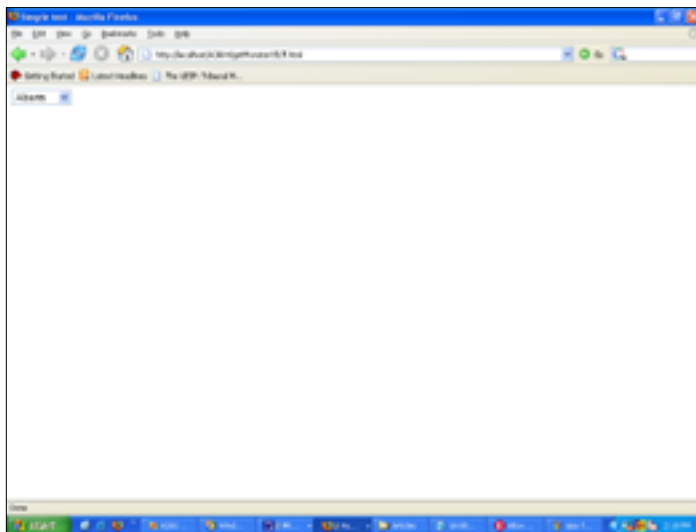

Figure 3

Another possibility is to display the results using the HTML Docu-ment Object Model's appendChild method like this:

```
document.getElementById('tagid').appendChild(whatever);
```

Just in case you're wondering, whatever is in the examples above, it's essentially whatever you'd like it to be. It could be text, as in the value of a particular XML node, or it could be XHTML that was created on the server. I've even seen the XMLHTTP request object's responseText prop-erty used so that regular, not well formed, HTML could be used. While this might sound odd it's a good way to get a little extra mileage out of existing applications.

However, I'm personally fond of living dangerously. Yes, I'm one of those developers who's been accused of dabbling in the dark arts by some other developers; yes, I use XSLT on the client. Before running off to hide from the scary man, let me explain that if a Web browser supports the XML the odds are close to 100% that it also supports XSLT. Interested?

AJAX and the Dark Arts

Good, but I should warn you that certain things are easier in one browser than the other. For example, creating an XSLT processor is one step in Firefox, while it's multi-step in Internet Explorer; basically anything to do with XML is different from browser to browser. One of these differences is something called XML data islands, introduced in Internet Explorer 5.0, it's a way of embedding XML in the body of an HTML document. I've decided to use this technique as well as demon-strate that the XMLHTTP request object can be used without SOAP to 'GET' a file directly from the Web server. The end result of this endeavor is shown in Listing 2 and Figure 2, look it over and I'll be right here to explain some oddities.

I'd like to point out several things about Listing 2. First, the Cascad-ing Style Sheet, whose sole purpose it to insure that the XML data island doesn't display in the browser. Another item is how the serialization and de-serialization of XML is handled by the individual Web browsers. Working around these differences is often enough to push one over the edge, but once the code exists it's merely a matter of cloning it as often as needed. Although, being a lazy, er, efficient type, I usually create JavaScript classes for this kind of thing.

Conclusion

What I've shown here is just a taste of some of the power and prob-lems that one encounters when developing an AJAX application. And while developing the first application is often time-consuming, once these techniques have been mastered subsequent applications come a lot easier than the first. Yes, hard work is its own reward, although having a hot buzzword on the resume is good. Come to think of it the paycheck is nice too. ⬤

Maven: A Different Way of Looking at Software Development

A real-time engineering framework

by Jason van Zyl

Software development is typically carried out in an opaque environment where progress can be slow and (too often) the resulting build processes lack visibility, transparency, and collaboration. The Apache Software Foundation's Maven project approaches the problem of building software by providing the technical underpinnings for a set of development methods that enable engineers and other stakeholders in a project to optimize build reliability, accelerate build velocity, and capture and share build knowledge. Maven differs from current script-based approaches to building software by first defining, standardizing, and then publishing the build process as a logically organized structured lifecycle. Having evolved from the Open Source development world of distributed, asynchronous, iterative, and highly component-based engineering, Maven's technology helps development teams effectively cooperate to create and deliver successful software projects consistently.

While people new to Maven tend to think of it as strictly a tool for builds (akin to Ant), Maven actually offers a set of tools, including a Project Object Model (POM), user-defined plug-ins, and functionally rooted best practices, whose goal is to measurably improve software development processes as a whole, rather than simply enable builds to compile.

But while there's definitely a "non-technical" element to developing software with Maven, its strong use of patterns and emphasis on community-oriented practices, the key to building with Maven comes from how its tooling technology actually enables and reinforces development best practices across a team. As anyone who has ever been part of a successful project knows, teams must creatively combine technical and non-technical elements to deliver quality projects. Maven's goal, in terms of its tooling, is to provide a solid and cohesive technical framework that can mitigate some of the human factors that hinder software development (e.g., one developer's code contribution breaking the build, non-standard locations for relevant cross-related project resources, and the inability to integrate new tools into the build process). Moreover, once engineers are relieved of the more mundane, time-consuming aspects of the build process, they can get on with the important work of creating value at the application level.

So let's take a closer look at the Maven project and explore how, when combined with good internal build procedures, its use can improve overall software development processes:
- Apache Maven 2.0, the foundational POM-based build engine for Java projects
- Apache Continuum 1.0, the Continuous Integration server fully integrated with Maven
- Plug-ins, the standard way to work in Maven and extend its functionality
- Maven SCM, the abstraction layer for SCMs in Maven's plug-ins and in Continuum, which supports CVS and SVN, as well as Perforce, Starteam, and ClearCase
- Maven Wagon, the abstraction layer for general transport, heavily used in Maven and the Maven-Ant tasks library for moving artifacts between repositories
- Doxia, the content generation library used to generate project sites and create build reports
- Surefire, the abstraction layer for testing frameworks that support JUnit and general scripting (TestNG support is in development)
- Archetypes, a rapid project prototyping tool that quickly sets up a template Maven project

Typically, developers who come to Maven are already familiar with Ant, so there's a natural association made between the two projects. Ant is a script-based toolbox, and an excellent one at that. However, Maven isn't simply a build tool and, as such, it's not a replacement for Ant. Instead, Maven combines its POM, plug-ins, Continuum, and embedded build lifecycle phases (which can actually include Ant tasks) to offer engineers a software development infrastructure that delivers project visibility, re-usability, maintainability, and comprehensibility.

About the Author
As chief architect and co-founder of Mergere, Inc., Jason van Zyl focuses on improving the software development infrastructure associated with medium to large-scale projects, hich has led to the founding of the Apache Maven project. He continues to work directly on Maven and serves as chair of the Apache Maven Project Management Committee.

Originally Maven arose from a very practical need to make several Apache projects functionally build in the same way. Then, as now for most developers, being able to freely move between multiple projects meant that each developer needed to clearly know and understand how each of the individual projects worked. This meant that for every project a developer needed to build, he had to repeatedly decipher and learn not only how to build the project in question, but also become familiar with its related testing, documentation, reporting, and deployment practices. Since all of the original Apache projects shared a number of build characteristics (lifecycle phases), Maven was developed to harness these commonalities. Given that all software projects need to be built, tested, packaged, documented, and deployed at the most base level – Maven was created to logically organize this build model and then orchestrate the process. Since there are also a great number of sub-functions possible at each phase of the build, test, package, document, and deploy lifecycle, Maven's build model lets users use (and create) plug-ins to meet build-specific requirements. But, at the same time, Maven constrains the overall build process to a single recognizable and repeatable model, creating a build framework (versus a build tool) that saves developers from having to (re)learn unique project builds time and time again.

Use of Patterns

Maven's build pattern emphasis was heavily influenced by the Gang of Four's design patterns that in turn borrowed heavily from Christopher Alexander's work in the realm of architecture. According to Alexander, "... patterns help create a shared language for communicating insight and experience about problems and their solutions." Maven's goal is to define a build structure so that problems can be approached consistently in terms of a common framework. By providing a lingua franca build, Maven facilitates build compre-

hension based on the following underlying framework concepts of:
• Convention over configuration
• Model-driven declarative execution
• Standard directory layout
• Build lifecycle

Each of these concepts helps developers deal with their build infrastructure at a higher level of abstraction, allowing more effective and timely communication. Let's explore each of these concepts further.

