



**Database:**  
64-Bit Linux Apps Need High-Quality  
64-bit Database Connectivity  
22 Mike Frost

**Object Database:**  
Mixing Data & Data Structures  
in an Object Database  
28 Rick Grehan



# ENTERPRISE OpenSource MAGAZINE

Visit us at [opensource.sys-con.com](http://opensource.sys-con.com)

THE LEADING MAGAZINE FOR ENTERPRISE AND IT MANAGEMENT

FEBRUARY/MARCH 2007 VOLUME 5 ISSUE 2

## A ROADMAP TO OPEN SOURCE ADOPTION



12 Ibrahim Haddad

### PLUS

- 3 **WELCOME TO OPENSVILLE**  
whurley (forward by Mark R. Hinkle)
- 4 **DEVELOPING AN APP USING THE  
ECLIPSE BIRT DESIGN ENGINE API**  
Jason Weathersby, Jane Tatchell, and Tom Bondur
- 18 **10 OPEN SOURCE  
PACKAGES YOUR ENTERPRISE  
SHOULD BE USING**  
Rod Cope
- 26 **AM I SEEING PYTHON EVERYWHERE?**  
Paul Nowak
- 32 **EXPERIMENTING WITH  
MONTAVISTA LINUX**  
Miroslaw Zakrzewski



PRESORTED  
STANDARD  
US POSTAGE  
PAID  
ST. CROIX PRESS

SEE PAGE 21  
End Annual  
ENTERPRISE 2007  
OPENSOURCE  
CONFERENCE+EXPO

SEE PAGE 25  
2007  
VIRTUALIZATION  
CONFERENCE+EXPO  
[www.virtualizationconference.com](http://www.virtualizationconference.com)

June 25-27, 2007  
New York City



# VERIO HOSTING IS LINUX WITH A LINEAGE.

- **Root Access:** Providing the control you need.
- **Advanced FairShare Technology:** Better resource management means better performance.
- **Support That's Actually Supportive:** Award-winning support provided by system administrators.

## Announcing Verio Linux® VPS.

At Verio, we have a long-standing commitment to open source, dating back to our roots in FreeBSD. Now, as the pioneer in virtual private server (VPS) technology and as a hosting provider backed by the financial resources of the world's largest telecommunications company, we bring something extra to Linux: reliability. To learn more, call 1-877-837-4654 or visit [www.verio.com/linuxlineage](http://www.verio.com/linuxlineage).

There is no substitute for the right foundation.

Verio and the Verio logo are trademarks and/or service marks of Verio Inc. in the United States and other countries. Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries. All other names are trademarks or registered marks of their respective owners. ©2007 Verio Inc. All rights reserved.

*Build on us.*  
**VERIO**  
An NTT Communications Company

## EDITORIAL BOARD

### Editor-in-Chief

Mark R. Hinkle mrhinkle@encoreopus.com

### Contributing Editor

Ibrahim Haddad ibrahim.haddad@sys-con.com

### Migration Editor

Jon Walker jwalker@sys-con.com

### Commercialization Editor

Paul Sterne sterner@sys-con.com

### Desktop Technology Editor

Tim Griffin tim@sys-con.com

### Review Editor

Matt Frye mattfrye@sys-con.com

### Editor

Philip Peake philip.peake@sys-con.com

### Contributing Editors

Kevin Larue kevinl@inspireinc.com

Christopher Negus cnegus@mwmt.net

### Contributor

Rob Jones rob@hotlinuxjobs.com

## INTERNATIONAL ADVISORY BOARD

Wim Coekaerts; Director of Linux Engineering, Oracle

Brian E. Ferguson; Partner, McDermott, Will & Emery

John Fowler; Executive VP, Network Systems Group, Sun Microsystems

Gaël Duval; Cofounder/Director of Communication, MandrakeSoft

Samuel J. Greenblatt; Sr VP and Chief Architect, Linux Tech. Group CA

Scott Handy; VP, Linux Strategy and Market Development, IBM

Bruce Perens; Perens, LLC

Stacey Quandt; Principal Analyst, Quandt Analytics

Thomas Reardon; VP and GM, Client Product Group Openwave Systems

John Weathersby; Executive Director, Open Source Software Institute

Ranajit Nevatia; Director of Linux Strategy, VERITAS

Andy Astor; co-founder and CEO, EnterpriseDB

## EDITORIAL

### Executive Editor

Nancy Valentine nancy@sys-con.com

### Research Editor

Bahadır Karuv, PhD bahadir@sys-con.com

## OFFICES

### SYS-CON MEDIA

577 Chestnut Ridge Rd. • Woodcliff Lake, NJ 07677

Telephone: 201 802-3000 • Fax: 201 782-9600

Enterprise Open Source Magazine (ISSN #PENDING)

is published monthly (12 times a year)

by SYS-CON Publications, Inc.

Postmaster send address changes to:

### ENTERPRISE OPEN SOURCE MAGAZINE

SYS-CON MEDIA • 577 Chestnut Ridge Rd. • Woodcliff Lake, NJ 07677

Copyright © 2007 by SYS-CON Publications, Inc.

All rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopy or any information, storage and retrieval system, without written permission. For promotional reprints, contact reprint coordinator. SYS-CON Publications, Inc., reserves the right to revise, republish and authorize its readers to use the articles submitted for publication. All brand and product names used on these pages are trade names, service marks, or trademarks of their respective companies.

### WorldWide Newsstand Distribution

Curtis Circulation Company, New Milford, NJ

### FOR LIST RENTAL INFORMATION:

Kevin Collopy: 845 731-2684, kevin.collopy@edithroman.com

Frank Cipolla: 845 731-3832, frank.cipolla@epostdirect.com

# Welcome to Opensville



By whurley (Foreward by Mark Hinkle)

I had originally written an editorial for this month's issue titled, "Is Commercialization Killing Open Source?" Then I read William Hurley's blog (<http://talk.bmc.com/blogs/blog-whurley/whurley/opensville>). William or as his friends call him, "whurley," is the chief architect of open source strategy at BMC. He gets to the heart of an issue that is being brought to light as a greater number of businesses adopt open source business models: "As a greater commercial influence exerts itself in the *open source community*, will these companies run roughshod on those early pioneers who have demonstrated the effectiveness of the open source model?"

Presently there are four multi-billion dollar companies that have a very significant open source element to their businesses: IBM, Sun, Novell, and Red Hat (you could make an argument for HP and Dell too if you like). In addition to these juggernauts, there is growing investment in open source models. In the first quarter of 2007 the following companies raised approximately \$100 million to fund businesses that directly rely on open source software or services.

- Avidence ([www.avidence.com](http://www.avidence.com)) – \$5 million, Series A
- Adaptive Planning ([www.adaptiveplanning.com](http://www.adaptiveplanning.com)) – \$7.5 Million, Series C
- Black Duck Software – ([www.blackduck.com](http://www.blackduck.com)) – \$12 million, Series C
- Fonality ([www.fonality.com](http://www.fonality.com)) – \$7 million, Series C
- Groundwork Open Source ([www.groundworkopensource.com](http://www.groundworkopensource.com)) – \$12.5 million, Series C
- Greenplum ([www.greenplum.com](http://www.greenplum.com)) – \$19 million Equity and Debt offering
- Penguin Computing ([www.penguincomputing.com](http://www.penguincomputing.com)) – \$9 million, Series B
- PostPath ([www.postpath.com](http://www.postpath.com)) – \$15 million, Series C
- rPath ([www.rpath.com](http://www.rpath.com)) – \$9.1 million, Series B
- Terascale ([www.terascale.com](http://www.terascale.com)) – \$3 million, Series A

As money flows into an industry that was once largely dismissed as subversive by proprietary software vendors, new entrants are claiming their "open source" status as an advantage over proprietary vendors.

Rather than my usual rant, I thought I would let someone who ironically works for a company that has a proprietary software heritage make a plea for better citizenship in the open source community:

*Nestled between Proprietary and Freedomberg, Opensville is a utopia. Everyone who lives in the adjacent cities spends their free time in Opensville. The parks are beautiful, the shopping is amazing, and the nights are pure Vegas. Sounds like a great place, huh? One problem: no one actually wants to live there. No one wants to pay the taxes or put in the effort it takes to keep the city running. Welcome to Opensville, population zero.*

*Wit or truth? Why, a bit of both, of course. There are too many entities taking advantage of open source technology without giving back. Some are literally pillaging the community that butters their bread. How long before we all suffer the effects? If major project contributors were to stop work, how would that affect the industry as a whole?*

—continued on page 15

### About the Author

Mark R. Hinkle, editor-in-chief of Enterprise Open Source Magazine, is the vice president, Community and Business Development at Zenoss Inc. He serves as a founder of the Open Management consortium and is the author "Windows to Linux Business Desktop Migration" (Charles River Media).

[mrhinkle@encoreopus.com](mailto:mrhinkle@encoreopus.com)

# Developing an Application Using the Eclipse BIRT Design Engine API

## Creating a customized report design application

by Jason Weathersby, Jane Tatchell, and Tom Bondur

**T**his article is the second in a series on developing an application using Eclipse BIRT Engine APIs. It focuses on developing an application using the Eclipse BIRT Design Engine API. The last article focused on the Eclipse BIRT Report Engine API.



The Eclipse Business Intelligence Reporting Tool (BIRT) is a set of plug-in extensions that enable a developer to add reporting functionality to an application. BIRT provides a Design Engine API that a developer can use to create a customized report design application. The `org.eclipse.birt.report.engine.api` package contains a set of interfaces and implementation classes that supports integrating the design-time part of BIRT into a reporting application.

### Programming with a Report Design

A reporting application typically generates a report from a report design. In this type of reporting application, you develop a report design and include the design along with the application at deployment. Any changes to the generated report depend on the values of report parameters and the data from the data set. To access the report design, the application uses an `IReportRunnable` object.

Sometimes business logic requires changes to the report design before generating the report. You can make some changes by using parameters and scripting. Other changes can only occur by modifying the report design itself.

A reporting application can make changes to the report design and the Report Object Model (ROM) elements that make up the design. To access the structure of the report design, the application obtains a `ReportDesignHandle` object from the design engine. To access the design engine, an application must first instantiate a report engine, as in any other reporting application.

The `ReportDesignHandle` object provides access to all properties of the report design and to the elements that the report design

contains. The model API provides handle classes to access all ROM elements. For example, a `GridHandle` object provides access to a grid element in the report design. All ROM element handles, including the report design handle, inherit from `DesignElementHandle`. Report items inherit from `ReportElementHandle` and `ReportItemHandle`.

After making changes to a report design or its elements, the application can write the result to a stream or a file. The report engine can then open an `IReportRunnable` object on the resulting design and generate a report.

An application typically accesses the items in a report design to do one of the following tasks:

- Modify an existing report design programmatically to change the contents and appearance of the report output. An application can modify page characteristics, grids, tables, and other report items in the design, the data source, and the data set that extracts data from a data source.
- Build a report design and generate report output entirely in an application without using BIRT Report Designer.

A reporting application can access and modify the structures in a template or a library file like the structures in a report design. The techniques described in the rest of this article are applicable to these files as well as to report designs.

The functionality of a template is identical to a report design. For this reason, the `ReportDesignHandle` class provides access to a template. The `LibraryHandle` class provides access to a library. Both these classes derive from the `ModuleHandle` class, which provides the fields and methods for the common functionality, such as accessing elements in the file.

The package that contains the classes and interfaces to work with the items in a report design, library, or template is `org.eclipse.birt.report.model.api`.

### About the Authors

Jason Weathersby is the BIRT evangelist at Actuate Corporation. Jane Tatchell and Tom Bondur are content development managers in the Developer Communications Group of Actuate engineering. The authors are members of the extended BIRT development team at Actuate and have backgrounds in both computer science and technical writing. Collectively, they have many years experience in technical consulting, training, writing, and publishing about reporting, business intelligence tools, and database technologies.

[jweathersby@actuate.com](mailto:jweathersby@actuate.com)

[jtatchell@actuate.com](mailto:jtatchell@actuate.com)

[tbondur@actuate.com](mailto:tbondur@actuate.com)

## BIRT Model API Capabilities

A report developer can write an application that creates and modifies a report design programmatically. The BIRT model API has the same capabilities as BIRT Report Designer. For example, the following list shows some of the ways in which you can use the BIRT model API to manipulate a report design programmatically:

### Modify a report item in a report design:

- Format a report item, change the font, font color, fill color, format, alignment, or size. Modify the expression or other property of a report item.
- Change the data set bound to a table or list.

### Add a report item to a report design:

- Add a simple report item such as a data item, label, or image.
- Set the value to display in the new report item, such as the expression of a data item or the text in a label item.
- Create a complex item such as a grid, table, or list.
- Add other items into a grid, table, or list.

### Change the structure of a report design:

- Add or delete a group or column in a table or list.
- Add a report parameter.

### Modify non-visual elements in a report design:

- Specify a data source for a data set.
- Set a design property such as a report title, author, wallpaper, or comment.
- Set a page property such as height, width, or margins.

## Opening a Report Design Programmatically for Editing

To access a report design and its contents, the application must instantiate a report engine then use a ReportDesignHandle object. You instantiate a ReportDesignHandle by calling a method on another class such as the model class, SessionHandle, or the report engine interface, IReportRunnable.

The SessionHandle object manages the state of all open report designs. Use a SessionHandle to open, close, and create report designs, and set global properties, such as the locale and the units of measure for report elements. The SessionHandle can open a report design from a file or a stream. Create the session handle only once. BIRT supports only a single SessionHandle for a user of a reporting application.

### Configuring the Design Engine To Access a Design Handle

The DesignEngine class provides access to all the functionality of the ROM in the same way that the ReportEngine class provides

access to report generation functionality. To create a DesignEngine object, you first create a DesignConfig object to contain configuration settings for the design engine. The DesignConfig object sets up custom access to resources and custom configuration variables for scripting. Instantiate a DesignEngine object with the DesignConfig object as an argument to the constructor.

Create the SessionHandle object by calling the method, newSessionHandle() on the DesignHandle object. To open the report design, call the method, openDesign(), on the SessionHandle object. This method takes the name of the report design as an argument and instantiates a ReportDesignHandle object.

### Using an IReportRunnable Object To Access a Design Handle

You can also open a report design from an IReportRunnable object by using the getDesignHandle() method. The ReportDesignHandle object provides access to the design opened by the report engine. Changes to the report design do not affect the IReportRunnable object. To generate a report from the changed report design, you must reopen the design as an IReportRunnable object.