Convention over Configuration

One of Maven's central tenets is to provide logical default strategies for the most common build tasks, so that developers can avoid the mundane details. This notion is known as "convention over configuration" and has been popularized by the Ruby on Rails (ROR) community, specifically encouraged by ROR's creator David Heinemeier Hansson, who summarizes the notion as follows:

"Rails is opinionated software. It eschews placing the old ideals of software in a primary position. One of those ideals is flexibility. The notion that we should try to accommodate as many approaches as possible, that we shouldn't pass judgment on one form of development over another. Well, Rails does, and I believe that's why it works.

With Rails, you trade flexibility at the infrastructure level to gain flexibility at the application level. If you are happy to work along the golden path that I've embedded in Rails, you gain an immense reward in terms of productivity that allows you to do more, sooner, and better at the application level.

"One characteristic of opinionated software is the notion of "convention over configuration." If you follow basic conventions, such as classes are singular and tables are plural (a person class relates to a people table), you're rewarded by not having to configure that link. The class automatically knows which table to use for persistence. We have a ton of examples like that which

all add up to make a huge difference in daily use."

Hansson articulates very well what Maven strives to deliver: using standard conventions saves time, makes it easier to communicate to others, and allows engineering teams to get to the important work of creating value in their applications faster and with less effort. With Maven, developers won't spend a lot of time getting a development infrastructure functioning, instead they "slot in" various pieces where required, while Maven takes care of the tasks (goals) for that lifecycle phase. Maven handles all the building, document generation, or deployment work through its POM.

Model-Driven Declarative Execution

In Maven the POM is the central axis of work, since Maven is – by design – project-centric. Everything Maven needs to work with a given project is stored in the POM. Further, Maven uses the POM to execute and drive all of a project's build-re-lated functions. Maven's model-driven execution relies on the POM (an XML-based metadata description of a project), which is stored in a standard location with its related project.

In short, the POM contains every important piece of information about your project and is the "one-stop shop" for finding anything related that project. Below is a simple example POM that contains the key elements common to all POMs:

```
<project>
<modelVersion>4.0.0</modelVersion>
<groupId>com.mycompany.app</groupId>
<artifactId>my-app</artifactId>
<packaging>jar</packaging>
<version>1.0-SNAPSHOT</version>
<name>Maven Quick Start Archetype</name>
<url>http://maven.apache.org</url>
<dependencies>
<dependency>
<groupId>junit</groupId>
```
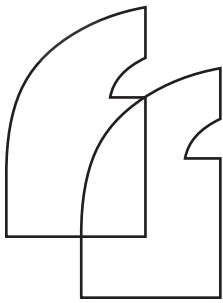
Maven's real-time engineering framework supplements community-oriented best practices to measurably improve **the predictably and velocity of software development and not just software builds**

```
<artifactId>junit</artifactId>
<version>3.8.1</version>
<scope>test</scope>
</dependency>
</dependencies>
</project>
```

The POM is typically stored in a file named pom.xml and sits at the top level of your project. The following are descriptions of the key elements in the POM above:

- **project** – the top-level element in all Maven pom.xml files
- **modelVersion** – indicates the version of the object model this POM is using (the model version itself changes infrequent-ly but it's mandatory to ensure stability if/when the Maven community deems a model change necessary)
- **groupId** – indicates the unique identifier of the organization/group that created the project (groupId is one of the key iden-tifi-ers of a project and is typically based on the fully qualified organization domain name, e.g., org.apache.maven.plugins is the groupId for Maven's plug-ins)
- **artifactId** – indicates the unique base name of the primary artifact being gener-ated by this project, typically a JAR file (a typical artifact would have the form <artifactId>-<version>.<extension>, e.g., myapp-1.0.jar)
- **packaging** – indicates the package type to be used by this artifact (e.g., JAR, WAR, EAR, etc. and describes that the artifact produced is JAR, WAR, or EAR, but can also indicate the specific lifecycle to be used)
- **version** – indicates the version of the artifact generated by the project.
- **name** – indicates the display name used for the project (used in Maven-generated documentation)
- **url** – indicates where the project's site can be found (used in Maven-generated docu-mentation)
- **description** – provides a basic description of the project (used in Maven-generated documentation)

Standard Directory Layout

Another tenet Maven follows to simplify project development is the use of standard directory layouts for project sources, addi-tional project resources, and documentation (generally referred to as "project content"). Maven encourages a common arrangement for project content so engineering teams can quickly and predictably find any project-re-lated items whenever Maven builds projects. This tenet dovetails with Maven's "convention over configuration" concept. So, while the location of the project content isn't critical to a project's success, it is critical that every member of that project team (both now and in the future) knows exactly where to look to find the project's content. Maven's standard direc-tory structure ensures that developers can become familiar with a project tree and easily find project content simply by navigating that project's Maven structure.

Build Lifecycle

Recall earlier we said that while projects can have nearly infinite variability, Maven binds these potential inconsistencies and discrepancies in the well-defined structure of its build framework. The framework that Maven uses to accommodate project varia-tions is defined as the build lifecycle. In Maven this frame-work consists of an ordered series of lifecycle "phases," where each phase can perform one or more actions (a k a "goals" in Maven-speak). The build lifecycle's phases use highly abstracted names such as compile, test, package, and deploy. While any number of goals (actions) can be performed during the test phase, the framework for the lifecycle itself remains well-defined and clearly articulated for any developer working on the project. Dur-ing testing, for example, an engineer may need to run unit tests, create databases, deploy EAR files to an application server, and run coverage tools. With Maven, there are no restrictions as to the number or type of goals required; there is simply a convention (an embedded best practice) that defines how, where, and when these actions will be processed – specifically in the test phase, as it's defined in Maven's build lifecycle.

Visibility and Collaboration

Consider your own development environ-ment, where everyone is constantly rooting around trying to figure out where all the differ-ent bits and pieces that make up a given proj-ect are located. This means there's very little opportunity for the team, or any individual en-gineer, to comprehend the project as a whole. While Maven's basic concepts of convention over configuration, model-driven declara-tive execution, standard directory layout, and build lifecycle all support build comprehen-sibility and maintainability, the next step in improving the software development lifecycle requires build visibility and reusability.

Without visibility, it's unlikely that an individual engineer will ever know what another engineer has contributed. As such, there's a good chance that useful code won't be leverageable – forcing teams to lose out on the network effects of code reusability. Even a well-maintained build can end up being a silo if there's no way to effectively share that project knowledge widely. As a result, Maven provides standard reports and a continuous integration server, Continuum, for the spe-cific purpose of improving project visibility. Maven's reporting capabilities, when used in conjunction with Continuum, provide real-time metrics regarding not only the overall health of any Maven-built project, but also fine-grained details about code coverage, code quality, developer activity, and project issues.

Maven's goal of delivering project visibility, re-usability, maintainability, and comprehen-sibility takes it beyond the confines of being a build tool. It ensures that organizations and teams can leverage not only all project intelligence, but also the ongoing health as-sessments of their projects. Maven's real-time engineering framework supplements commu-nity-oriented best practices to measurably im-prove the predictably and velocity of software development and not just software builds.

Complex JAVA J2EE Hosting made easy.

WebAppCabaret ᔆᵐ

http://www.webappcabaret.com/jdj.jsp

1.866.256.7973

JAVA J2EE-Ready Managed Dedicated Hosting Plans:

Xeon III
SAMEDAY SETUP

Dual 3.2 GHz Xeons
4GB RAM
Dual 73GB SCSI
1U Server
Firewall
Linux
Monitoring
NGASI Manager

$**399**
monthly

Pentium 4 I
SAMEDAY SETUP

FREE SETUP

2.4 GHz P4
2GB RAM
Dual 80GB ATA
1U Server
Firewall
Linux
Monitoring
NGASI Manager

$**199**
monthly

2nd month FREE

4Balance I

1 Database Server
and 2 Application
Servers connected
to 1
dedicated
load
balancing device.
Dual Xeons.
High-Availability.

$**1724**
monthly

NEW! **Geronimo 1.0 Hosting . Virtual Private Servers(VPS) starting at $69/month**

At **WebAppCabaret** we specialize in **JAVA J2EE Hosting**, featuring
managed dedicated servers preloaded with most open source JAVA technologies.
PRELOADED WITH:
JDK1.4 . JDK1.5 . Tomcat . Geronimo . JBoss . Struts . ANT . Spring . Hibernate
Apache . MySQL . PostgreSQL . Portals . CRM . CMS . Blogs . Frameworks
All easily manage via a web based control panel.
Details:
-All Servers installed with the latest Enterprise Linux
-Firewall Protection
-Up to 60 GB daily on site backup included at no extra charge per server.
-Database on site backup every 2 hours
-Daily off site database backup
-A spare server is always available in case one of the server goes down
-Intrusion detection.
-24x7 Server and application monitoring with automatic self healing
-The Latest Bug fixes and Security updates.
-Tier 1 Data Center. 100% Network Uptime Guarantee
-Guaranteed Reliability backed by industry-leading Service Level Agreements

Log on now at **http://www.webappcabaret.com/jdj.jsp** or call today at
1.866.256.7973

NGASI POWERED

WebAppCabaret ᔆᵐ

JAVA J2EE Hosting

Eclipse: a Solid Desktop, Rich-Client, or Embedded Application Framework

A general purpose platform

by Todd Williams

By now, you've probably heard about Eclipse as "the Open Source Java IDE" (http://www.eclipse.org). Today, several companies have looked past the Java IDE plug-ins provided as part of Eclipse, and are creating products that use Eclipse as a tool integration platform, both inside and outside of the Java arena. But what about using royalty-free, Open Source Eclipse technology as a general-purpose application framework for your next desktop, fat client, or embedded application? With the support provided by the Eclipse Rich Client Platform (RCP) and the embedded version of the same (eRCP) the idea is certainly not as strange as it first sounds. So we'll explains why Eclipse is a solid desktop, rich-client, or embedded application framework with the potential to greatly simplify and accelerate development as well as forever change the way developers think about writing Java applications.

About the Author
Todd Williams is Genuitec's VP of technology and leads its Eclipse Technology Consulting Practice. He has over 20 years of industry experience in developing computing infrastructures, large-scale distributed software architectures, and optimizing development processes, techniques, and tools. Todd has been Genuitec's representative to the Eclipse Foundation since 2002 and currently holds an elected seat on the Eclipse Foundation's board of directors.

Software development is often about compromises. Currently, one of the most visible debates centers on the tradeoffs between productivity applications and browser-based UIs. In spite of what current media coverage might lead one to believe, the industry hasn't decided to throw away all its desktop applications in favor of browser-based UIs rendered with some combination of HTML/XML/XSLT/Javascript. The reason can be summed up in three words: "the user experience." Form follows function... is the key criteria for judging usability. In practice, high user interactivity or complex data relationships make delivering user interfaces as a desktop application less of a choice and more of a requirement.

In today's computing environments it's important to deliver user interfaces that can run on a wide variety of platforms. The range is broad – including small handheld devices as well as server consoles. When users interact with applications in the window management environments they're most familiar with, using the application must feel natural and predictable.

Building a productivity application means starting with a good design and a supportive architecture. Since there's no universally accepted application framework, most developers design their own architecture and then build it into a framework. However, the cost of this approach is considerable expense, time, debugging, support, and aggravation expended on solving a problem that's peripheral to building the functionality of the intended application.

A much better approach than "rolling your own" application framework would be to find one that could fulfill the design requirements while simplifying and accelerating project development. A "wish list" for such a framework would likely contain the following:

- Implements a clear, consistent, and cohesive architecture
- Supports development and execution on all the major desktop platforms (Windows, Mac OS X, Linux, QNX Photon, Pocket PC, HP-UX, AIX, Solaris)
- A snappy UI response that maintains the platform's native look-and-feel
- Provides a large variety of widgets, both standard (i.e., button and checkbox) and extended (i.e., toolbar, tree view, and progress meter)
- Provides extensive text processing that includes editors, position/change management, rule-based styling, content completion, formatting, searching, and hover help
- Supports using platform-specific features (i.e., ActiveX) and legacy software, if desired
- Enables branding the application
- Contains an integrated help system
- Manages user configuration and preferences
- Supports remote discovery and installation of application updates
- Created and backed by respected software

companies experienced in creating object oriented frameworks

- Supports internationalization and national language translation
- Designed for flexibility with natural features for adding new functionality
- "Pay" only for what you need – base frameworks can be easily reduced as well as extended to tailor capabilities to precise requirements

To complete our "wish list" we might as well add that the technology be used and supported by a multi-industry charitable foundation, created and maintained by an Open Source community, royalty-free and licensed to provide worldwide redistribution rights. Although these requirements may sound like a pipe dream, it's likely that Java application developers already have this incredible application framework installed. It's Eclipse.

Can Eclipse Be Used as an Application Framework?

The Eclipse Project FAQ say "The Eclipse Project is an Open Source software development project dedicated to providing a robust, full-featured, commercial-quality industry platform for the development of highly integrated tools." So by definition, Eclipse is an open platform for tools integration, not an IDE. The issue has been confused because a complete industrial-strength Java IDE is available in the Eclipse Platform in the form of plug-in components that extend Eclipse's basic framework facilities.

Eclipse provides the framework for combining disparate tools into a single integrated application with a seamless user interface. New tools are integrated into the Eclipse Platform and its user interface through plug-ins that extend Eclipse's facilities and provide new functionality. Eclipse plug-ins can also extend other plug-ins. When an Eclipse-based application initializes, it discovers and activates all of the plug-ins that have been configured for the environment. An Eclipse application is quite literally the sum of its parts since it's capable of performing any function that has been added to it by the plug-ins it currently contains.

Since being able to write and test such plug-ins is essential to the success of Eclipse, the Eclipse Platform is bundled with a plug-in development environment (PDE) and a set of Java development tools (JDT) to support it. Eclipse's developers clearly trusted the power of the frameworks they created. The entire development environment is just another set of tools integrated into the platform using

the standard plug-in techniques. The Eclipse Platform itself was itself created by developers using the Eclipse-based Java IDE (initially in beta form). And, since it's Open Source, anyone can inspect the code and understand in great detail exactly how the frameworks are supposed to be used.

It's this practice of packaging the development tools with the platform that causes some people to be confused about the nature of Eclipse. The JDT components are so effective that they're attractive to all Java developers, not just those writing plug-ins. On the surface, Eclipse appears to be just an excellent Java IDE. But instead of thinking about Eclipse simply as a Java IDE, try to think about it as a productivity application that happens to include a Java IDE built using the underlying Eclipse Platform as an application framework.

Eclipse Framework Features

Eclipse embodies an extensible design that maximizes its flexibility as an architectural platform. At its core, the Eclipse Platform contains an efficient implementation of the OSGi R4 core framework specification known as Equinox, which is used to bootstrap the application. Up from that, the Eclipse architecture defines sets of layered subsystems that allow it to be used as a framework for a portable application (or suite) that's not an IDE at all. And, since the frameworks are layered and coupled only at distinct architectural interfaces, an application can be built by combining only the

frameworks it needs, while eliminating those that it doesn't.

The following sections describe the primary Eclipse features that make it attractive as a general application framework.

Extensibility Model

Requirements change over time so developers often expend considerable effort designing applications so that they're flexible and extensible. Eclipse is built around a highly flexible and extensible plug-in model so any type of capability can be added to the platform. If an application can be thought of as a tool, or set of tools, it immediately becomes apparent that its functionality can be added to an Eclipse-based framework as a set of plug-ins just as Eclipse's native Java IDE capabilities have been.

Content Model

Eclipse provides a content model built around the concept of a workbench in which tools (capabilities) can be installed. The tools operate on resources organized into projects in the workspace. Projects contain a tree structure of resources, which are folders and files containing any type of content. The core platform provides a large number of extension points that allow the customization of all aspects of resource lifecycle management.