### How To Open a Report Design for Editing

The code sample in Listing 1 creates a DesignEngine object that it uses to create a SessionHandle object. The code then uses the SessionHandle object to open a report design.

## Using a Report Item in a Report Design

A report item is a visual element in the report design. Typically, a report developer adds a report item to the design in the BIRT Report Designer by dragging an item from the palette to the layout editor. Sometimes you have to change the properties of certain report items in the design before running the report. An application uses methods on the ReportDesignHandle class to access a report item either by name or from a list of items in a slot in a container report item.

A slot is a logical component of a report item. For example, a table element has five slots: Header, Detail, Footer, Groups, and Columns. In turn, each of these slots can have further slots. Each slot has zero or more members of the appropriate report item type. For example, the Header, Detail, and Footer slots all contain elements of the RowHandle type. RowHandle has a Cell slot that contains all the cells in the row. For a visual representation of the slots in an individual report item, see the Outline view in BIRT Report Designer.

### President & CEO

Fuat Kircaali fuat@sys-con.com

### Group Publisher

Jeremy Geelan jeremy@sys-con.com

## ADVERTISING

### Senior VP, Sales & Marketing

Carmen Gonzalez carmen@sys-con.com

### Advertising Sales Director

Megan Mussa megan@sys-con.com

### Associate Sales Manager

Corinna Melcon corinna@sys-con.com

## EVENTS

### Events Manager

Lauren Orsi lauren@sys-con.com

Sharmonique Shade sharmonique@sys-con.com

## PRODUCTION

### Art Director

Alex Botero alex@sys-con.com

### Associate Art Directors

Abraham Addo abraham@sys-con.com

Louis F. Cuffari louis@sys-con.com

Tami Lima tami@sys-con.com

## CUSTOMER RELATIONS

### Circulation Service Coordinator

Edna Earle Russell edna@sys-con.com

Alicia Nolan alicia@sys-con.com

## SYS-CON.COM

### Information Systems Consultant

Robert Diamond robert@sys-con.com

### Web Designers

Stephen Kilmurray stephen@sys-con.com

Richard Walter richard@sys-con.com

## ACCOUNTING

### Financial Analyst

Joan LaRose joan@sys-con.com

### Accounts Payable

Betty White betty@sys-con.com

## SUBSCRIPTIONS

TOLL FREE 888-303-5282

201-802-3012

subscribe@sys-con.com

For subscriptions and requests for bulk orders, please send your letters to Subscription Department

Cover Price: \$5.99/issue

Domestic: \$49.99/yr (12 issues)

Canada/Mexico: \$79.99/yr

all other countries \$99.99/yr

(U.S. Banks or Money Orders)

Back issues: \$12 U.S. \$15 all others

### Accessing a Report Item by Name

To make a report item accessible by name, the item must have a name. A report developer can set the name in BIRT Report Designer or programmatically by using the item's `setName()` method. To find a report item by name, use the `findElement()` method. This method returns a `DesignElementHandle` object. All report items derive from this class.

### Accessing a Report Item by Iterating through a Slot

To access a report item through the report design's structure, the application first gets the slot handle of the report body by calling the `getBody()` method. This slot handle holds the top-level report items in the report design. For example, consider a simple report structure that has three top-level items: a grid containing header information, a table containing data, and a label that displays a report footer. Figure 1 shows its outline view in BIRT Report Designer.

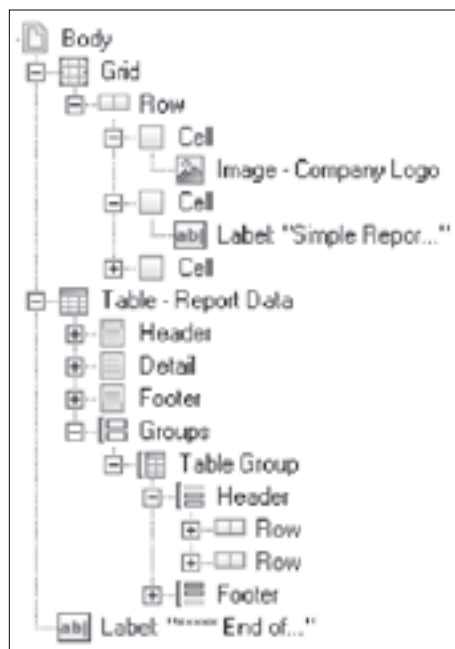


Figure 1: Slots in a report design

To access the top-level items in this report design, you iterate over the contents of the body slot handle. These contents all derive from `DesignElementHandle`. To access the iterator for a slot handle, call `SlotHandle`. iterator. Each call to `Iterator.getNext()` returns a report item. Alternatively, to access a report item at a known slot index, call `SlotHandle.get()`. The slot index number is zero-based. The `ReportDesignHandle` class also provides finder methods, which can access an item or other report element by name.

### Examining a Report Item Programmatically

To examine a report item, check the class of the report item, cast the object to its actual class, then call methods appropriate to that class. For example, the class of a label element handle is `LabelHandle`. To get the text that the label displays, call `LabelHandle.getText()`.

Some report items, such as a label or a text element, are simple items. Other items, such as a grid or a table element, are structured items. You can access properties for the whole of a structured item in the same way as for a simple item.

You can also iterate over the contents of the structured item. For example, use this technique to determine the contents of a cell in a table. To access the contents of a structured item, you call a method to retrieve the slot handle for rows or columns. For example, to access the `RowHandle` objects that make up a table element's footer, call `TableHandle.getFooter()`. Table and list elements also have a slot for groups. Like the body slot handle, the slot handles for the contents of structured report items can contain zero, one, or multiple elements.

### Accessing the Properties of a Report Item

To provide information about report items, each class has getter methods specific to the report item type. For example, an image element handle, `ImageHandle`, has the `getURI()` method. This method returns the URI of an image referenced by a URL or file path. The `DesignElementHandle` class and other ancestor classes in the hierarchy also provide generic getter methods, such as `getName()`.

Some properties of a report item are simple properties, with types that are Java types or type wrapper classes. An example of this type of property is the name property, which is a `String` object. Some of these properties, like name, have arbitrary values.

Other simple properties have restricted values from a set of BIRT String constants. The interface, `DesignChoiceConstants` in the `org.eclipse.birt.report.model.api.elements` package, defines these constants. For example, the image source property of an image element can have only one of the values, `IMAGE_REF_TYPE_EMBED`, `IMAGE_REF_TYPE_EXPR`, `IMAGE_REF_TYPE_FILE`, `IMAGE_REF_TYPE_NONE`, or `IMAGE_REF_TYPE_URL`.

Other properties are complex properties and the getter method returns a handle object. For example, the `DesignElementHandle.getStyle()` method returns a `StyleHandle` object and `ReportItemHandle.getWidth()` returns a `DimensionHandle` object.

The handle classes provide access to complex properties of a report item, as described later in this article. These classes provide getter methods for related properties. For example, `StyleHandle` classes provide access to font and background colors.

### How To Access a Report Item by Name

The code sample in Listing 2 finds an image item by name, checks its type then examines its URI. The variable, `design`, is a `ReportDesignHandle` object.

### How To Use the Report Structure To Access a Report Item

The code sample in Listing 3 finds an image item in a grid, checks its type, then examines its URI. Use this technique for generic code to navigate a report design structure or if you need to find an item that doesn't have a name. The variable, `design`, is a `ReportDesignHandle` object.

## Modifying a Report Item in a Report Design Programmatically

To set the simple properties of report items, each class has setter methods specific to the report item type. For example, an image element handle, `ImageHandle`, has the `setURI()` method. This method sets the URI of an image referenced by the URL or file path. The `DesignElementHandle` class and other ancestor classes in the hierarchy also provide generic setter methods, such as `setName()`. Setter methods throw exceptions, such as `NameException`, `SemanticException`, and `StyleException`.

To set the attributes of a complex property, such as a style, you must call methods on a handle object, as described later in this article. These classes provide setter methods for related properties. For example, `StyleHandle` classes provide access to style properties, such as font and background color.

Changes that you make to items in the report design don't affect the design file until you save the design to disk or to a stream. After saving the design, get an `IReportRunnable` handle for the modified design to generate a report.

### How To Change a Simple Property of a Report Item

The code sample in Listing 4 uses a method on `LabelHandle` to change the text in a label. The variable, `design`, is a `ReportDesignHandle` object. This sample accesses the label by name. You can also access a report item by navigating the report structure.



## YOU ALWAYS HAD THE BRAINS. IT WAS THE TECHNOLOGY THAT WAS A LITTLE SCATTERED.

### The HP BladeSystem c-Class with Insight Control Management.

The intuitive HP BladeSystem c-Class thinks just like you do — letting you monitor your infrastructure while helping to analyze your future needs. First, HP's OnBoard Administrator gives you out-of-the-box setup and configuration combined with power, cooling and enclosure management. After that, the Insight Control software steps in to let you control the rest of your environment, locally or remotely. And thanks to the integrated Insight Display — our unique LCD screen — you can interact right at the source to manage, deploy or troubleshoot.

Simply plug in the HP ProLiant BL460c server blade, featuring Dual-Core Intel® Xeon® Processors, and you'll get faster performance and versatility to support 32- and 64-bit computing environments. Use the HP BladeSystem c-Class for your business and you'll experience greater control over your time and resources.



Experience the HP BladeSystem and download the IDC White Paper "Enabling Technologies for Blade Management."



Click [YouAlwaysHadIt.com/brains7](http://YouAlwaysHadIt.com/brains7)  
Call 1-866-625-4085  
Visit your local reseller



Dual-Core is a new technology designed to improve performance of multithreaded software products and hardware-aware multitasking operating systems and may require appropriate operating system software for full benefit; check with software provider to determine suitability; not all customers or software applications will necessarily benefit from use of this technology. Requires a separately purchased 64-bit operating system and 64-bit software products to take advantage of the 64-bit processing capabilities of the Dual-Core Intel Xeon Processor. Given the wide range of software applications available, performance of a system including a 64-bit operating system will vary. Intel's numbering is not a measurement of higher performance. Intel, the Intel Logo, Xeon and Xeon Inside are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries. The information contained herein is subject to change without notice. ©2006 Hewlett-Packard Development Company, L.P.

## Accessing and Setting Complex Properties

Complex properties use BIRT handle objects to access data structures. For example, a `DimensionHandle` object provides access to size and position properties, such as the absolute value and the units of the width of a report item.

Some String properties on a handle object, such as the font style and text alignment on a style handle, have restricted values defined by constants in the interface, `DesignChoiceConstants` in the `org.eclipse.birt.report.model.api.elements` package. For example, a font-style property can have only one of the values, `FONT_STYLE_ITALIC`, `FONT_STYLE_NORMAL`, and `FONT_STYLE_OBLIQUE`.

## Using a Property Handle

To access complex properties, you use getter methods on the report item. For example, to access the width of a report item, call the method `ReportItemHandle.getWidth()`. This method returns a `DimensionHandle` object. To work with complex properties, you use getter and setter methods on the handle object. For example, to get and set the size of a dimension, you use `DimensionHandle.getMeasure()` and `DimensionHandle.setAbsolute()`, respectively.

When you set a value on a complex property, the change to the handle object affects the report item straightaway. You don't call an additional setter method on the report item itself.

## Using Styles on a Report Item

The `StyleHandle` class provides access to many fundamental properties of a report item, such as margin size, text alignment, background color, borders, font, and so on. `StyleHandle` provides a full set of getter methods for each style property. For simple properties, `StyleHandle` provides setter methods. To modify complex properties, you use setter methods on the property handle object, not on the style handle itself.

A report item can use two styles: a private style and a shared style. The handle classes for these styles are `PrivateStyleHandle` and `SharedStyleHandle`, respectively. Both classes derive from `StyleHandle`.

A private style contains the settings that the report developer chose in the property editor when designing the report. Shared styles appear in the Outline view in BIRT Report Designer. You use shared styles to apply the same appearance to multiple items in a report design. Changes to a shared style affect all report items that use the style. Style settings in a private style override the settings in a shared style.

## How To Change a Complex Property of a Report Item

The code sample in Listing 5 shows how to use `PrivateStyleHandle` and `ColorHandle` objects to change the background color of a label. The variable, `design`, is a `ReportDesignHandle` object. This sample accesses the label by name. You can also access a report item by navigating the report structure.

## Adding a Report Item to a Report Design Programmatically

A reporting application can use a simple report design or a template to create more complex designs. The application can add extra report items to the design's structure based on external conditions. For example, based on the user name of the user requesting the generation of a report, you can add extra information to the report for that category of user. You use the same techniques to add content to a new design if you create a design entirely with the API.

The class that creates new elements, such as report items, in a report design is `ElementFactory`. This class provides methods of the form, `newXXX()`, where `XXX` is the report item or element to create. The method `newElement()` is a generic method that creates an element of any type. To access the element factory, call the `ReportDesign.getElementFactory()` method.

You can place new report items at the top level of the report design, directly in the Body slot, in containers such as a cell in a table or grid, or on the master page. You can add a simple item, such as a label, or complex items, such as a table with contents in its cells. Wherever you add a new report item, the location is a slot, such as the body slot of the report design or a cell slot in a row in a table. To add a report item to a slot, you use one of the `SlotHandle.add()` methods. The method has two signatures that support adding the report item to the end of a slot, or to a particular position in a slot.

Table and list elements are container items that iterate over the rows that a data set provides. For these report items to access the data rows, you must bind them to a data set. The table or list element provides data rows to the report items that it contains. For this reason, you usually bind only the container item to a data set, as described later in this article.

## How To Add a Grid Item and Label Item to a Report Design

The code sample in Listing 6 creates a grid item then adds a label item to one of the cells in the grid. An application can create any other report item in a similar manner. The variable, `design`, is a `ReportDesignHandle` object.

## Accessing a Data Source and Data Set with the API

This section will explain how to use ROM elements that aren't report items. To use other ROM elements, such as the libraries that the report design uses, you employ similar techniques.

You access the report design's data sources and data sets from methods on the `ReportDesignHandle` instance in a way similar to other report elements. The model classes that define a data source and data set are `DataSourceHandle` and `DataSetHandle`, respectively. A data set provides a report item such as a table with data from a data source. For a report item to access the data set, use the `setDataSet()` method.

You can use a finder method on the report design handle to access a data source or data set by name. The finder methods are `findDataSource()` and `findDataSet()`, respectively.

Alternatively, to access all the data sources or data sets, you can use a getter method that returns a slot handle. The getter methods are `getDataSources()` and `getDataSets()`, respectively. To access the individual data sources or data sets in a slot handle, you iterate over the contents of the slot handle in the same way as for any other slot handle.

## Data Source Classes

`DataSourceHandle` is a subclass of `ReportElementHandle`. You get and set report item properties for a data source in the same way as for any other report element. `DataSourceHandle` also provides methods to access the scripting methods of the data source.

The two subclasses of `DataSourceHandle`, `OdaDataSourceHandle` and `ScriptDataSourceHandle`, provide the functionality for the two families of BIRT data sources. For more information about ODA data sources, see the Javadoc for the ODA API in the Open Data Access (ODA) 3.0.0 API Reference. The scripting methods for a scripted data source fully define the data source.

## Data Set Classes

`DataSetHandle` is a subclass of `ReportElementHandle`. You get and set properties for a data set in the same way as for any other report element. `DataSetHandle` also provides methods to access properties specific to a data set, such as the data source, the data set fields, and the scripting methods of the data set.

The two subclasses of `DataSetHandle`, `OdaDataSetHandle` and `ScriptDataSetHandle`, provide the functionality for the two families of BIRT data sets. For more information about ODA data sets, see the Javadoc for the ODA API.

## OPEN POSSIBILITIES

**May 8–11, 2007**

The Moscone Center, San Francisco, CA

JavaOne Pavilion: May 8–10, 2007

[java.sun.com/javaone](http://java.sun.com/javaone)



### > **JAVA™ TECHNOLOGY IS NOW OPEN—AND SO ARE THE POSSIBILITIES**

The 2007 JavaOne<sup>SM</sup> conference has expanded and is definitely one conference you won't want to miss. With the decision to open source Java<sup>™</sup> technology, 2007 marks a major milestone for the Java platform. Whether your passion is scripting languages, open source, SOA, Web 2.0, mashups, or the core Java platform, this is a conference that has something for almost all technology developers.

#### LEARN MORE ABOUT\*:

##### > Scripting

(JavaScript<sup>™</sup> Programming Language, PHP, Ruby on Rails, Python, and More)

##### > Open Source and Community Development

##### > Integration and Service-Oriented Development

##### > Web 2.0 Development

##### > AJAX

##### > Java Technology and the Core Java Platforms (EE/SE/ME)

##### > Compatibility and Interoperability

##### > Business Management

**SAVE \$100\*\***  
**Register Today!**

Please use priority code: J7PA3E0S

\* Content subject to change.  
\*\* Offer not available on-site.

Attend the JavaOne conference, and you will have many opportunities over the course of four days to network with like-minded developers; attend in-depth technical sessions; engage with your peers in Hands-on Labs and BOFs; and experience general sessions featuring speakers from Intel Corporation, Motorola, Oracle, and Sun Microsystems. Meet face-to-face with leading technology companies, and test-drive the latest tools and technologies shaping the industry.

#### PLATINUM COSPONSORS



**MOTODEV**  
The Motorola developer network

**ORACLE**

#### GOLD COSPONSORS



**NAVTEQ**

**NOKIA**

#### SILVER COSPONSORS

**INTERSYSTEMS**

**PARASOFT**  
We make software work.

**TERRACOTTA**

## Code Listings

**Listing 1: Opening a report design for editing**

```
// Create a design engine configuration object.
DesignConfig dConfig = new DesignConfig( );
DesignEngine dEngine = new DesignEngine( dConfig );
// Create a session handle, using the system locale.
SessionHandle session = dEngine.newSessionHandle( null );
// Create a handle for an existing report design.
String name = "./SimpleReport.rptdesign";
ReportDesignHandle design = null;
try {
    design = session.openDesign( name );
} catch (Exception e) {
    System.err.println
        ( "Report " + name + " not opened!\nReason is " +
          e.toString( ) );
    return null;
}
```

**Listing 2: Finding a report item with a given name**

```
DesignElementHandle logoImage = design.findElement( "Company Logo" );
// Check for the existence of the report item.
if ( logoImage == null ) {
    return null;
}
// Check that the report item has the expected class.
if ( !( logoImage instanceof ImageHandle ) ) {
    return null;
}
// Retrieve the URI of the image.
String imageURI = ( (ImageHandle ) logoImage ).getURI( );
return imageURI;
```

**Listing 3: Navigating the report structure to access a report item**

```
// Instantiate a slot handle and iterator for the body slot.
SlotHandle shBody = design.getBody( );
Iterator slotIterator = shBody.iterator( )
// To retrieve top-level report items, iterate over the body.
while ( slotIterator.hasNext( ) ) {
    Object shContents = slotIterator.next( );
    // To get the contents of the top-level report items,
    // instantiate slot handles.
    if ( shContents instanceof GridHandle ) {
        GridHandle grid = ( GridHandle ) shContents;
        SlotHandle grRows = grid.getRows( );
        Iterator rowIterator = grRows.iterator( );
        while ( rowIterator.hasNext( ) ) {
            // Get RowHandle objects.
            Object rowSlotContents = rowIterator.next( );
            // To find the image element, iterate over the grid.
            SlotHandle cellSlot =
                ( ( RowHandle ) rowSlotContents ).getCells( );
            Iterator cellIterator = cellSlot.iterator( );
            while ( cellIterator.hasNext( ) ) {
                // Get a CellHandle object.
                Object cellSlotContents = cellIterator.next( );
                SlotHandle cellContentSlot =
                    ( ( CellHandle ) cellSlotContents ).getContent( );
                Iterator cellContentIterator =
                    cellContentSlot.iterator( );
                while ( cellContentIterator.hasNext( ) ) {
                    // Get a DesignElementHandle object.
                    Object cellContents =
                        cellContentIterator.next( );
                    // Check that the element is an image.
                    if ( cellContents instanceof ImageHandle ) {
                        String imageSource = ( ( ImageHandle )
                            cellContents ).getSource( );
                        // Check that the image has a URI.
                        if ( ( imageSource.equals(
                            IMAGE_REF_TYPE_URL ) ||
                            ( imageSource.equals(
                                IMAGE_REF_TYPE_FILE ) ) ) ) {
                            // Retrieve the URI of the image.
                            String imageURI = ( ( ImageHandle )
                                cellContents ).getURI( );
                        }
                    }
                }
            }
        }
    }
}
```

**Listing 4: Changing the text property of a label report item**

```
// Access the label by name.
LabelHandle headerLabel = ( LabelHandle ) design.findElement( "Header Label" );
try {
    headerLabel.setText( "Updated " + headerLabel.getText( ) );
} catch ( Exception e ) {
    // Handle the exception
}
```

**Listing 5: Changing a complex property of a report item**

```
// Access the label by name.
LabelHandle headerLabel = ( LabelHandle ) design.findElement( "Header Label" );
try {
    // To prepare to change a style property, get a StyleHandle.
    StyleHandle labelStyle = headerLabel.getPrivateStyle();
    // Update the background color.
    ColorHandle bgColor = labelStyle.getBackgroundColor();
    bgColor.setRGB( 0xFF8888 );
} catch ( Exception e ) {
    // Handle any exception
}
```

**Listing 6: Adding a container item to the Body slot**

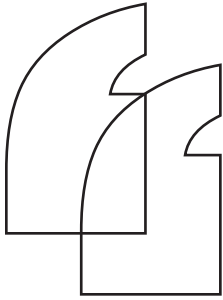
```
// Instantiate an element factory.
ElementFactory factory = design.getElementFactory( );
try {
    // Create a grid element with 2 columns and 1 row.
    GridHandle grid = factory.newGridItem( "New grid", 2, 1 );
    // Set a simple property on the grid, the width.
    grid.setWidth( "50%" );
    // Create a new label and set its properties.
    LabelHandle label = factory.newLabel( "Hello Label" );
    label.setText( "Hello, world!" );
    // Get the first row of the grid.
    RowHandle row = ( RowHandle ) grid.getRows( ).get( 0 );
    // Add the label to the second cell in the row.
    CellHandle cell = ( CellHandle ) row.getCells( ).get( 1 );
    cell.getContent( ).add( label );
    // Get the Body slot. Add the grid to the end of the slot.
    design.getBody( ).add( grid );
} catch ( Exception e ) {
    // Handle any exception
}
```

**Listing 7: Modifying a data set**

```
// Find the data set by name.
DataSetHandle ds = design.findDataSet( "Customers" );
// Find the data source by name.
DataSourceHandle dso = design.findDataSource( "EuropeSales" );
// Check for the existence of the data set and data source.
if ( dso == null ) || ( ds == null )
{
    System.err.println( "EuropeSales or Customers not found" );
    return;
}
// Change the data source of the data set.
try
{
    ds.setDataSource( dso );
} catch ( SemanticException e1 ) {
    e1.printStackTrace( );
}
```

**Listing 8: Binding a data set to a report item**

```
// Find the table by name.
TableHandle table = ( TableHandle ) design.findElement( "Report Data" );
// Find the data set by name.
DataSetHandle ds = design.findDataSet( "EuropeanCustomers" );
// Check for the existence of the table and the data set.
if ( table == null ) || ( ds == null ) {
    System.err.println( "Incorrect report structure" );
    return;
}
// Change the data set for the table.
try {
    table.setDataSet( ds );
} catch (Exception e) {
    System.err.println( "Could not set data set for table" );
}
```



# The Eclipse Business Intelligence Reporting Tool (BIRT) is a set of plug-in extensions that enable a developer to add reporting functionality to an application. **BIRT provides a Design Engine API that a developer can use to create a customized report design application**

## *Using a Data Set Programmatically*

Typically, a reporting application uses data sets and data sources already defined in the report design. You can use the data set's `setDataSource()` method to change the data source of a data set. For example, based on the name of the user of the reporting application, you can report on the sales database for a particular geographical region, such as Europe or North America.

## *Changing the Properties of a Data Set*

Changing the properties of a data set means considering the impact on the report design. If you change the data source of a data set, the type of data source must be appropriate for the type of data set. You must also be certain that the new data source can provide the same fields as the original data source.

## *How To Change the Data Source for a Data Set*

The code sample in Listing 7 shows how to check for a particular data source and data set in a report design then changes the data source for the data set. The code finds the data source and data set by name.

Alternatively, use the `getDataSets()` and `getDataSources()` methods. Then use the technique for iterating over the contents of a slot handle. The variable, `design`, is a `ReportDesignHandle` object.

## *Changing the Data Set Binding of a Report Item*

You can also use the report item's `setDataSet()` method to set or change the data set used by a report item. If you change the data set used by a report item, you must ensure that the contents of the report item access only data bindings that are supplied by the new data set. If necessary, you must change the references to data bindings in data elements, text elements, and scripting methods. If the data bindings in the old data set don't match the names or data types of the fields that the new data set provides, you must correct the data bindings before you generate a report from the modified report design. Use the `ReportItemHandle` method, `columnBindingsIterator()` to iterate over the column bindings that the report

item uses. The items in the list are `ComputedColumnHandle` types. This class provides methods to access the name, expression, and data type of the column binding.

To access the data set column and expression that a data item uses, call the methods, `getResultSetColumn()` and `getResultSetExpression()`. You can compare the data type and name with the result set columns that the data set returns.

## *How To Bind a Data Set to a Table*

The code sample in Listing 8 shows you how to check for a particular data set in a report design then changes the data set for a table. The code finds the table and data set by name. Alternatively use slot handles to navigate the report design structure. The variable, `design`, is a `ReportDesignHandle` object.

## *Saving a Report Design Programmatically*

After making changes to an existing report design or creating a new report design, you can choose to save the design for purposes of archiving or future use. To overwrite an existing report design to which the application has made changes, use the `ReportDesignHandle` `save()` method. To save a new report design or to keep the original report design after making changes, use the `ReportDesignHandle` `saveAs()` method.

Alternatively, if you don't need to save the

changes to the report design, use the `ReportDesignHandle` `serialize()` method. This method returns an output stream. The report engine can generate a report by opening a stream as an input stream.

If you don't need to make any further changes to the report design, use the `ReportDesignHandle` `close()` method to close the report design.

## *How To Save a Report Design*

The code below saves the open report design. The variable, `design`, is a `ReportDesignHandle` object:

```
design.saveAs("sample.rptdesign");
design.close();
```

## *Creating a Report Design Programmatically*

You can build a report design and generate the report output in an application without using BIRT Report Designer. You use the `createDesign()` method on the session handle class, `SessionHandle`, to create a report design. You use the other model classes to create its contents.

## *How To Create a New Report Design*

The following code creates a report design:

```
SessionHandle session = DesignEngine.newSession( null );
ReportDesignHandle design = session.createDesign( );
```



## About the book

This article is an excerpt from the book *Integrating and Extending BIRT* by Jason Weathersby, Don French, Tom Bondur, Jane Tatchell, and Iana Chatalbasheva and published by Addison-Wesley. The book is the second volume in a two-book series about business intelligence and reporting technology. The book introduces programmers to BIRT architecture and the reporting framework. It shows programmers how to build and deploy customized reports using scripting and BIRT APIs. It also describes how to use key extension points to create a customized report item, a rendering extension for generating output other than HTML or PDF, and an Open Data Access (ODA) driver for a new data source.



# Is the Open Source Development Model Right for Your Organization?

## A roadmap to open source adoption

by Ibrahim Haddad

**T**he open source development model has unique characteristics that make it in some instances a superior model for developing software compared to the traditional software engineering cascade model. As with other practices, the open source development model had its advantages and inconveniences. Will adopting the open source development model improve the way your corporate developers work and produce software? What are the best practices from the open source development model that we can use in a corporate environment?



The open source software development model has a different process and set of values than traditional proprietary software development model. The traditional software development process consists of six activities illustrated in Figure 1: collecting and analyzing requirements, designing a solution approach, developing the code, testing, deploying, and maintaining. After each step is finished, the process proceeds to the next step.

The open source development model has key differences compared to the traditional model of developing software (collect requirements, design, implement, test, release, and maintain).