The hierarchical, categorized nature of the content model lends itself to many kinds of productivity applications with a bit of thought. For example, a simple e-mail client could be built on a workspace that contains a single project associated with the user's e-mail account. The user's project could contain folders for common functional e-mail elements such as inbox, outbox, and sent items. Each of these folders could contain the corresponding set of e-mail messages as project resources.

Native Widgets

The Eclipse platform contains a standard widget toolkit, SWT, implemented natively on all supported Eclipse platforms. SWT contains a large set of events, layout managers, and widgets. When a supported platform doesn't contain a native widget supported by Eclipse, such as a toolbar on Motif, an emulated widget for that platform is provided. SWT also interacts with native desktop features, such as drag-and-drop. SWT can also use OS-specific components, such as Windows Active/X controls, if such functionality is more desirable than full platform portability. So far, SWT has been proven on the Windows Win32 and Pocket PC, Photon, Motif, and GNU window

managers, covering deployment platforms from high-end workstations to embedded devices.

Although the Java language already contains two widget toolkits, AWT and Swing, the Eclipse group still chose to implement their own. The detailed reasons for this choice can be found in the Eclipse Overview white paper. However, to prove that this was the right decision, all one needs to do is compare the look-and-feel of a Swing or AWT application of your choice with that of Eclipse. Eclipse looks, feels, and responds like a native application on whatever platform it's running on.

User Interface Framework

To build a graphical interface, SWT can either be used directly or through JFace, the user interface framework of the Eclipse platform. JFace includes dialog, preference, progress reporting, and wizard frameworks as well as image and font registries that make user interface creation very straightforward.

The Eclipse platform supports a multi-window, MDI-like user interface presentation. On top of JFace and SWT the Eclipse workbench provides a framework for building perspectives, editors, and views that provide the structure for user interaction. Editors handle resource lifecycle interactions such as creating, editing, saving, and deleting. Views are used to provide supplementary information about an object with which the user is interacting. Examples include outline, pending tasks, and property views. A perspective is a stacked, tiled, or detached arrangement of views and editors. Only one perspective is visible in a window at a time but you can open multiple windows to view multiple perspectives simultaneously.

The Eclipse user interface framework is extensive, flexible, and powerful. And, even if it doesn't do everything you need, it can easily be extended in less time and with fewer resources than designing and building your own.

Update Manager

Historically one of the biggest problems associated with applications is the support cost incurred to package, distribute, maintain, and upgrade the application as new versions are released. This cost increases when a large and dispersed user community uses the application. With an offering's success and broad deployment, support after the sale can become time-consuming and expensive.

Component maintenance and upgrade facilities were part of the design of Eclipse from the beginning. To control ongoing cost and re-

move maintenance issues that could become barriers to project development and deployment, the Eclipse platform contains a flexible update manager. The update manager can be configured to initially install new components or updates to existing components from a remote server. As you release new versions of your application or add-on components, distribution can be as easy as packaging them using Eclipse facilities and putting them on your update server.

Help System

Every professional desktop application has a help system for end users and Eclipse is no different. However, Eclipse's help system isn't simply built from a static group of HTML files that document Eclipse. Rather, it's a framework for providing both searchable and context-sensitive help that's open to extension by documentation plug-ins. As a result, for any application built on Eclipse, everything's available for constructing, packaging, and shipping a complete, custom, context-sensitive help system without buying third-party tools.

Using Eclipse as an Application Framework

So starting with the underpinnings of a Java IDE as an application framework may at least sound possible, but why would anyone do it? Well, Eclipse satisfies the full function and facilities wish list mentioned earlier, while providing the program development environment for building the project as a series of Eclipse plug-ins. At the outset the frameworks provide an empty, featureless application that is architecturally sound, extensible for future enhancements, and can upgrade itself remotely.

The main question then becomes how much of Eclipse is required? Simply stated, an application can be built on the Eclipse framework by removing functionality that's not important and adding functionality that
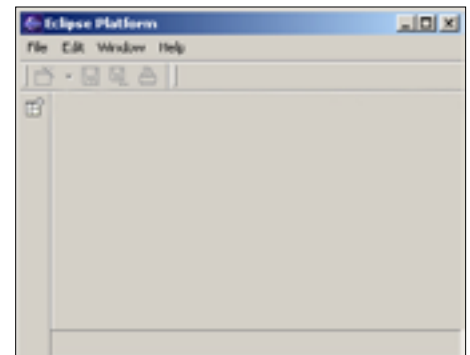

Figure 1: An "empty" Eclipse-based application

REALWORLD FLEX SEMINAR

www.flexseminar.com

Learn How to Build the Next Generation of Web Apps
from the Experts!

August 14, 2006

The Roosevelt Hotel
New York City

Adobe Flex 2 is a complete, powerful application development solution for creating and delivering cross-platform rich Internet applications (RIAs), within the enterprise and across the web. Not until now has there been a way for enterprise programmers and architects to work with existing tools of choice, familiar programming models and integration with existing systems and infrastructure. Multi-step processes, client-side processing, direct manipulation and data visualization are all key factors in the Flex solution.

The "Real-World Flex" One-Day Seminar will delve deep into the central workings of Flex so that the seminar delegates can integrate this timely new technology into their applications, creating powerful interactive content. Delegates will learn from the experts who not only created the product but also those who are using it to the massive benefit of their employers, their customers and the Web.

"Go Beyond AJAX with Flex."

The list of topics at the Real-World Flex Seminar includes:

✓ *The Flex Approach to RIA Development*
✓ *Bridging Flex and AJAX*
✓ *Integrating The SPRY Framework*
✓ *Using Flex Builder 2*
✓ *Flex for Java Developers*
✓ *ActionScript 3.0 Tips and Tricks*
✓ *How To Use ColdFusion with Flex*
✓ *Leveraging Flash*
✓ *Preparing for Adobe Apollo*
✓ *MXML Master Class*

Who Should Attend?

✓ *CEOs and CTOs*
✓ *Senior Architects*
✓ *Project Managers*
✓ *Web programmers*
✓ *Web designers*
✓ *Technology Evangelists*
✓ *User Interface Architects*
✓ *Consultants*
✓ *Anyone looking to stay in front of the latest Web technology!*

Early Bird:
(Hurry for Early Bird Pricing) $395*

Conference Price:
(Register Onsite) $495*

OFFER SUBJECT TO CHANGE WITHOUT NOTICE, PLEASE SEE WEBSITE FOR UP-TO-DATE PRICING

* Golden Pass access includes Breakfast, Lunch and Coffee Breaks, Conference T-Shirt, Collectible Lap-Top Bag and Complete On-Demand Archives of sessions in 7 DVDs!

is. The more challenging part is where to begin. The easiest cases are in the extremes. For example, when building a commercial IDE, like we do with MyEclipse Enterprise Workbench, it makes sense to start with the complete Eclipse Platform download, as well as a few other Eclipse projects, and build on top of them. At the other extreme, when building an application for an embedded device or any other environment where size constraints are paramount, then either Equinox or eRCP would make a more reasonable starting point. If the deployment target has a few more resources, but still don't require the vast majority of the platform's features then using the RCP (available from the platform download page) as the primary framework is likely the right starting point. With a little configuration on the base RCP you can quickly set up an "empty" application, as shown in Figure 1, and then concentrate on adding only what adds value, rather than infrastructure.

Once the starting platform has been determined, building an application is simply a matter of writing plug-ins to add features to the basic Eclipse framework and branding them appropriately for the intended audience. For example, a large application is typically written as multiple custom perspectives and supporting views using many plug-ins. Alternatively, to integrate a suite of small applications, perhaps each one can be a single perspective in its own plug-in. Along those lines, Eclipse can also be used as a portal to integrate all of a company's homegrown applications. The possibilities are truly endless. And, just to prove the point, here's a very wide assortment of Eclipse-based applications from all over the world.

GumTree is an Open Source graphical user interface framework for building scientific instrumentation consoles as shown in Figure 2.