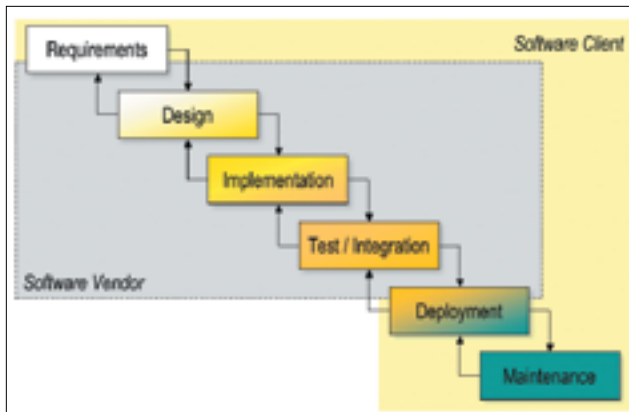
The open source development model, illustrated in Figure 2, starts with the idea for a new project, a new functionality or capability for an existing open source software component. The next step is to provide a design for the implementation and then a prototype of the capability and translate it from an idea into running software. At the moment the software runs, it's released as a development release, even though it may contain known and unknown bugs. This follows the spirit of release early and release often.

The software will be tested by the community, which discusses the software through mailing lists and discussion boards and provide feedback, bug reports, and fixes through the project mailing list. The feedback is recorded and taken into consideration by project members and maintainers to improve the implementation and then a new development release will be available. This cycle repeats as

often as needed until project members feel the implementation is stable enough. When the implementation is released as stable, the development cycle continues with the development release (also called the development tree) until a newer stable release is available.

Some of the unique characteristics of the open source development model include:

- **Bottom up development:** Project members who do the most work get the most say when it comes to making design and implementation decisions. Those who do the most work get the most say. Relationships between developers are very important.
- **"Release early, release often":** Don't wait to have a fully working version to make the code public. This release philosophy allows for peer review, where all members of the community can comment and offer suggestions and bug fixes. It also allows for small incremental changes that are easier to understand and test. Open source projects tend to make a release available early to be used by the user community and then update the release as the software is modified. This practice is described as "release early, release often." The open source community believes that this practice leads to higher-quality software because of peer review and the large base of users who are using and testing the software, accessing the source code, reporting bugs, and contributing fixes. A side benefit of having many people looking at the code is that the code is reviewed for adherence to coding style; fragile or inflexible code can be improved because of these reviews.



**Figure 1:** The cascade model of traditional software engineering

(SOURCE: BILL WEINBERG, OPEN SOURCE DEVELOPMENT LABS, 2006)

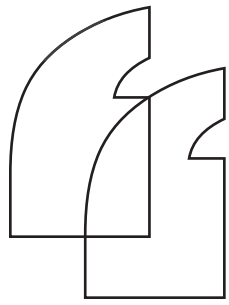
- **Peer review:** Members of the open source project review the code, provide comments and feedback to improve the quality and functionality, and test to catch bugs and provide enhancements as early as possible in the development cycle. The result is high-quality code.
- **Small incremental changes:** In open source project development, additional features are often small and non-intrusive and for good reason:
  - It's easier to understand small patches and code changes than big changes in the code or big architectural redesigns.
  - The small changes are important because they help focus the testing phase, which is cyclical and ongoing with every increment of the software.
  - A small change is less like to have unintended consequences.
- **Features that ignore security concerns are flagged:** The open source community takes security very seriously and any development or capability that jeopardizes the security of the software is flagged and not included in the software until the security concern is dealt with.
- **Continuous quality improvement:** This is due to the extensive peer review and quick bug fixes.
- **Test projects:** In many cases, test projects are created for large open source projects to create test suites and automate testing.
- **End-user involvement in the entire process:** In Figure 2, we notice that the users are involved in all phases of development in the open source model.

(SOURCE: BILL WEINBERG, OPEN SOURCE DEVELOPMENT LABS, 2006)

## Communication

Open source developers primarily communicate with each other using mailing lists. In the table below, we illustrate some slight differences concerning communication in an open source project compared to a corporate project.

Open Source	Corporate
<ul style="list-style-type: none"> <li>• Open Source developers are distributed across the world</li> <li>• No face-to-face meetings</li> <li>• No conference calls</li> </ul>	<ul style="list-style-type: none"> <li>• Depending on the size of the company, developers can be in different geographies</li> <li>• Weekly or bi-weekly project reviews to track progress, lead by project managers</li> <li>• High reliance on conference calls and face-to-face meetings</li> </ul>
<ul style="list-style-type: none"> <li>• E-mail is very important as the primary mean of communication between open source project members</li> <li>• Discussions happen on open mailing lists</li> </ul>	<ul style="list-style-type: none"> <li>• E-mail is important</li> <li>• Discussions are mostly face to face and in conference calls</li> <li>• A lot of one-to-one e-mails between project members</li> </ul>
<ul style="list-style-type: none"> <li>• Many open source projects use chat for quick developer and user discussions</li> </ul>	<ul style="list-style-type: none"> <li>• The use of chat software among corporate developers is growing as a cheap way to communicate versus travel for face to face meetings</li> </ul>



# Many companies are adopting some of the practices open source development model has special chara

## Project Hierarchy

Open source projects are organized differently than corporate projects. In the table below, we illustrate some key differences between open source projects and corporate projects focusing on project organization and hierarchy.

Open Source	Corporate
<ul style="list-style-type: none"> <li>• Open source development teams primarily work together in a decentralized fashion with little hierarchy</li> <li>• Hierarchy is loose and flexible</li> <li>• Those who make the most contributions have the most say about the project</li> </ul>	<ul style="list-style-type: none"> <li>• Well structured with defined roles for the project manager, project architect, senior developer, etc.</li> </ul>
<ul style="list-style-type: none"> <li>• There are no formal requirements for joining and no formal rules for participating</li> <li>• The lack of formality doesn't mean that there are no standards for participating or behaving</li> <li>• There are strong unwritten rules that govern all community interactions</li> <li>• Community members are expected to interact respectfully, make reasoned arguments about why a particular course of action is right, and above all, contribute to the community</li> </ul>	<ul style="list-style-type: none"> <li>• There are formal processes to follow when an individual wants to work on a new project</li> <li>• Individuals follow and respect company rules and regulations, and are expected to contribute to the success of the project</li> </ul>
<ul style="list-style-type: none"> <li>• Bottom-up development approach where decisions and power is as close to the bottom as possible (i.e., developers who write the code have a say in the direction of the project)</li> </ul>	<ul style="list-style-type: none"> <li>• Top-down development approach where project management makes the decisions and pushes it down to the implementers</li> </ul>
<ul style="list-style-type: none"> <li>• Meritocracy drives advancement and acceptance</li> <li>• As developers prove their competence and their contributions prove to be valuable to the project, they become more influential</li> </ul>	<ul style="list-style-type: none"> <li>• Corporate adopts specific criteria as part of its performance management</li> </ul>
<ul style="list-style-type: none"> <li>• Open source project members work on a project when, and as much, as they feel like it</li> <li>• Open source project members work on a project until they get bored and loose interest in the project</li> </ul>	<ul style="list-style-type: none"> <li>• Members of a project are fully dedicated to it and must dedicate all their time to the project</li> <li>• Must respect project deadlines and deliverable schedules</li> <li>• Can't stop working on a project without management approval</li> </ul>
<ul style="list-style-type: none"> <li>• Quality levels are often negotiable since the first goal is to provide a working prototype/proof-of-concept, but after several cycles the quality improves tremendously</li> </ul>	<ul style="list-style-type: none"> <li>• Quality is very important and often specific quality goals are request by customers</li> </ul>
<ul style="list-style-type: none"> <li>• The project leader is usually the person who originated the project or the person with the most technical competence and contributions working on the project.</li> <li>• The project leader manages the project by consensus, leading by example</li> <li>• The project leader is responsible for developing a common understanding of what functionally the upcoming release will contain, encourage new developers to join the project, help developers select a portion of the project to work on, and solve any conflicts that arise between team members</li> </ul>	<ul style="list-style-type: none"> <li>• The project leader is usually the manager assigned to the project by management</li> <li>• The project leader is responsible for project requirements, communicating them, assigning developers portion of the work, and resolving conflicts</li> </ul>

# of the open source development; the characteristics that make for faster development, faster testing, higher innovation, peer review, total openness, and transparency

## Cultural Differences

Working with the open source community is very different from the traditional corporate development environment and has a different process and set of values from the traditional proprietary development model. In the table below, we illustrate some key cultural differences between an open source development environment and a corporate development environment.

Open Source	Corporate
<ul style="list-style-type: none"><li>• Open source developers work on what they find interesting and bring tremendous energy to the project they contribute to</li></ul>	<ul style="list-style-type: none"><li>• Corporate developers work on projects they are assigned to</li></ul>
<ul style="list-style-type: none"><li>• Open source developers are usually volunteers who donate their time to open source projects that benefit the community as a whole</li></ul>	<ul style="list-style-type: none"><li>• Corporate developers are paid to work on company projects</li></ul>
<ul style="list-style-type: none"><li>• Motivation for improving and developing a given piece of software is unpredictable. It might vanish or decrease depending on the interest in this piece of software. Release schedules are uncertain.</li></ul>	<ul style="list-style-type: none"><li>• Motivation for improving and developing a given piece of software is driven by customer demand</li></ul>
<ul style="list-style-type: none"><li>• Open source developers work on features of interest to them. As such, they don't work to meet specific deadlines, but work as long as they're interested in the project.</li></ul>	<ul style="list-style-type: none"><li>• Corporate developers are paid by their companies to devote their time to the projects they're assigned to</li></ul>
<ul style="list-style-type: none"><li>• Open source developers work in the open with full transparency and extensive peer review of their code</li><li>• All code developed for the project will be viewed, reviewed, and enhanced</li></ul>	<ul style="list-style-type: none"><li>• Development typically takes place in a product group that is often closed and not available to others in the company for cultural reasons and little peer review outside the group that did the development</li></ul>
<ul style="list-style-type: none"><li>• Open source developers welcome code contributions written by other developers</li></ul>	<ul style="list-style-type: none"><li>• Corporate often suffer from the "not invented here" syndrome in accepting code written by others</li><li>• Moving from writing propriety code to contributing source code to open source or using code developed by others is a new way of doing things.</li><li>• Many corporations are developing open source policies and procedures, and creating open source training for their employees</li></ul>
<ul style="list-style-type: none"><li>• Open source developers are famous for their code reuse practices and try to avoid doing something twice if it can be automated</li></ul>	<ul style="list-style-type: none"><li>• Corporates are encouraging code reuse among their developers in an effort to produce reusable software to help cut their costs</li></ul>
<ul style="list-style-type: none"><li>• Open source developers maintain a source code tree that is open and available for all to see and access. They follow the release early release often practice that gives a good estimate of the progress and helps catch bugs early</li></ul>	<ul style="list-style-type: none"><li>• Corporate developers follow strict rules when it comes to accessing source code trees and offering stable releases</li></ul>

## from the editor


—continued from page 3

Let's use the monitoring segment of systems management as an example. Several "open source contributors" simply download code from popular projects and then "build" their software, service, or company on top of it. These contributors often refer to "improvements" they've made. Where are these improvements? Why weren't they contributed to the community from which they took the code? Open source should be about working together for common benefit.

Nagios is one of the most popular monitoring projects in open source, and one of the most abused. There are countless projects, products, and services predicated on the Nagios code base – some symbiotic, others non-contributing parasites. What separates legitimate use from outright exploitation? Where would you draw the line? Should violators be black-listed by the community?

To me, open means that everyone can participate on a level playing field. As a community we have to take the good with the bad, but I cringe when I see a project taking more than its fair share of punishment. How will the community address this problem? Should there be a ratings system? A sort of mooch-o-meter to rank companies and projects that use open source? Would that subjective hierarchy help or hurt the community? How would it be regulated?

The community has to answer some of these questions if open source is to continue to flourish. Everyone who leads, participates in, or utilizes an open source project should realize they have a personal interest in protecting it from abuse. Keeping the pirates honest will take effort, but the repercussions of apathy will affect us all in the future. Besides, tales of the pirate hunters are often more exciting than the tales of the pirates themselves.

You can read this and other open source musings by whurley at <http://talk.bmc.com/blogs/blog-whurley/whurley/>. 

### About the Author

Whurley (William Hurley) is the chief architect of Open Source Strategy at BMC Software, Inc. He is the chairman of the Open Management Consortium, a non-profit organization advancing the adoption, development, and integration of open source systems management. Named as an IBM Master Inventor, whurley has received numerous awards including an IBM Pervasive Computing Award and Apple Computer Design Award.

## The Benefits of Adopting Open Source Working Methods


There are several open source development practices that corporates can benefit from adopting in their development environment that can improve code quality, communication, effectiveness, and performance.

- **Using open development methods “à la Sourceforge”**
  - Open source code tree: Make source code available to others to review and offer feedback and suggest improvements (peer review). Inside a company, this lets teams work across organizational lines and lets others add value to the software. Different users tickle different bugs, leading to higher quality. The practice of incrementally adding functions allows for better testing and better chances of capturing bugs. Cooperation is good and benefits all.
  - Open mailing list used for all project-related discussions.
  - Bug tracking systems.
  - Technical support tracking systems.
  - Patch tracking systems.
  - Feature request tracking systems.
- **Fast development cycle with small incremental changes**
  - Adopt the “release early and release often” practice.
  - Go through the cycle several times.
  - Apply small incremental changes in the release to make it easier to understand and test.
  - Faster development builds.
  - Shorter time-to-market.
- **Pay special attention to quality and security**
- **Encourage reuse**
  - Promote and encourage company developers to use open source software and tools in their development environment where it might meet their needs
  - Include open source software in products based on a set of criteria such as technical merits, time-to-market advantage, and avoiding vendor lock-in.
  - Code reuse improves efficiency and increases cost savings.
- **Build reusable software components**
  - Don't keep reinventing the wheel and don't act superior. If someone has already implemented the capability or feature you need, use it, and build on top of it.
  - When you develop from scratch, keep reuse in mind, and develop code in modules that can be used by others and by you for other situations without much modification.
- **Respect and follow community coding style**
  - The open source community follows a strict coding style to make it easier to understand the code, review it, and revise it quickly.
- **Flag problems early and review with the team**
  - Hiding problems or bugs until you come up with a solution isn't encouraged.
  - It's advisable to report bugs or problems when they turn up; the community will help you come up with a workaround or propose and help implement a better solution.
  - Openness and honesty is key.
- **Foster innovation**
  - New ideas have a better chance if engineers can review the source code and experiment with and build proof-of-concept code and test different methods.