EclipseTrader is an Open Source set of plug-ins for the Eclipse RCP dedicated to the building of an online stock trading system, featuring shares price watching, intra-day and historical charts with technical analysis indicators, level II/market-depth views, news watching, and integrated trading. The main view is depicted in Figure 3.

Azureus implements the BitTorrent client protocol through Eclipse RCP plug-ins and comes bundled with many invaluable features for both beginners and advanced BitTorrent users. Azureus is typically one of the most downloaded applications at SourceForge and interface looks native on any platform, thanks to SWT, as shown in Figure 4.

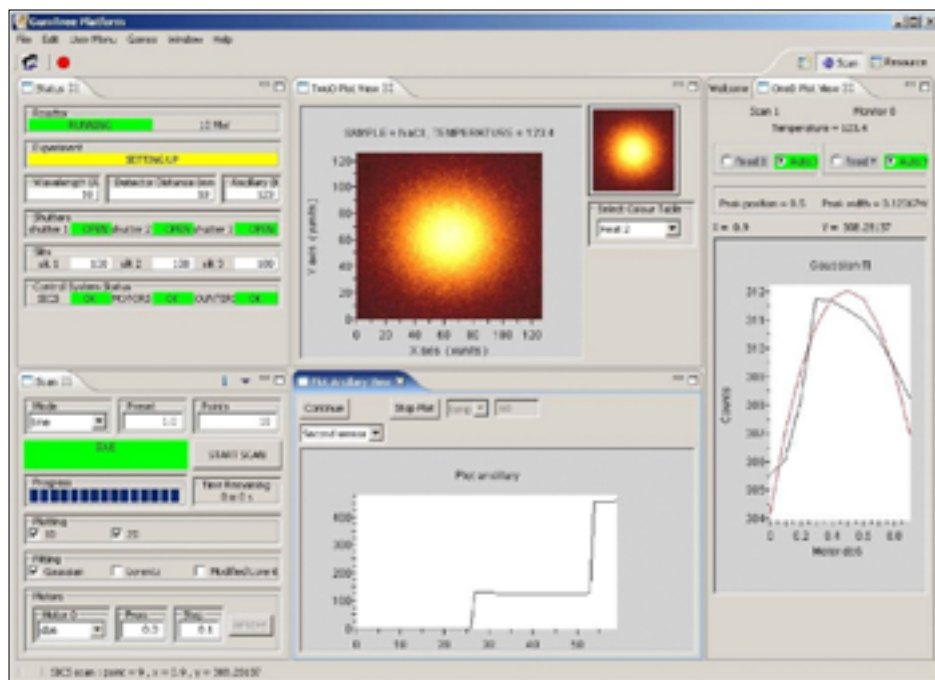Qanyon World Factbook application was


Figure 2: GumTree

Figure 3: Eclipse Trader

written to explore using Eclipse RCP in a distributed environment. Similar to the CIA World Factbook web site, the Qanyon World Factbook should display country information, albeit in a rich client environment, as shown in Figure 5.

Going Forward –
What's Next for Eclipse?

Eclipse is continuously evolving and will continue to grow both vertically further

into the software tools space and horizontally into completely new market segments. Interestingly, the growth into new industry verticals will be for the same reasons that Eclipse was formed in the first place. Although Eclipse was initially formed to build an integration platform for software tool providers, the separate availability of the RCP changes everything. Rather than being a platform exclusively for tool providers, Eclipse has become a general-purpose
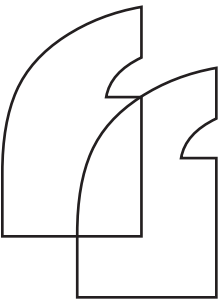
> Eclipse is constantly expanding, evolving, and surprising all of us. **So much so that it would have been impossible to envision where it has gone in its few years of existence**
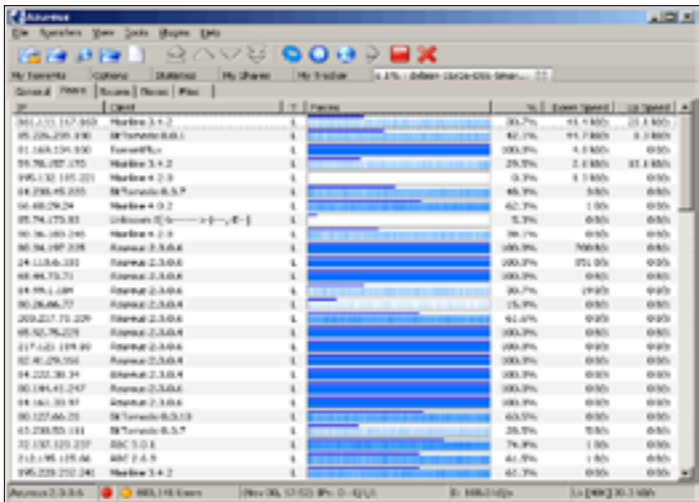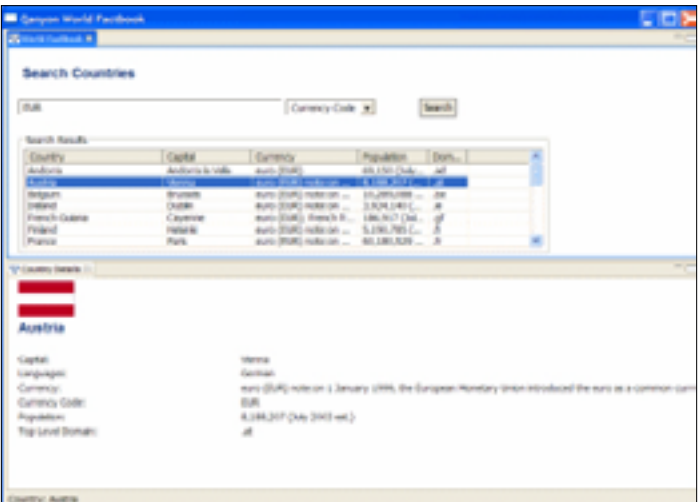

Figure 4: Azureus

Figure 5: Qanyon Wolrd Factbook

platform that has simply been leveraged initially in the software tools arena. With this seminal change, Eclipse will begin drawing participants from other verticals who want to cooperate in the same way that the current group of tool providers has. In the near future I expect to see interest in building infrastructure for productivity applications, reporting tools, security, process workflow, and business intelligence among others. Now that Eclipse is completely open and inclusive across the entire software industry, its membership and growth will explode in the coming years.

Another vehicle of Eclipse's future growth will likely come from completely outside the software industry. Consortia from such diverse industries as healthcare, automotive, and finance regularly set software platform and interoperability standards. However, without a portable, cross-platform implementation of the standards, each consortium member must independently construct its own, solely based on the industry specifications. This tremendous duplication of work is both expensive and error-prone. Collaborating on building a common set of specification-compliant infrastructure would universally cut costs   while insuring interoperability. But what competitors require before they can cooperate is a level playing field that benefits all of them equally. When they begin to research their options, they will find that Eclipse's maturity, extensibility, and royalty-free redistribution model is very attractive as the base for their collaborative development efforts.

Eclipse is constantly expanding, evolving, and surprising all of us. So much so that it would have been impossible to envision where it has gone in its first few years of existence. And, going forward, doing a reasonable job predicting what is next for Eclipse seems just as difficult. There's only one thing for certain; the future is arriving every day and no one really knows what it holds. Software visionary Alan Kay once said, "The best way to predict the future is to invent it." And, whatever the "next big thing" is, one thing is increasingly likely; it will be built on Eclipse. ⊙

Taking a Solutions Approach

Interview with Collax' Paula Hunter

interviewed by Mark R. Hinkle

Paula Hunter is an industry veteran who has an extensive background in Linux and open source. She was tapped in May of this year to lead the U.S. marketing efforts for Collax (www.collax.com), a Linux distribution aimed at addressing the needs of the small and medium business. Collax is set to expand on their European success in the U.S. this summer offering a solution that could rival more traditional offerings like Microsoft Small Business Server.

Paula Hunter
VP of U.S. marketing for Collax, is aiming to make a splash with Collax U.S. launch

About the Interviewer
Mark R. Hinkle is the vice president of strategy and corporate development for Emu Software as well as the editor-in-chief of **Enterprise OpenSource Magazine**. *Mark served on the Formation Board of The Desktop Linux Consortium and is also the author of the upcoming book* Linux Business Desktop Migration for Windows Users *from Charles River Press.*
mrhinkle@encoreopus.com

EOSM: First Paula – catch us up on you, where you've been, how you landed with Collax.
Paula Hunter: When I left UnitedLinux, I started working with the Open Source Development Labs (OSDL) – as you know, the leading advocate for Linux. I am very proud of the work done at OSDL to drive adoption of Linux in the enterprise. The time is right to drive Linux into a new market segment – small and medium-sized businesses – so Collax was a perfect transition for me. Collax was created by a strong engineering organization, has had phenomenal success in Europe to-date, and I believe will be very well received here in the U.S. I am also happy to be working again with our CTO, Boris Nalbach; I have great respect for his work at SUSE.

EOSM: What is Collax? What are you trying to do?
PH: We believe that Linux and open source software can offer significant cost-of-ownership savings to small and medium businesses when packaged and delivered properly. In developing the Collax Business Server, a top priority was to simplify Linux while retaining all of the benefits (speed of innovation, cost savings, scalability, etc.). We wanted to make it easy for the channel and the SMB end user to deploy Linux, without having to invest in incremental staff and training. By taking a solutions approach, we take the guesswork out of pulling together the necessary components to deploy effective security, networking, and communications systems. Collax offers an easy-to-deploy, multi-function solution that can handle high-level security services, complex routing, and networking and generic server tasks, which all take advantage of the latest innovations in open source software. We are delivering Linux and open source software with an easy-to-use graphical user interface for installation, updates, and maintenance that just lets business owners focus on getting their work done.

EOSM: Why now? Why do you think SMBs are ready to adopt open source?
PH: Over the past several years, the large enterprise has been the leading adopter of Linux in the IT market. During this early phase, there was confusion about open source software (OSS) and the various business models that emerged. We are now at a maturation point with Linux and OSS where ISVs and channel partners understand that there is tremendous opportunity, and that the technology and business models are hardened. Add to that the fact that the channel is always looking for new ways to differentiate and deliver value to its customers.

These two players (the channel, and ISVs) were the necessary ingredients for a successful move into the SMB market. This places Collax in an excellent position to capitalize on this new wave of opportunity.

As for the end users, they really don't care much about operating systems; they want best-of-breed solutions that can be managed cost effectively. They care that it's coming from a trusted partner and want something that is affordable, and secure with no (or little) downtime. Customers will rely on their reseller to make the best choices for them, and we believe that choice will be Collax.

EOSM: What about Red Hat/Novell, even Xandros? Can't someone else build this?
PH: Red Hat and Novell have a basic challenge in addressing the SMB market: they have already chosen the enterprise as the focus of their development. Large companies that have embraced Linux are looking for a

broad set of options and features, and have the IT resources to deploy complex solutions. It would be very difficult for a small business, or even their reseller, to deploy a Linux distribution that was designed for the enterprise market. The expertise does not exist today, and, frankly, the offerings are more than a small business needs. For a reseller to consider switching from an already complex Microsoft offering (SBS), it has to be simple. As for Xandros, I have a lot of respect for what they are doing on the desktop. The user interface is so important and is arguably the biggest hurdle for Linux to succeed on the desktop. That said, I don't believe VARs are looking for an SBS clone. Anyone who has installed several of the necessary components related to SBS would tell you that the experience is frustrating. Why would anyone want to emulate that? We want to focus on simplicity.

EOSM: Are you concerned about yet another distro? How will SMBs manage/integrate different distros?
PH: In one sense, we don't even view ourselves as a distribution. We are offering a turn-key server solution rather than a set of components (that the customer has to decide which ones to deploy). We are buffering the end user from what would be a very different experience for them if they had to touch a Linux distribution. Our graphical user interface is the key element that allows a small business to deploy Linux, without building up Linux expertise.

I think another server solution is essential to the success of Linux in the SMB space, for the reasons I stated (why existing distros have challenges in this segment). As for managing and integrating the server, the majority of deployments will be implemented by a trusted partner (VAR, systems integrator). That partner is likely to choose one distro, and our belief is that they will look for the solution that deploys cost-effectively and is easy to manage. Most channel partners want to make money on services, but they want to push the cost of delivering those services down.

EOSM: Since your offering is based on open source products, aren't you worried about competitors coming in since the barrier to entry seems very low?
PH: As many early entrants to the open source world have found, it takes a lot more than technology to deliver solutions to market. For one thing, you need a go-to-market strategy and the infrastructure to support it that requires investment in staff and systems. Our investors felt we had a viable strategy. Customers and resellers also want a strong support infrastructure, and a commitment to regular and timely updates. They want the best that open source has to offer, but they don't want to become open source community members. With Boris as our CTO, we know that he understands the open source development process, is a respected member of the development community, and can buffer the customer and VARs from having to navigate the wide range of projects that we monitor for best-of-breed components. There will be other entrants, but we have more than a year of development and deployment behind us. With over 8,000 installations worldwide, we have learned a lot and are gaining momentum.

EOSM: Why do you believe resellers will find Collax attractive? What is the value proposition for channel partners?
PH: Collax offers resellers the opportunity to tap into the rapidly growing Linux market without requiring a significant investment in Linux expertise. Our solutions were designed to be easy to deploy and maintain, thus reducing the cost of delivering service and support to the end customer. Resellers can reduce their dependence on one supplier (Microsoft) and meet their customers' ongoing infrastructure needs, and Collax solutions offer them a new line of business to differentiate themselves from their competition. ⭘

About Collax

Collax' goal is to make information technology so easy that in the end it's as clear and reliable as a hi-fi stereo system.

CFO
Bill Hite

CEO
Olaf Jacobi

CTO
Boris Nalbach

Under the motto "Collax – Simply Linux" the company offers, as the core of a computer network, a Linux-based server solution (with or without hardware) for departments and subsidiaries of small and medium-sized enterprises. The software solution turns a PC into a multifunction server combining all of the main administration, communication and security functions into a cheaper comparable products.

Collax's technology has several thousand installations on the market. The company currently has 30 people in its Munich and Freiburg offices. The Company plans to open its US headquarters in Boston Massachusetts in 2006.

All advantages of Linux, combined in one graphical user interface.
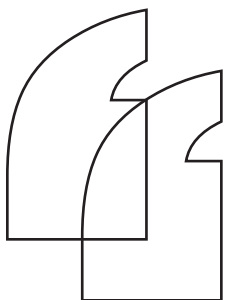
Resellers can reduce their dependence on one supplier (Microsoft) and meet their customers' ongoing infrastructure needs, **and Collax solutions offer them a new line of business**

Introducing the Malware Trend

Protecting your IT infrastructure

by David Perry

Malware is a generalized term to describe a collection of viruses, worms, spyware, and Trojans. The Microsoft Windows community is familiar with malware since the Windows industry is based on many independent software vendors that develop vertical applications. The complexity of a mixed operating environment presents an ideal opportunity for malware since security vulnerabilities are everywhere, and it's difficult for IT management to distribute patches as needed.

Over the past few years, major virus outbreaks have shut down businesses – from Bank of America to mom-and-pop shops. Recovery from an outbreak is so expensive that companies often have to reallocate their budget to pay for IT security expenses. Security is no longer a luxury of large businesses, but an infrastructure investment for business continuity.

Cross-platform infection recently became a reality, and there are now three types of viruses to keep you on the alert:

1. Open source applications have fewer resources to deal with security patch issues, and certain Linux viruses utilize open source vulnerabilities to attack.
   (http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=ELF_KAITEN.AM)

2. A new breed of viruses infect both Linux and Windows file systems. This is a common operating environment in education and research organizations.
   (http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=ELF_BI.A)

3. The major push by UNIX to replace Microsoft Office applications also attracts hackers. The XML_DUSTAR.A is the first virus to attack StarOffice/OpenOffice Suites on Windows 98, ME, NT, 2000, XP, and Server 2003; Linux; and Solaris.
   (http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=XML%5FDUSTAR%2EA&VSect=P)

About the Interviewer
David Perry is an authority on computer virus prevention with more than 25 years in the technical support and education field.