Recommended Practices	Description
<b>Increase team communication</b>	Using mailing lists, chat software, wikis
<b>End-user feedback</b>	Involve the end user to get feedback as you proceed
<b>Peer review</b>	Encourage peer review and provide an environment that welcomes feedback and suggestions
<b>Release early and often</b>	Adopt the “release early release often” development practice for the many benefits it offers as compared to the traditional release model, and follow the model of continuous integration and automated test environments
<b>Transparency</b>	Adopt transparency and openness by using open source code trees, bug tracking database, and mailing lists that are open to the whole company.
<b>Good code design</b>	Build a minimal code base and add all the functions and capabilities as separate modules to encourage reuse and ensure easier testing.

## Conclusion

The open source development model has proved to be a very successful model with hundreds of open source projects that can be used as a success story. This development model has special characteristics that allow faster development, faster testing, higher innovation, peer review, total openness and transparency. In this article we reviewed the open source development model and compared it to the traditional corporate development model. Many companies are adopting some of the practices of the open source development model for the advantages it offers.

Will these practices be right for your company? You be the judge! 

### About the Author

*Ibrahim Haddad is Director of Embedded & Open Source Technology at Motorola where he is responsible for defining and developing the requirements for Motorola Software Group's open source initiatives. Prior to Motorola, Dr. Haddad managed the Carrier Grade Linux and Mobile Linux initiatives at the Open Source Development Lab (now the Linux Foundation), which included promoting the development and adoption of Linux and open source software in the communications industry. He is co-author of two books on Red Hat Linux and Fedora, a contributing editor of the Linux Journal, Linux Planet, and Enterprise Open Source Magazine, and a featured speaker and panelist at industry conferences. He received his BSc and MSc degrees in Computer Science from the Lebanese American University, and his PhD in Computer Science from Concordia University in Montreal, Canada*



**IT360° is a powerful multi-sector experience.**

Get the IT360° advantage – a single source of knowledge, realistic strategies, and tools to help you solve the critical burning issues today, paving the road for efficiency and productivity tomorrow. [www.it360.ca](http://www.it360.ca).

**Conference:** April 30 – May 2, 2007

**Trade Show:** May 1 – May 2, 2007

**Metro Toronto Convention Centre – TORONTO, CANADA**

IT360° Conference and Expo is an ITWorld Expo event produced by ITWorld Canada the trusted name in Information Technology Resource Media.

[www.it360.ca](http://www.it360.ca)

PRODUCED BY:



# 10 Open Source Packages Your Enterprise Should Be Using

## Open source alternatives besides Linux

by Rod Cope

**L**inux tends to take center stage when it comes to support and other services for enterprise open source users. However, there are literally thousands of other solid open source packages available that perform a wide variety of functions. Unfortunately, there's a real lack of information about the options and considerations for selecting open source that not only meets the functional and technical requirements of specific tasks, but has the support and backing that enterprises need to manage risk. As a result, with enterprise developers lost in a sea of open source options, it can be a daunting task to make the best choice.

Highlighted below are 10 popular open source projects that enterprises can look to when considering open source alternatives in their IT infrastructure. I've highlighted a few key open source components in some of the most asked-for categories – Web Services, Service Oriented Architecture (SOA), Integration, Frameworks, and Libraries. Each project selected has at least one vendor offering commercial support. At the end of this article, I'll offer a basic checklist for those who are evaluating open source to ensure that the open source software they select fills all of their technical and business-related needs while managing corporate risk.

### Web Services

#### XFire

XFire is a Web Services framework (hosted by Codehaus) that allows developers to create and/or consume Web Services. Given its simplicity of use and built-in testing tools, it makes short work of Web Services by effectively eliminating the manual labor of generating WSDL and other artifacts of SOAP. XFire is compatible with a variety of commercial and open source Web Services frameworks, including Apache Axis and Microsoft Web Services.

Major pros of this relatively new project are that it's up-to-date, fast, built to integrate

with other frameworks like ServiceMix, and supports JAX-WS – an easy-to-understand architecture for Web Services development that can be used to build Web applications and Web Services with newer XML-based functionality.

However, because XFire is so new, many organizations have already become comfortable using Apache Axis – the original open source Web Services offering. Companies might feel more comfortable choosing Axis over XFire simply because of name recognition.

The license for Xfire isn't an OSI-approved license but it is very liberal, only requiring a copyright notice. Xfire is in the process of merging with Celtix, backed by Iona Technologies. Envoi Solutions and OpenLogic offer commercial support.

#### Axis2

Similar in function to XFire, Axis2 is a core engine for Web Services. Like XFire, Axis2 supports SOAP and other standards, but it also has integrated support for the Representational State Transfer (REST) style of Web Services. Axis2 is a more efficient, modular, faster, and more XML-oriented (it has the new fast AXIOM XML parser) solution than the original version. It supports plug-in modules that extend functionality for features such as security and reliability, factors critical to enterprise IT.

The primary downside of Axis2 is that it's plagued by the stigma of the first Axis, which has the reputation of being poorly documented and difficult to use. However, Axis2 does offer more documentation, which is a marked improvement over the earlier version.

Axis2 is an Apache Software Foundation project and is available under the Apache 2.0 license. Commercial support is available from several companies including Covalent, OpenLogic, and WSO2.

Choosing between Axis2 and XFire really comes down to what you need to plug into.



#### About the Author

Rod Cope founded EJB Solutions, later renamed OpenLogic, in 1998 (where he serves as CTO) to provide enterprise-grade tools and solutions that remove barriers to effective software development and deployment. Rod has provided technical leadership on significant projects for companies such as IBM, Anthem, Ericsson, Ford, Goodyear, Integral, CourseNet, and Digital Thoughts.

XFire is meant to be easily pluggable and work with a slew of other frameworks, including ServiceMix, while Axis2 is better suited for standalone use, although it's also pluggable if necessary. These two offer less expensive open source alternatives to proprietary Web Services solutions like those offered by Microsoft. Unless companies are using a full-blown SOA implementation where they get everything from a vendor (BEA, for example), they would probably want to opt for an open source solution like these two for reasons of cost and simplicity.

## SOA

### ActiveMQ

ActiveMQ is the most popular and powerful open source Message Broker. Although not quite a full-blown SOA solution, its flexible messaging technology is required for any SOA implementation. Widely considered one of the best Java Messaging Service (JMS) implementations available, ActiveMQ is fast, pluggable, and easy to embed into homegrown software, especially Spring-based applications. It's easily manageable through JMX, and it works with Apache Axis2 and XFire as well as servers like JBoss, WebLogic, and Geronimo. It also supports REST, many cross-language clients and protocols, including Java, C, C++, Perl, PHP, Ruby, and Python, and a wide variety of transport protocols. Regarding functionality, ActiveMQ provides a number of advanced messaging services offered by commercial vendors, such as Message Groups, Virtual Destinations, Wildcards, and Composite Destinations.

The drawback with ActiveMQ is that it's still fairly young and evolving, so it might require heavier configuration rework than enterprise developers want to accept. Although this extreme configurability is a major asset of ActiveMQ, it requires time to configure correctly for your circumstances.

ActiveMQ is an Apache Software Foundation project and available under the Apache 2.0 license. Commercial support is available from LogicBlaze and OpenLogic.

### ServiceMix

This project provides an Enterprise Service Bus (ESB) that combines the functionality of a Service Oriented Architecture (SOA) and an Event Driven Architecture (EDA) to create an agile enterprise ESB. It's built on Java Business Integration (JBI) and supports BPEL in conjunction with rules engines. It has dozens of transports and plug-ins and is lightweight, easily embeddable, and offers

integrated Spring support. It works standalone or within Geronimo or JBoss and sits on top of ActiveMQ. It has e-mail integration, Web Service integration, virtual file system integration, XSLT transformation, content-based routing, and Groovy support for end-point scripting.

ServiceMix is the most configurable and adaptable open source ESB implementation available. Like ActiveMQ, though, ServiceMix's configurability is a blessing and a bane. The project is still young, and it will go through many iterations and require a lot of configuration by developers. Also these open source alternatives don't offer a lot of user interfaces for policy administration, management, control, flow design, or the other bells and whistles offered by commercial competitors like Sonic and BEA.

ServiceMix is an Apache Software Foundation project and is available under the Apache 2.0 license. Commercial support is available from LogicBlaze and OpenLogic.

## Integration

### POI

Java-based POI gives developers easy and direct access to Microsoft Office files, including Word and Excel. The documents can be parsed and generated, which makes it easy to create them on-the-fly for downloading purposes from a corporate Web site. So, if a de-

## Checklist for Evaluating Open Source

Evaluating open source projects requires the same due diligence as reviewing commercial and proprietary solutions. Thus, there are two key areas to evaluate: functional and technical capabilities, and the ecosystem of services around the product (support, maintenance, community, commercial backing, etc.). With traditional proprietary commercial products, there is a vendor (along with partners) that typically provides the ecosystem of services. As part of any open source evaluation, an enterprise needs to understand those services and how they meet company IT needs.

It's no different with open source software. In this case, the services may come from commercial open source providers, from the community at large, or from other service providers. Whichever option you choose, you should evaluate the following key criteria:

- Are certified versions and updates available from a trusted source?
- How viable is the community? Structure? Size? Longevity? Activity?
- What type of license does it use? What are the restrictions? What are the licenses associated with the project and with other open source products it depends on?
- Is commercial-grade support available? What is the quality and responsiveness of support from the community?
- Is indemnification available? Have there been any IP actions?
- Does the product have the functionality needed?
- Does the product meet my technical requirements? Is the product available on the platforms I need?
- What other open source products are bundled in? What products does it depend on?
- What is the complexity of configuring and integrating this product with other components?
- How will I manage the stream of security patches, bug fixes, and other updates?
- What is the ROI for this product?

veloper needs to create a report that's required on the business end in Excel or Word, POI is the open source tool of choice.

Unfortunately, although it's top-notch for reading Word or Excel files, it currently can only create Excel files. The functionality to create Word files is reportedly in development.

POI is part of the Apache Software Foundation's Jakarta Project and is available under the Apache 2.0 license. Commercial support is available from OpenLogic.

## iText

This library lets Java developers create PDF files (without shelling out big bucks to Adobe for a PDF writer). Unlike HTML, which is browser-dependent, iText allows generation of read-only, platform-independent documents containing text, lists, tables, and images. Generally, iText is best used for Web sites that need to generate on-the-fly PDFs with the aforementioned elements. It's used by BIRT (Business Intelligence Reporting Tools, a recent addition to Eclipse) and other reporting tools to create downloadable PDF versions of their generated reports. There are no significant drawbacks to using this tool.

iText is available under the Mozilla Public License or LGPL. Commercial support is available from OpenLogic.

## Frameworks

### Hibernate

Hibernate is considered the object relational mapping tool of choice for Java developers. It's full-featured, robust, supported, well tested, updated, and heavily used in the enterprise. Hibernate is a critical component of Red Hat's JBoss Enterprise Middleware System (JEMS), which comes with extensive professional support, consulting, and training services. Hibernate supports over a dozen databases, including Oracle, DB2, MySQL, and PostgreSQL and also works either standalone or in conjunction with application servers like Geronimo and JBoss. It's highly configurable for performance tuning, offers a sophisticated caching mechanism, and has generation tools so one can generate database schemas from Java objects (or vice versa) or generate both from a mapping file that describes how the Java objects are to be persisted in the database. It also lets you drop to native SQL if the standard Hibernate Query Language (HQL) is insufficient.

There is currently a pending patent infringement suit filed by Firestar against Red Hat over Hibernate's technology. Although the tool itself is solid and many people believe the lawsuit

is without merit, enterprises should take this into account when selecting this tool.

Hibernate is available under the LGPL. Commercial support is available from a variety of companies including OpenLogic and Red Hat.

### Spring

Spring, available under the Apache License, is a highly configurable, lightweight, aspect-oriented, flexible application framework that can be used as an alternative to heavy J2EE applications. It can work standalone or in conjunction with application servers like JBoss or Geronimo. The big claim to fame for Spring is that it supports dependency injection and inversion of control, which means that business objects don't have to know all of the details of system configuration a priori; instead, Spring configures applications at start-up time by injecting each object with its necessary dependencies. This creates a very loosely coupled system that can be easily configured through XML whereby one can tune and reconfigure just by changing the XML description and without modifying the underlying code. This is quite valuable when transitioning from a development environment to QA to production where database connection parameters change, clustering is introduced, and other configuration properties need to be modified.

Spring is compatible with many of the other projects already discussed, including XFire, ActiveMQ, ServiceMix, and Hibernate. The only downside with Spring is that all those configuration parameters amount to vast expanses of XML. Without some upfront thought into your configuration strategy, all that uncompileable XML can get out of hand.

Spring is available under the Apache 2.0 license. Commercial support is available from Interface21 and OpenLogic.

## Libraries

### JUnit

JUnit, released under the Common Public License Version 1.0 and hosted on SourceForge, is the de facto standard for unit testing in the Java development world. It supports agile methodologies and makes it easy to blend testing infrastructure into build tools and environments so that developers can quickly see if a change they have made will break the code base. It integrates well into Ant and Maven, the two most popular open source build tools for Java developers, and Eclipse, the most popular IDE. JUnit can also work both in standalone mode through a command line and through

its own user interface to display results. In either case, errors and failures are reported clearly (with optional HTML output through integration with Ant) so developers know exactly where to look for problems.

JUnit is available under the Common Public License. Commercial support is available from OpenLogic.


### Jakarta Commons

Jakarta Commons is a three-part project focused on reusable Java components and is already heavily used at the enterprise level. The Commons Proper, a repository of reusable Java components, the Commons Sandbox, a workspace for Java component developers, and the Common Dormant, a repository for inactive Sandbox components, make up Jakarta Commons. The Commons is a great default library for enterprise developers that covers a range of basic development needs, from configuration support to logging to programmatic HTTP access to Command Line Interface (CLI) parsing to database connection pulling to e-mail generation to XSLT processing. It offers a solid base layer for every enterprise developer to look at instead of creating task-specific widgets — the fact is they can probably find what they need in Jakarta Commons.

The potential downside is that a lot of tools remain in the sandbox, meaning they're subject to change. For developers, this could mean needing to rewrite code in the future, which can be time-consuming.

Jakarta Commons is an Apache Software Foundation project and available under the Apache 2.0 license. Commercial support is available from OpenLogic.