Introduce Enterprise IT Strategy for a Heterogeneous Environment

Although malware was perceived by the UNIX community as a Windows problem, the UNIX community is typically more tech savvy and proactive when dealing with security threats. Patch management is handled with great care since most UNIX servers are a platform for large databases and mission-critical systems.

Since its inception, Linux has evolved from a platform for the hobbyist community to an alternative to UNIX and Windows servers. Many enterprises, including Google, for example, are testing large deployments of Linux technology. Linux servers that were initially deployed as e-mail and Web servers have now migrated to replace Windows file servers. As a file server, a Linux server shares its file system with Windows clients through FTP and SAMBA. This sharing is usually on the departmental level or small business. The lack of supervision on those servers tends to make patch management more challenging, and the multi-purpose nature of Linux servers increases their exposure to outside attack.

For most enterprises that want to minimize operating costs, Linux offers a great return on investment and superior reliability. The IT strategy for the enterprise is to optimize each business process silo with department and business units taking new initiatives. The Linux server is the best candidate for a business initiative. The heterogeneous environment for the enterprise is a natural consequence of organic growth as Linux builds from its success in live deployment.

The AV Compliance Policy for Enterprise Customers and the Top Concern for Security Deployment

The enterprise not only needs security policies to protect its IT infrastructure, it needs to enforce security compliance. Since viruses account for 70% of all enterprise security problems, antiviral compliance is the top concern when deploying a security solution. The initial damage and cleanup

process of an antiviral problem is expensive since it is often necessary to dispatch IT personnel. The nature of viruses requires antivirus security and enforcement on every desktop and server. This is the best way to limit viruses from spreading when an outbreak occurs.

The global supply chain also introduces third-world countries to computers and links their systems to first and second world countries. The low-cost nature of Linux systems makes it the largest adoption among manufacturers and connecting to those systems is a necessity for global trade. The challenge, however, is to address unknown viral threats in the Linux platform just like Windows.

Conclusion

There are companies in the business world that utilize Linux as an infrastructure for its openness, reliability, and performance. Enterprises that have already adopted Linux servers need a reliable and integrated security solution to protect mixed environments with both Windows and Linux platforms. For complex Linux environments, system administrators need to consistently deploy patches to multiple distributions of Linux. This will

reduce the risks posed by constantly changing applications or sharing configurations.

A migrating enterprise needs a strategy that lowers the risk of migration from Windows to Linux, and Microsoft Office to Open Office. The intercompany and intracompany network application traffic needs to be filtered and secured for viral outbreak prevention. ⭕

The Trend Strategy and Solution for the Enterprise Environment

Trend Micro has a strategy to deal with this security problem using a centralized server to push out the latest malware patches for both platforms – Linux and Windows. Trend Micro Enterprise Protection Strategy offers multiple layers of products and services to protect mixed-platform environments, including complex distributions of Linux technology. Trend Micro handles Debian, Red Hat, Red Flag, and SuSE with its product release 2.5. This helps solve a problem that most global enterprises face today. While users in Asia, Europe, and North America tend to use different distributions, being able to tackle all various distributions can be a huge benefit if the outbreak is able to infiltrate an network segment with low protection coverage. Global IT managers no longer need to master multiple distributions in order to achieve 100% security patch coverage.

Whenever there is an outbreak, Trend Micro notifies its subscribers to ensure the day-zero protection. The push model of its security patches helps enterprise IT managers reduce their polling efforts by checking disparate vulnerability information sources. With limited resources and more demands on today's IT administrators, Trend Micro helps automate the security patch process, creating a relationship of trust and reliability.

Technalign Ships Frontier Version of Linux

(*Delta, CO*) – Technalign, Inc., has started shipments of TaFusion MEPIS Linux Frontier. Frontier is the next generation of TaFusion MEPIS Linux that is Ubuntu based. Version XI is still available from Partners and Technalign directly.

The company also announced that they would start development of the 64-bit version of the operating system to be followed by their Enterprise Server offering.

Although Frontier was scheduled for release in September, MEPIS development occurred faster than initially expected. Frontier is designed to be a simple to use system based on Ubuntu. The previous version, Version XI, was Debian based. A test drive is available from the Technalign download server located at www.tafusion.org, which is not installable but allows individuals to test and run the software off a live CD.

There are several versions of Frontier available which include the OEM, Simply and the PRO DVD. Each product is bundled with a CrossOver Office Standard trial. Other bundles include full CrossOver Standard and Professional. These bundles allow companies to run Microsoft Windows application. The base system comes with a 2 CD set instead of the single CD in Version XI. A product upgrade is also available for those wishing to move from the OEM or Simply.

www.technalign.com

Arkeia Software Announces New Partners in Asia-Pacific Region

(*Tokyo*) – Arkeia Software, a provider of data protection software for networked storage, announced its extended reach into the Asia-Pacific region with the signing of seven distributors tasked with delivering Arkeia solutions to China, Hong Kong, Japan, Korea, Philippines, Singapore, and Thailand.

The seven new distribution partners are responsible for sales and marketing activities in their regions, as well as providing support and other technical services to customers located throughout those geographical areas. The new distribution partners will build businesses based on the recently announced Arkeia Network Backup Version 5.5, featuring full Unicode character support.

The new distributors include StorArray Technologies in Shanghai, Guangzhou and Beijing, China; Computer Dynamics Corporation in Tokyo, Japan; Apposite Solutions Ltd. in Hong Kong; Valueline Systems & Solutions Corp. in Quezon City, Philippines; Resolvo Systems Pte. Ltd. in Singapore; BEC Solutions Inc. in Seoul, South Korea; and GrandLinux Solution Co., Ltd. in Bangkok, Thailand.

www.arkeia.com

IBM Extends Open Source Virtualization Solutions

(*Armonk, NY*) – Building on the company's commitment to providing clients with open solutions based on Linux and virtualization technologies, IBM has announced that its extensive portfolio of middleware and systems platforms will support Novell's new SUSE Linux Enterprise 10 offerings. The new SUSE Linux Enterprise offerings incorporate the open source Xen virtualization software to help businesses increase server utilization and lower management costs.

IBM will support Xen technology as part of the Virtualization Engine portfolio on the company's Intel and Opteron processor–based server and IBM BladeCenter systems. In addition, IBM has plans to support SUSE Linux Enterprise 10 from Novell across its entire IBM systems hardware line and IBM middleware portfolio and provide services support.

Xen is an open source virtualization software that allows multiple operating systems to run concurrently on the same physical server, allowing customers to consolidate their current workloads onto a single server. Recent tests by Novell shows customers may have the ability to increase server utilization by nearly 70 percent, as well as help increase the efficiency of managing and provisioning Linux systems.

www.ibm.com

Hyperic Releases Source Code for Hyperic HQ Management Platform

(*San Francisco*) – Hyperic Inc. has released the source code for Hyperic HQ, an open source platform for managing all commercial and open source IT assets, and also unveiled a new community portal to promote collaborative development of the application. Hyperic announced last month that it has moved to an open source license model after two years of deploying its previously commercial IT

management solution in over 100 companies including several Fortune 100 corporations.

The Hyperic HQ source code can be downloaded at no charge at both the Hyperic and SourceForge Websites under version 2 of the GNU General Public License (GPL V2).

In addition to offering code downloads and binaries, Hyperic's new community portal at www.hyperic.org offers a central resource for developers, ISVs, OEMs, system integrators, and others who want to participate in enhancing and extending the capabilities of the Hyperic platform. The portal will host forums for each Hyperic HQ component and provide a location for submitting and obtaining developer contributions ranging from new plug-ins to functional enhancements, localized versions and documentation.
www.hyperic.com

Fujitsu SSL Selects OpenLogic to Manage Open Source Stacks

(Broomfield, CO) – OpenLogic, Inc., a provider of software, stacks, and services that enable enterprises to deploy and manage customized open source environments, has announced that Fujitsu Social Science Laboratory Ltd. (Fujitsu SSL) has selected OpenLogic's solution to manage open source integration for Fujitsu SSL and its clients.