## Conclusion

Obviously, neither the list of 10 products above nor the evaluation is all-inclusive. For every project mentioned, there are others that do the same tasks. That being said, open source often fights an uphill battle with respect to enterprise adoption largely due to the inaccurate perception that it is more complicated and less reliable than proprietary offerings. However, with the proper due diligence, enterprises can ensure that the open source projects they select are reliable, proven, and supported. And while only a fraction of the 144,000 open source projects on SourceForge may meet this stringent enterprise litmus test, many have matured, been battle-tested, and serve as legitimate alternatives to proprietary software in business-critical environments. 

**“Businesses that ignore the potential of SOA will find themselves outpaced by rivals who improve their agility and transform themselves into new kinds of enterprises**

— Yafim Natis, Gartner Analyst

**3-DAY EVENT!**

# SOAWorld

**Plus** **2007**

## Enterprise OpenSource Conference & Expo 2007

### TOPICS INCLUDE:

#### SOA Web Services

- > AJAX and SOA
- > Web 2.0
- > Universal SOA
- > Protecting Web Services
- > Troubleshooting SOA
- > Governance
- > Open-Source SOA
- > XBRL
- > Service Virtualization

#### Open Source

- > Open Source Business Models
- > Open Source ESB
- > OpenAjax Alliance
- > SaaS and Open Source
- > Spring, Hibernate and Eclipse
- > Seam
- > Open Source Penetration
- > Monetizing Open Source
- > Open Source Databases
- > AMQP
- > Open Source Middleware

**June 25-27, 2007**

**Roosevelt Hotel / New York City**

**Register Online!** [www.SOAWorld2007.com](http://www.SOAWorld2007.com)

11th International  
**SOAWorld**  
CONFERENCE & EXPO

2007 is to many industry insiders shaping up to be a major inflection point in software development and deployment, with SOA, Web Services, Open Source, and AJAX all converging as cross-platform and cross-browser apps become the rule rather than the exception.

Accordingly the 11th International SOA Web Services Edge 2007 again seeks to offer comprehensive coverage and actionable insights to the developers, architects, IT managers, CXOs, analysts, VCs, and journalists who'll be assembling as delegates and VIP guests in The Roosevelt Hotel in downtown Manhattan, June 25-27, 2007

Co-located with the 2nd Annual Enterprise Open Source Conference & Expo, the event will deliver the #1 i-technology educational and networking opportunity of the year. These two conference programs between them will present a comprehensive view of all the development and management aspects of integrating a SOA strategy and an Open Source philosophy into your enterprise. Our organizing principle is that delegates will go away from the intense two-day program replete with why-to and how-to knowledge delivered first-hand by industry experts.

**Visit [soaeosconference.sys-con.com](http://soaeosconference.sys-con.com) for the most up-to-the-minute information including... Keynotes, Sessions, Speakers, Sponsors, Exhibitors, Schedule, etc.**

2nd Annual  
**ENTERPRISE > 2007  
OPENSOURCE**  
CONFERENCE+EXPO

[SOAEOSCONFERENCE.SYS-CON.COM](http://SOAEOSCONFERENCE.SYS-CON.COM)

**REGISTER ONLINE TODAY**

**SAVE \$200!**

(HURRY FOR EARLY-BIRD DISCOUNT)

### BROUGHT TO YOU BY:



» **SOA World Magazine**  
focuses on the business and technology of Service-Oriented Architectures and Web Services. It targets enterprise application development and management, in all its aspects.



» **Enterprise Open Source Magazine**  
EOS is the world's leading publication showcasing every aspect of profitable Open Source solutions in business and consumer contexts.



For more great events visit [www.EVENTS.SYS-CON.COM](http://www.EVENTS.SYS-CON.COM)

**Exhibit and Sponsorship Info:**

**Call 201-802-3020 or email [events@sys-con.com](mailto:events@sys-con.com)**

# 64-Bit Linux Applications Need High-Quality 64-bit Database Connectivity

## Not choosing the right 64-bit database connectivity can cheat businesses out of the full benefit of 64-bit Linux

by Mike Frost

**S**ome businesses using Linux as an internal server platform may only now be confronting the challenge of migrating to 64-bit Linux distributions but are actually stepping into familiar territory for most Linux users in the business world. 64-bit Linux has been running for years on chipset families such as Intel's EM64T (Extended Memory 64 Technology) and Itanium, AMD's Athlon 64 and Opteron, and IBM's POWER. In addition, 64-bit Linux distributions have been offered for some time from top vendors such as Red Hat and Novell/SuSE, and have been available as a server operating system choice from hardware vendors such as Dell, IBM, and HP.



With all of the availability and access to 64-bit Linux around, why hasn't business been more aggressive in purchasing and deploying 64-bit Linux server platforms? What initially made many businesses hesitant to embrace 64-bit Linux were general concerns about application migration. What are the costs of migrating our existing 32-bit applications to 64-bit? What degree of benefit would our 32-bit applications experience by being migrated to 64-bit? As time has passed, new processor architectures such as x86-64 made 64-bit Linux more attractive to IT organizations of all shapes and sizes. x86-64 architecture allows both 64-bit and existing 32-bit applications to run simultaneously on a 64-bit operating system platform. Because of this, IT organizations now have much greater flexibility to decide which applications to migrate to 64-bit and when to migrate sparing them the business disruption and expense of a wholesale overhaul of all 32-bit applications in one enormous project.

The question of which applications would benefit the most from migrating to 64-bit is a subjective one, but in general, the answer is memory-hungry, data-intensive, multi-user applications such as relational database management systems (RDBMSs), business intelligence (BI), and data warehousing applications. When run as 32-bit, these applications can easily

hit the upper limit of addressable memory that's capable of being accessed by any 32-bit application even when there's more physical memory available to the operating system itself. The result of hitting this upper limit is an increase in the amount of paging to disk that must take place for the application to accomplish its tasks. This increase in disk I/O has the predictable effect of producing a performance bottleneck in the application that limits the ability of the application to scale for larger data sets or numbers of concurrent users. By running as 64-bit, these same applications can access all of the addressable memory available to the operating system and can therefore run entirely in memory if the Linux platform has sufficient RAM. The application's performance and support for additional concurrent users can then scale with the total amount of memory available to the operating system thus eliminating the performance and capacity bottleneck introduced by running as 32-bit.

Surprisingly for most of you reading this article, there is another key question that businesses of all kinds must consider when developing a strategy for developing or migrating applications to 64-bit on Linux. And it's the X factor in this scenario: "What impact will database connectivity have on the success of my 64-bit Linux applications?"

### About the Author

Mike Frost is a product manager for DataDirect Technologies, the software industry leader in standards-based components for connecting applications to data, an operating unit of Progress Software Corporation. In his role, Mike is involved in the strategic marketing efforts for the company's connectivity technologies. He has more than six years of experience working with enterprise data connectivity and is currently involved in developing data connectivity components including ODBC, JDBC, ADO.NET, and XML.  
[mike.frost@datadirect.com](mailto:mike.frost@datadirect.com)

## The Importance of Database Connectivity

Most people that I meet and ask about their database connectivity strategy usually tell me the same thing in one of three ways:

1. "All database connectivity is pretty much the same."
2. "Database connectivity is a commodity now."
3. "My database connectivity is 'good enough.'"

What's not said, but implied is the obvious, "I don't think I need a database connectivity strategy." The myth that each of these statements is built on is the idea that a database connectivity solution from one vendor is architecturally identical to what's offered by a different vendor. What this myth fails to take into account is the fact that there are very different approaches to designing and developing database connectivity. The architecture chosen for a given set of database connectivity components can mean the difference between an end result of poor quality and an end result of high quality.

Most business applications that run on Linux handle database access to a relational data source through some sort of data access

API such as ODBC, JDBC, or even some of the proprietary APIs available from the database vendors. Even if these applications weren't written using one of these APIs directly, it's a good bet that they make use of these APIs and subsequently load a driver or some type of database client libraries under the covers. Whether your business already has Linux applications compiled as 64-bit or is implementing a plan for migrating applications to 64-bit, you will want to carefully consider the type of database connectivity that's used. While there are an increasing number of options to choose from, picking high-quality 64-bit database connectivity can make a difference in determining whether your 64-bit Linux applications actually experience the kinds of performance benefits available to 64-bit applications or suffer from a range of potential performance and scalability limitations.

After investing a lot of time and effort in planning what applications to migrate to 64-bit then actually doing the work to migrate the applications, it seems baffling why some architects and developers wouldn't take the threat of introducing performance and scalability bottlenecks into an important application more seriously. Perhaps it's because they don't know

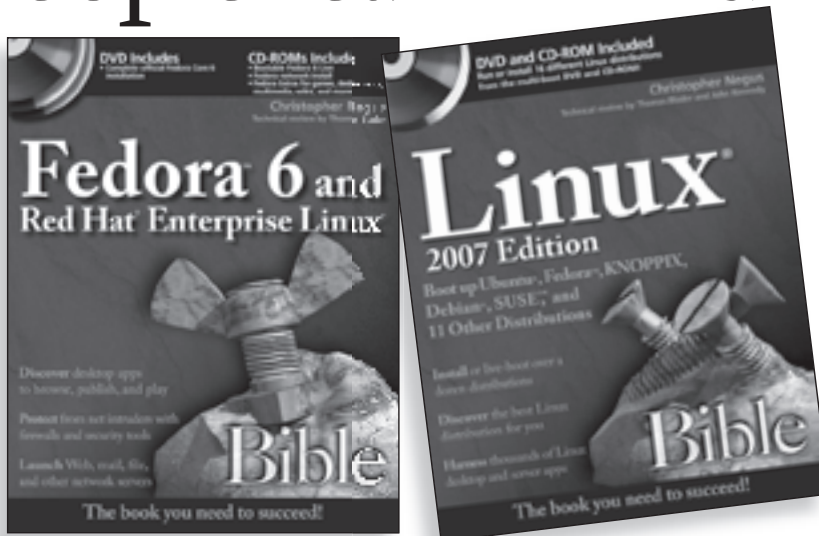
what characteristics to look for when choosing database connectivity and the impact that these characteristics can have on the success or failure of their 64-bit Linux applications.

## CPU-Bound, Not I/O Bound

As mentioned earlier, memory-intensive applications running as 64-bit can benefit from gaining access to more addressable memory space and thus can potentially access all of the memory available to the operating system. This can allow the application to run entirely in memory, which eliminates the performance and scalability bottlenecks introduced by the increased (and unnecessary) need to page to disk. For important 64-bit applications that run on Linux, businesses typically invest in server hardware with multiple CPUs and very large amounts of RAM to ensure that the application can run entirely in memory and support an increase in data set size, volume, or number of users. This investment works if the application is using database connectivity that is also designed to take advantage of increased numbers of CPUs or increased amounts of memory.

Along with disk I/O, network I/O is one of the two biggest factors impacting application performance. In the typical client/server

# When Chris Negus speaks, people learn Linux!



978-0-470-08278-2

978-0-470-08279-9

**Get the latest Linux Bibles by Christopher Negus.  
The books you need to succeed.**



deployment scenario, the application and database connectivity components run on one tier and connect across the network to an RDBMS located on a second tier. Poor-quality database connectivity components communicate with the RDBMS inefficiently due to an unnecessary amount of network I/O. These kinds of database connectivity components are said to be **I/O-bound** since they result in increased network I/O, which in turn introduces a performance and scalability bottleneck that can impact applications of any type. Most businesses can't upgrade the speed of their corporate network or WAN environments to compensate for the performance penalty paid for excessive network I/O, so the performance cost of repeatedly unnecessary TCP/IP round-trips adds up quickly for even a simple application operation. As a result, even though more RAM or CPUs can be added to the Linux application server platform, no additional performance or scalability benefit will be observed in the application.

When choosing database connectivity for your 64-bit Linux applications, look for components that aren't I/O-bound and introduce network I/O-related performance and scalability bottlenecks to your application. Such high-quality database connectivity components are said to be **CPU-bound**, meaning their performance and scalability is designed to be tied to the resources of the machine they're running on, rather than the speed and efficiency of the network. In contrast to poor-quality (I/O-bound) database connectivity, high-quality (CPU-bound) database connectivity exhibits better performance and scalability as more RAM and/or CPUs are added to the application platform. With high-quality data connectivity, the investment in time, effort, and resources that business put into migrating their key Linux applications to 64-bit can be fully realized.

## Architecture Matters

Database client libraries are supplied by

the database vendor and are made up of a collection of shared libraries that allow a user to develop an application using a proprietary API that is specific to the database being used. While it's possible (and not unusual) for applications to be developed to load the database client libraries directly, most business applications are developed to use standards-based database connectivity components such as ODBC or JDBC drivers. The most common architecture for standards-based data connectivity, the one most people think of when they think of ODBC drivers, is the so-called "classic" or "client-based" architecture. This approach to building database connectivity means that the driver or provider is built on top of the database client libraries that in turn connect to the RDBMS.


There's another kind of standards-based database connectivity that doesn't use database client libraries to connect to an RDBMS. The architecture used by this kind of database connectivity goes by different terms depending on the data access API in question. In the Java world, it's called **Type 4** and signifies a single-tier JDBC driver that communicates directly with a given RDBMS via TCP/IP. In ODBC, terms such as **wire protocol** and **clientless** are generally used to denote a similar architecture where a single-tier ODBC driver communicates directly with the RDBMS without using the database client libraries.

While most in the Java community are familiar with Type 4 JDBC driver architecture this approach and its benefits are much less known to users of other data access APIs such as ODBC. Many architects and developers are under the false impression that client-based database connectivity is inherently faster than standards-based database connectivity such as ODBC or JDBC. This is based on the assumption that because the RDBMS vendors developed their own underlying "wire-level" API to communicate with the database then RDBMS vendors will naturally build their database client libraries to be the best-performing

database connectivity components available. In fact, independent research contradicts this assumption and offers proof that this is not true. A report from database expert Ken North demonstrates that an ODBC application using an ODBC driver that uses clientless architecture is actually faster than the same application coded to talk directly to the database client libraries. Impressive results that should interest anyone using 64-bit applications on Linux.

In any situation where performance and scalability are a concern, but particularly where an investment is made in 64-bit, considering an inferior database connectivity option simply because it's "good enough" doesn't make sense. Just like a race car driver who wants to drive his car as fast as possible wouldn't tolerate a device that puts an artificial limit on his car's maximum speed, an architect or developer shouldn't accept the use of database connectivity components that offer anything less but the best possible performance and scalability. Look for Type 4 JDBC drivers, wire protocol ODBC drivers, or clientless database connectivity components that don't use database client libraries to get the most out of your 64-bit Linux applications.