Fujitsu SSL has chosen the OpenLogic Enterprise technology platform to configure, integrate, and maintain open source solutions for its clients. Fujitsu SSL will use extensibility features in OpenLogic Enterprise to configure mixed stacks that include open source, commercial and proprietary software. In addition, OpenLogic will provide Fujitsu SSL with technical support services for several open source products.

Industry analyst firm Gartner has predicted that open source software will continue to grow in Asia, with 60 percent of large and mid-size government agencies using it in their core business processes by 2010, compared to less than 15 percent in late 2004.
www.ssl.fujitsu.com

JasperSoft Launches JasperForge.org

(San Francisco) – JasperSoft Corporation, a provider of open source business intelligence, has announced the launch of JasperForge.org, an online site that provides free access to collaborative tools to build, deploy, and extend open source business intelligence (BI) software. JasperForge, located at , also provides users with a central place to download BI software, learn configuration tips, and share BI deployment experiences with other people.

JasperForge.org provides developers with free access to a wide range of collaborative development tools including discussion forums, version control, wikis, a development dashboard, issue tracking, and automated project monitoring. IT managers can easily find and download open source BI software from a central location and read blogs from JasperSoft executives and other members of the open source BI community.
www.jasperforge.org

Empower Technologies and Texas Instruments Streamline Consumer Electronic Development

(Richmond, BC) – Simplifying the integration of Linux and DSP applications, Empower Technologies (Empower) and Texas Instruments Incorporated (TI) have announced the release of Empower's latest LinuxDA Embedded Operating System (LEOs 2.3) running Linux Kernel 2.6.12 along with TI's DSP/BIOS Link for TI's OMAP5912 dual-core digital signal processor (DSP) and ARM general purpose processor (GPP). The Linux Kernel 2.6.12 is available with the OMAP5912 and TI's latest software tools, such as DSP/BIOS Link, Code Composer Studio Integrated Development Environment and Reference Framework 6 (RF6) so developers can create consumer electronics products at reduced cost, risk and time-to-market.

Available in the LDK5912 Development Kit, LEOs for TI's OMAP5912 dual-core processor with Linux Kernel 2.6.12 is the latest Linux operating software released by Empower. A complete development solution, Empower's LDK5912 comes with a flexible hardware board based on TI's OMAP5912 dual-core processor and solid software platform, making the kit ideal for applications that require high-processing performance and long battery life in a feature-rich, small package. Developers can use the LDK5912 to compile, download and test applications within minutes of set-up using the latest LEOs Linux kernel and TI's software tools offering.
www.empowertechnologies.com
www.ti.com/3p

The Free Standards Group Unites Linux Printing Initiatives

(San Francisco) – The Free Standards Group (FSG), a nonprofit organization dedicated to developing and promoting open source software standards, has announced Linuxprinting.org, the de facto standard repository for printer drivers on Linux, is merging with the FSG's OpenPrinting workgroup and will be integrated and supported in the Linux Standard Base (LSB). The result will be easier and standardized printing functionality on Linux and an ease of support for Linux and printing vendors and makes the Free Standards Group the central organization for printing on Linux and open source Unix. With this announcement, the Free Standards Group unifies all standardization activities for the Linux meta-platform, including desktop, server, application interoperability, developer support, and printing.

Major printing vendors including Ricoh, HP, Epson, Kyocera, and Brother currently use Linuxprinting.org as their standard support vehicle for their Linux printer drivers. The system is used by the printer setup tools of most Linux distributions including Red Hat, Novell, Ubuntu, Debian, and Mandriva. The database currently lists 250 free software printer drivers and 2000 printers, with 10,000 people visiting the site for printing support everyday. The FSG is also announcing is has hired Till Kamppeter, the current manager of Linuxprinting.org, to oversee this project and other Linux printing initiatives.

The OpenPrinting workgroup has recently released two specifications that enhance the interaction of applications and printers and help to achieve the goal of a standardized print environment in *nix environments. The two specifications – Print Application Programming Interface (PAPI) Version 1.0 and Job Ticket Application Programming Interface (JTAPI) Version 1.0 – are available at http://www.openprinting.org.

Rich Internet Applications: AJAX,

A Look at the Eclipse Callisto Release

Providing a more transparent and predictable development cycle

by Chris Aniszczyk & Gunnar Wagenknecht

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.

About the Authors
Chris Aniszczyk is a software engineer at IBM Lotus focusing on OSGi Services. He is an open source enthusiast at heart, works on the Gentoo Linux (http://www.gentoo.org) distribution, and is a committer on the Eclipse Modeling Framework Technology (EMFT) project and Eclipse Communications Framework (ECF).
zx@us.ibm.com

Gunnar Wagenknecht is a software engineer at Truition focusing on AJAX and Eclipse RCP user interfaces and usability. An Eclipse friend since 2001, he started Planet Eclipse and is a committer on the Platform UI R21 presentation component.
gunnar@wagenknecht.org

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 (http://www.eclipse.org/platform) is the heart of Eclipse and has three main pieces:
- **Java Development Tools (JDT)** – http://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)** – http://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)
http://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)

http://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)

http://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)

http://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)

http://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)

http://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)

http://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
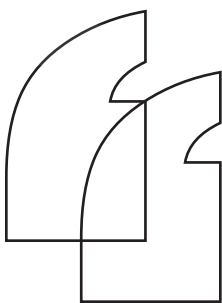
"From the user's perspective Callisto radically changes the way Eclipse **and the participating Eclipse projects get on the desktop**

WebTools Platform (WTP)
http://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)
http://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 (http://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 http://www.eclipse.org/callisto.
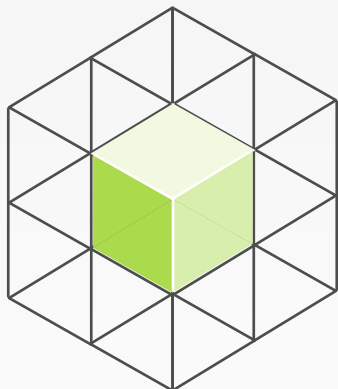
Open Management Consortium

Systems Management is Now Open

In May 2006 the Open Management Consortium was announced to help advance the promotion, adoption, development and integration of open source systems /network management software. The founding members of the consortium are Ayamon, Emu Software, Qlusters, Symbiot, Webmin, and Zenoss.

Specific objectives of the Open Management Consortium include:

- Create awareness of open source management tools in the market
- Provide education and resources to help end users make informed decisions regarding open source
- Establish conventions and standards that enable integration and interoperability
- Enable collaboration and coordination on common development projects
- Promote collaborative open source systems management solutions

Open source systems management replaces monolithic vendor lock-in with a modular approach. Pick what you need, customize it to your exact specifications and add to it as your needs change. Because open source products welcome contributions by users, partners and other third parties, they must be standards-based. This standards-based approach facilitates interoperability between open source solutions across the systems management life cycle.

AYAMON EmuSoftware Qlusters

symbiot security Webmin Zenoss
OPEN SOURCE ENTERPRISE MONITORING

www.open-management.org

I CAN'T GET MY DATA OUT OF OUR CRM SYSTEM.

CUSTOMIZING OUR CRM SYSTEM IS OFF-LIMITS.

OUR CRM SYSTEM IS SO INFLEXIBLE.

SugarCRM® gives you control over your CRM implementation that proprietary, closed source CRM restricts. Now you can fully manage your business critical data.

Start using the most powerful commercial open source CRM solution built on the modern LAMP architecture for managing sales, marketing, support and group collaboration activities today.

Get hooked on SugarCRM. Visit: **www.sugarcrm.com/eosm** or call **+1 408.454.6941**.

Commercial Open Source Customer Relationship Management

SUGARCRM®
COMMERCIAL OPEN SOURCE
www.sugarcrm.com • www.sugarforge.org
1.87SUGARCRM • +1.408.454.6941