## Conclusion

As businesses acquire more 64-bit Linux servers and migrate or develop new applications on this platform, expect to see 64-bit applications on Linux proliferate at an accelerated pace. While initial business strategies to migrate applications to 64-bit architecture center around a simple question of hardware availability and the potential impact of 64-bit architecture on the application itself, there's another key factor to consider – database connectivity. While an indifferent attitude to database connectivity might have been sufficient to get by with in the past, application performance and scalability can be negatively impacted by a continuation of this attitude as it relates to 64-bit application migration on Linux. Remember to look for and use CPU-bound, clientless data connectivity with your 64-bit Linux applications so you don't end up with a 64-bit application stuck running at 32-bit speeds. 

## References

- <http://www.eweek.com/article2/0,1895,1813588,00.asp>
- <http://www.tdwi.org/News/display.aspx?id=7845>
- <http://java.sun.com/products/jdbc/driverdesc.html>
- [http://www.sqlsummit.com/PDF/Benchmarks\\_SQL\\_APIs.PDF](http://www.sqlsummit.com/PDF/Benchmarks_SQL_APIs.PDF)

## Look Out for These Features Too....

The following is a short list of other features to look for in high-quality database connectivity:

- Support for tuning the number of rows fetched at a time from the database server (tuning for more rows per fetch can increase performance by reducing the number of network round-trips)
- Support for result set metadata caching (network round-trips can be reduced for successive executions of the same SELECT statement because the result set metadata only needs to be requested once)
- Support for native parameter arrays (subject to its support in the underlying RDBMS)
- Support for options that limit the size of data fetched from long or LOB data type columns (even though long data in a particular row may be up to 4 GB in size, most situations don't require the entire result to be fetched)

# 2007 VIRTUALIZATION CONFERENCE + EXPO

[www.virtualizationconference.com](http://www.virtualizationconference.com)



Register Now!  
**EARLY-BIRD  
SAVINGS!  
\$200**

ROOSEVELT HOTEL  
NEWYORK CITY

**June 25-27, 2007**

## Virtualization: Solutions for the Enterprise.

### Delivering The #1 i-Technology Educational and Networking Event of the Year!

As today's CIOs and IT managers rise to the challenge of using their enterprise computing infrastructure as cost-effectively as possible while remaining responsive in supporting new business initiatives and flexible in adapting to organizational changes, Virtualization has become more and more important.

A fundamental technological innovation that allows skilled IT managers to deploy creative solutions to such business challenges, Virtualization is a methodology whose time has come. The fast-emerging age of Grid Computing is enabling the virtualization of distributed computing, of IT resources such as storage, bandwidth, and CPU cycles.

But Virtualization can also apply to a range of system layers, including hardware-level virtualization, operating system level virtualization, and high-level language virtual machines.

**Register Online!**

[www.VirtualizationConference.com](http://www.VirtualizationConference.com)



**THE FIRST MAJOR VIRTUALIZATION EVENT IN THE WORLD!**

**HURRY, FOR EXHIBITOR AND  
SPONSORSHIP OPPORTUNITIES  
CALL 201-802-3020**

*Learn from  
the Experts...*

— BROUGHT TO YOU BY —



For more great events visit [www.EVENTS.SYS-CON.com](http://www.EVENTS.SYS-CON.com)

COPYRIGHT ©2007 SYS-CON MEDIA. ALL RIGHTS RESERVED

ENTERPRISE  
**OpenSource**  
MAGAZINE

**SOA**WORLD  
MAGAZINE

# Am I Seeing Python Everywhere?

## Or is it just my imagination

by Paul Nowak

I've been a long-time Python fan. It's a language that's so easy to program with that I end up turning to it for a great many things and I find myself wishing for a Python interface in pretty much any application that moves or plays with data in anyway. I also give Python-based products a good hard look when comparing alternatives. We've adopted the Python-based Plone as the content management system for our Website (I know – Drupal has major momentum but Plone is a very solid product also) and it's been a very good experience for the sector we serve.



I'm writing this article because I've noticed that Python has been playing the key programming language role in three new places in the past month. See it once, no big deal. See it twice, raise some eyebrows. See it three times in widely different places? Well, that means it's time to write an article and get the buzz rolling on what the heck is going on here.

Here's where I've run into Python recently:

1. Python seems to be the primary language for writing simple GUI wrapper front ends for traditional command-line programs in Ubuntu, the wildly popular Linux distribution. Ubuntu runs GNOME by default so... what I am really talking about is GNOME. I'm a recent adopter of Ubuntu and as I've dug through some of the programs I'm using, I am seeing Python as the basis for a lot of them. This shows just how out of the loop I am on the core GNOME technology platform, but it's nice to see something you like forming the foundation of something new that you also like.
2. The OLPC – One Laptop Per Child project – is using Python very heavily. Python is a basis for the applications on the OLPC, and it's also a focal point of the educational aspect of the product in that if kids using the product are going to learn a programming language, Python is first among equals. Not only so kids learn the basics of programming, but also so they can begin to supply the OLPC with their own programs and applications, thanks to this Python-everywhere mentality in OLPC's approach.
3. Python is all over the IBM/Lenovo restore

CD collection for the Thinkpad series. I recently had to rebuild an IBM T43 Thinkpad from OEM disks and two of the disks seemed to have a complete Python 2.2 tree on them as I watched the dialog box fly by on the progress monitor. I've known for a long time that Python is part of the Open Office installer, but to see it here is two orders of magnitude different. IBM is using Python to install Windows! That's a bit of a twist in my view.

In each of the above cases, Python is moving away from what I consider its traditional roots as a scripting language for servers, for server-side applications, and for Web applications. That's the coolest thing about what's happening here. Python is already a cross-platform hit – running on Windows, Unix, and Mac. Now, it looks like Python is also a cross-layer language as well – running on servers in its traditional role and running on client PCs (Ubuntu and OLPC), filling multiple and very different roles there. One day I see Python as part of the Windows OEM installer kit for IBM/Lenovo Thinkpad OS rebuilds, the next as a front-end wrapper in GNOME, and the next filling a couple of roles in the OLPC client platform.

My hat is off to the Python founders and team. I'm seeing a long-time compelling language take big steps and grab mindshare with lots of people – people who are in very influential positions and have responsibilities for a diverse range of platforms. It's great to see that my favorite language from years back

### About the Author

Paul Nowak first used Linux in 1995 while migrating from Sun to Linux at the University of Michigan. He used Linux in subsequent IT projects including web, telecom, telemetry and embedded projects and is currently CIO of a small professional association based in Washington D.C.  
[paul@naaee.org](mailto:paul@naaee.org)

has grown beyond what any of us might have expected years ago when it was the lesser "P" in LAMP with the greater Ps being PERL and PHP.

My hunch about the future though is that what we are seeing today is just the beginning for Python. I think we are on the cusp of an explosion in software development, availability, and user experience. This explosion is going to be driven by wider adoption of open platforms based on open source \*nix systems.

As these systems go from sub 2% of the market to plus 10% of the client PC market, the platforms will become both more used and more of a focus for development. This will drive the typical adoption curve we see in technology. Adoption coupled with development, coupled with full open access, coupled with a really great programming language will drive massive amounts of development. Think of about 20 million kids running the OLPC around the world with open access to build new software tools for the platform they use. The intellectual talent of the kids of the world is bound to unleash a waterfall of new software.

The same thing will be happening among adults. I can tell you that when I see applica-

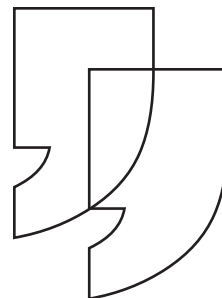
tions written in Python all over my Ubuntu desktop, I know that I too have the ability to participate in the development of software running on my own PC. I never had this feeling in Windows. It's a big change to think about and it's also big due to platform consistency from server to desktop to...eventually the phone.

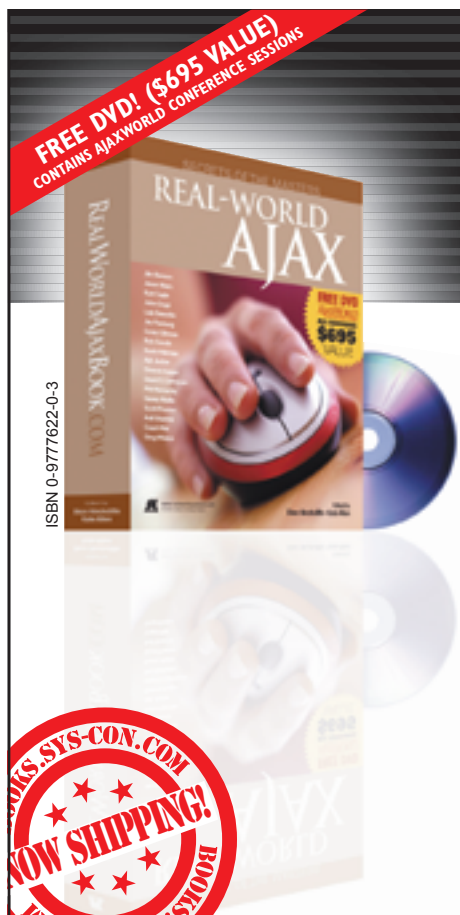
This turn of events just makes me happy. I think people should be able to build the technology they want and I also think people, working in a decentralized and networked world, will be an extremely powerful force in software – as we've seen already but to an extent I don't think we can imagine.

Software by the people for the people seems, to me, like it will meet our needs far better than the products we are seeing out of the corporate sector these days. In fact, more than ever, I would say open software will explode outward on a scale we don't currently anticipate. Open software's eventual place at the center of the software world is assured and Python is a growing part of ensuring the outcome.

Seeing Python everywhere is just a sign that these good times are starting to roll. The good old days are ahead for OSS and for Python. ○

Seeing Python everywhere is just a sign that these good times are starting to roll. The good old days are ahead for OSS and for Python.





# JOIN THE AJAX REVOLUTION!

## Secrets of the Masters: Real-World AJAX


Edited by Dion Hinchcliffe & Kate Allen

*"If you're looking for a one-stop shop for an AJAX book...you would have a long search to find a better overall resource than what you find in the chapters of this book."*

Order Online at [RealWorldAJAXBook.com](http://RealWorldAJAXBook.com) and get

# 40% OFF

Regular Bookstore Price!

 [books.sys-con.com](http://books.sys-con.com)  
from the World's Leading i-Technology Publisher © COPYRIGHT 2007 SYS-CON MEDIA

# Mixing Data & Data Structures in an Object Database

## An introduction to the trie

by Rick Grehan

When your next Java application calls for a database backend, before you reach for JDBC and a relational database, stop for a moment and consider another possibility: an object database.



As we hope to show in this article, an object database may not only simplify coding chores, but its capabilities may enable application solutions that you would otherwise not thought of.

Of course, the most apparent benefit you'll get from using an object database is the fact that you won't have to wrestle with two paradigms – object and relational – in a single application. The design of your application code and the design of your database will be object-oriented throughout.

However, the advantages you'll derive from an object database go beyond the simple fact that an object database is more easily incorporated into a project already written in an object-oriented language. One exceptional advantage of an object database is its ability to store structure as well as data. Put another way, its ability to preserve the structure that you've already built into your application. With a relational database, you have to tear down and reassemble that structure as you move data between the application and the database.

Typically, when one thinks of a database, one thinks of a repository of data. In a relational database, the information in the database is organized – more or less – along the lines of normalized data corralled into the proper tables. Data is arranged in “tuples,” which tends to gather related information together. But any sort of higher-level data structures must be implemented in the SQL code (procedures) that access the data. Not so with an object database, which can mingle data with data structures in the same database. In fact,

with the “right” object database engine, you can add new data structures to an existing database at will (or, rather, as the situation dictates).

We will illustrate what we mean using an example that combines two data structures. As our database engine, we will choose the open source object database db4o (available from [www.db4objects.com](http://www.db4objects.com)).

### Homemade Thesaurus

We will use as our illustration the data structures that one would employ to create something like ThinkMap's Visual Thesaurus ([www.visualthesaurus.com](http://www.visualthesaurus.com)). Visual Thesaurus is a clever visual interface that represents words as existing in a network of connected nodes. Each node is a word, and the connections represent relationships between synonyms.

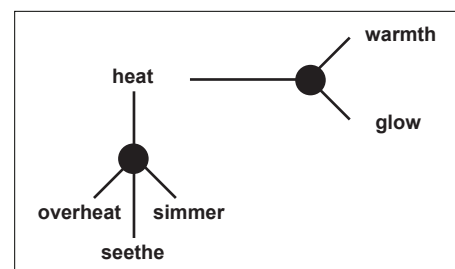


Figure 1: A mock-up of the user interface provided by Visual Thesaurus.

Our intent here is not to create our own version of Visual Thesaurus, instead, we're going to imagine what sorts of data structures might exist “behind” the user interface of a Visual-Thesaurus-like application and consider how we might represent those structures. Finally, we want to illustrate how we could use an object database to store our imaginary thesaurus' data.

Let's design our data structures from the bottom up. So, on the ground floor, we'll need a

#### About the Author

Rick Grehan is a QA Engineer for Compuware's NuMega Labs in Merrimack, NH. He has been programming for nearly 30 years, and has written software in languages ranging from Fortran to Fortran, 8-bit BASIC to Java, and 6502 assembly language to PHP. He is also a freelance writer. His articles have appeared in BYTE Magazine, JavaPro, Linux Journal, The Microprocessor Report, Embedded Systems Journal, and others. He has also co-authored three books; one on RPCs, another on embedded systems, and a third on object databases in Java. [regrehan@hotmail.com](mailto:regrehan@hotmail.com)

structure for storing each word. This structure will need to take into account the fact that a given word often has multiple parts of speech. Furthermore, each part of speech will connect to a different network of synonyms.

Consequently, we'll decompose this structure into two classes. First, a POSSynonyms class that will identify the part of speech and hold an array of references to the synonyms corresponding to that part of speech. The class looks like this:

```
public class POSSynonyms
{
    private int pos; // Part of speech
    private ThesaurusEntry[] synonyms; // Synonyms

    ... POSSynonyms methods ...
}
```

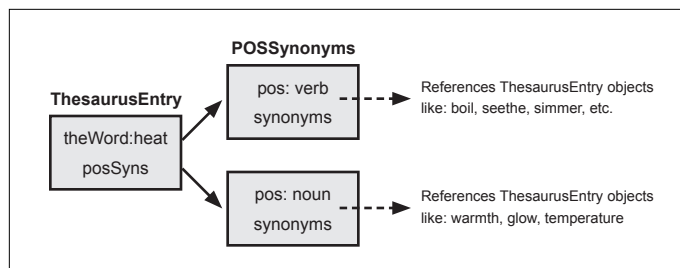
Obviously, this class depends on the ThesaurusEntry class, which is shown below:

```
public class ThesaurusEntry
{
    private string theWord;
    private POSSynonyms[] posSyns;

    ... ThesaurusEntry methods ...
}
```

The first element of this class, theWord, holds the string of the word itself. The second is the array of POSSynonyms.

The result lets us create a network of interconnected synonyms. You can get a feel for what it might look like if you examine Figure 2. The ThesaurusEntry object for the word “heat” would include a posSyns array pointing to a POSSynonyms object for the verb synonyms, and another for the noun synonyms. The verb POSSynonyms object would, in turn, include a synonyms’ array that points to ThesaurusEntry objects corresponding to the words boil, seethe, simmer, and so on. The POSSynonyms object would include a synonyms’ array pointing to ThesaurusEntry objects for warmth, glow, temperature, and so on.



**Figure 2:** The connections between ThesaurusEntry objects and POSSynonyms objects

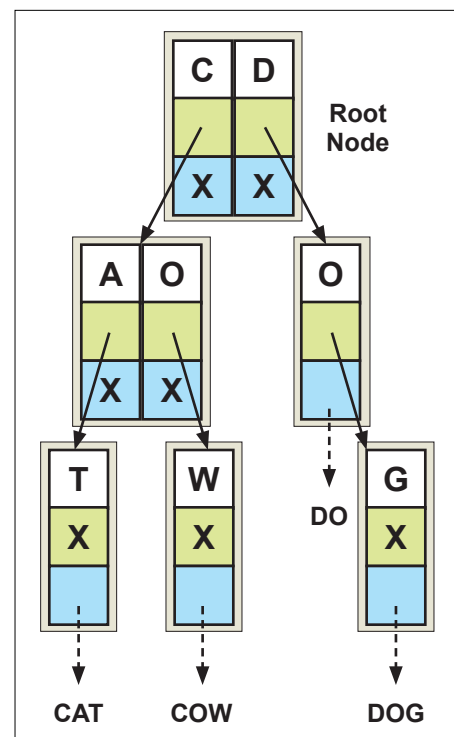
## Building a Trie

So much for the storage of words and connections to synonyms. Now we need a data structure that’s suitable for locating a word so a user can do a search. Note that this data structure only has to locate words. We don’t require lexicographical ordering (as would be provided in, say, a binary tree or a B-Tree). While there are several data structures that would serve for this task, for this example, we’ll use a trie.

A trie is a tree-like data structure. However, whereas most tree data structures store one or more entire keys (in our case, keys are words) in each node, a trie chops the key apart and stores pieces of it on each node. So our trie will store one letter on each node.

This will be easier to understand if you look at the diagram in Figure 3. Each node in a trie contains three elements and each element is an array:

- **A character array.** This holds the letters of the words that the trie stores.
- **A node-pointer array.** Members of this array point to child nodes in the trie.
- **A data-pointer array.** Members of this array point to the actual data that the trie indexes.



**Figure 3:** A trie data structure. This trie stores four words: cat, cow, do, and dog.

The trie is anchored by its root node. So, to locate a word in the trie you search for the first letter in the root. If you find the letter, you follow the corresponding node pointer to the child node. On the child, you look for the second letter in the word. If found you move deeper in the tree.

The search process terminates in one of three ways.

- You reach the end of the word, and the data-pointer references the word you’re looking for.
- You read the end of the word, and the data-pointer is null. In that case, the word is not in the trie.
- You encounter a node whose node-pointer is null, but you’re not done with the word. Again, the word is not in the trie.

Our implementation of a trie is composed of three classes. The first, the trie class, is the outermost encompassing class. The “outside world” interacts directly with objects of the Trie class, which encapsulates all the mechanisms needed to manage the trie.

The trie class includes a single data member:

```
private TriePnode root;
```

This root member is the node at which all searches begin. To be precise, the root node will hold all the first letters of all the words in the trie. The root node reference is of class `TriePnode`, which looks like this:

```
public class TriePnode
{
    private char[] chars;
    private TriePnode[] pnodes;
    private TrieDnode[] dnodes;

    ... TriePnode methods ...
}
```

The body of the trie structure is made up of `TriePnode` objects. Each `TriePnode` carries the array of characters that hold the letters of the words (as shown in Figure 1), the `pnodes` array (the pointers “down” into the lower levels of the trie), and the `dnodes` array (the references to the actual data objects).

Finally, the leaves of the trie are `TrieDnode` objects. The `TrieDnode` class is more or less a wrapper that includes a `ThesaurusEntry` item as its single member.

So now, we have all the pieces that we need to build our `Thesaurus`. We have the structures that hold the words and connections to the synonyms, and we have the structures that support searching.

## Time for Persistence

We now have all the pieces needed to build our thesaurus: structures to hold the words, connections among synonyms, and a trie for locating words. Now what we need is someplace to put everything. This is where our object database, `db4o`, steps on the stage.

`db4o` is an open source object database from `db4objects` available for downloading at <http://www.db4objects.com>. There are two versions of `db4o`, one for Java, and another for the .NET platform. (A Mono version is also available, though it's largely identical to the .NET version.) `db4o` can be used either as an embedded database library that executes in the same process space as your application, or in client/server form for remote or multi-user access to a database server. We'll be using `db4o` in its “monolithic” form (part of a single application).

`db4o` will do more for us than simply serve as a persistence mechanism for our thesaurus. It will give us everything we'll need to get all our classes into a database in the first place. Put another way, we can “bootstrap” our data into the database using `db4o`'s API then let `db4o` slip into the background as we use the navigation capabilities we've built into trie and `ThesaurusEntry` to fetch words and synonyms.

We'll be adding words “the hard way” – i.e., using hard coding – so you can see how the class methods work. Obviously, if we're going to add a lot of words, we'd write code to read a vocabulary from a file.

So, to add an entry for the word “heat” as both a noun and a transitive verb, our code would look like this:

```
ThesaurusEntry te;
POSSynonyms ps;
TrieDnode td;
Trie thesaurusTrie;
```

```
...
te = new ThesaurusEntry("heat");
ps = new POSSynonyms(PartsOfSpeech.NOUN);
te.addPOSSynonym(ps);
ps = new POSSynonyms(PartsOfSpeech.VERB_T);
te.addPOSSynonym(ps);
td = new TrieDnode(te);
thesaurusTrie.insert(te.getWord(), td);
...
```

The code should be pretty easy to follow. First, we create a new `ThesaurusEntry` for the word “heat.” Next, we create `POSSynonym` objects for noun and transitive verb. (`PartsOfSpeech` is a helper class we defined to enumerate the parts of speech.) The `POSSynonym` objects are added to the `ThesaurusEntry`, which is encapsulated in a `TrieDnode`. Finally, we insert the word into the trie, with the `ThesaurusEntry` node associated.

Similar code can be used to put more words into the trie. Notice, however, that this code doesn't create synonym connections. That happens with a bit of Java as follows:

```
...
td = thesaurusTrie.search("heat");
te = td.getData();
ps = te.getAPOSynonym(PartsOfSpeech.NOUN);
ps.addSynonym(thesaurusTrie.search("warmth").getData());
ps.addSynonym(thesaurusTrie.search("glow").getData());
...
```

This code assumes that we already have the words “heat,” “warmth,” and “glow” stored in the trie and that they have `NOUN` entries associated. We locate the “heat” `ThesaurusEntry` and create connections from it to “warmth” and “glow.”

Of course, we'll want connections pointing the other way too. So, we'd want code like:

```
...
td = myTrie.search("warmth");
te = td.getData();
ps = te.getAPOSynonym(PartsOfSpeech.NOUN);
ps.addSynonym(myTrie.search("heat").getData());
...
```

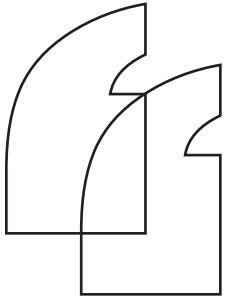
and something similar for “glow.” (Note that all this code could be considerably shortened if we dispensed with all the intermediate objects. We've shown them, however, to make the mechanics clearer.)

But, wait, this is all still in memory. How do we get it into our database? Actually, with `db4o`, nothing could be simpler. The code looks like this:

```
...
db = Db4o.openFile("thesaurus.yap");

... <code to create the trie here> ...

db.set(thesaurusTrie);
db.commit();
db.close();
...
```



# An object database can mingle data with data structures in the same database. In fact, with the “right” object database engine, **you can add new data structures to an existing database at will or as the situation dictates**

We’ve left out the imports, and we’ve indicated where all the code goes to build the trie. But once the trie is built, the only method we have to call to put the trie in the database is `db.set()`. (We also `commit()` the transaction and `close()` the database, but the real workhorse is `db.set()`.)

`db4o` implements persistence by reachability. That means that when we first store an object in the database with the `set()` method, `db4o` will crawl through our object’s tree, locating all referenced objects, and make them persistent too. In other words, when we store `ThesaurusTrie`, everything is references is stored too. `db4o` puts the whole kit and caboodle in the database for us in one shot.

## Reading Our Thesaurus

When we wrote our thesaurus, we did so by building all the data structures in memory then storing them all in the database. We could, if we wanted, read everything into memory for searching the database. However, we may want to be more memory-frugal. We may want to only load those parts of the data structures we need to satisfy a search.

`db4o` lets us do that by controlling the “activation depth” of objects read from the database. Setting the activation depth tells `db4o` how deep into an object tree it should go when it retrieves objects from the database.

You can see how this work if you look at the code for the `search()` method that we have overloaded to work with the `db4o` database:

```
public TrieDnode search(String key,
    ObjectContainer db) {
    TriePnode t;
    TrieDnode d;
    char c;
    int index;

    // Empty trie?
    if((t=root)==null) return(null);

    int slen = key.length();
    for(int i=0; i<slen; i++) {
        c = key.charAt(i);
        if((index=t.isCharOnNode(c))==-1) return(null);
        if(i==slen-1) break;
        t = t.getPnodePointer(index);
        db.activate(t, 2);
    }
    d = t.getDnode(index);
    db.activate(d, 2);
    return(d);
}
```

The algorithm begins at the root and verifies that the first character of the word exists on the root node. If so, the algorithm fetches the `TriePnode` corresponding to that character’s location in the node. Then the algorithm calls `db.activate(t,2)`. This tells the database to fetch refer-

ences at least two deep so that not only is the node itself fetched, but the content of the arrays in the node are fetched as well.

Similarly, after the call to `getDnode()` – which fetches the data node – we call `db.activate(d, 2)` to fetch the content of the `TrieDnode`’s `ThesaurusEntry`.

With our database-enabled search algorithm in hand, we can now construct a simple routine to fetch the synonyms for a particular word.

```
...
td = myTrie.search(args[0],db);
if(td==null) {
    ... word not found ...
}

te = td.getData();
for(int i=0; i<td.numberOfPOSes(); i++) {
    ps = te.getIthPOSSynonym(i);
    db.activate(ps,2);
    System.out.println(strPOS(ps.getPOS()));
    for(int j=0; j<ps.numberOfSynonyms(); j++)
        System.out.println(ps.getSynonyms(j).getWord());
}
...
```

The search algorithm returns a `TrieDnode`. We use `getData()` to fetch the `ThesaurusEntry` then step through all the parts of speech for that. Finally, we step through all the synonyms for each part of speech and display the word. The result would look something like this:

```
c:\ SearchDatabase heat
NOUN
warmth
glow
VERB
```

Which shows that the word “heat” is recorded in the thesaurus as both a noun and a verb, and the noun form of the word is associated with the synonyms “warmth” and “glow.” There are no synonyms associated (yet) with the verb form.

## Persistent Thesaurus

Of course, the charm of Visual Thesaurus is its almost life-like user interface and the ease with which one can explore a network of related terms. We will have to leave that implementation to the reader.

Our goal was to illustrate how easily an object database could be used to persist the data structures behind such a UI. The really nice advantage of using an object database like `db4o` is the fact that the structure that we have defined in our classes is the same structure that exists in the database. We didn’t have to write any translation code to move between our object structure and a relational representation. `db4o`’s easy-to-understand API made our construction work that much easier.

# Experimenting with MontaVista Linux:

## Lessons learned by solving a problem

by Mirosław Zakrzewski



This article presents my working experience with the MontaVista Linux distribution (MontaVista Professional Edition 4.0 base on the 2.6.10 Linux kernel) running on a PowerPC processor board. It will present the physical memory mapping in Linux and its limitations using examples of a real problem and the suggested implemented solution that involved changes to kernel configuration.

Linux plays a significant role in embedded application development. It proved to be a platform of choice for soft real-time applications and with Linux distributions such as MontaVista Linux is being improved and hardened for mission-critical embedded application that require a hard real-time response.

Standard Linux (from [www.kernel.org](http://www.kernel.org)) supports two kinds of real-time schedulers (**SCHED\_RR**, **SCHED\_FIFO**) and the preemptive kernel. The hard real-time additions to the Linux kernel offered by MontaVista makes Linux ready to be a platform for developing hard real-time embedded applications. However, at this point, our goal is not the real-time capabilities of Linux but to address some general issues.

Our primary task was to bring up a system on a PC board in record time. However, the applications to be ported required more memory than an existing Linux platform could support. In general, the problem wasn't in the availability of the physical RAM but in the fact that the kernel could only see part of it. The board was equipped with more physical memory than the original kernel was able to access. Facing this problem, we had to modify the Linux kernel to get access to the whole available RAM.

The work started with a cross-development environment setting and some kernel modifications described below. However, before we begin, let's look at the Linux addressing.

### Linux Address Space

Each process in the 32-bit address space can address 4GB of memory. This 4GB memory is divided into two parts:

- User space ( from **0x0** to **PAGE\_OFFSET - 1** )
- Kernel space ( from **PAGE\_OFFSET** to **0xFFFFFFFF** )

Usually, the **PAGE\_OFFSET** define is assigned the value of **0xC0000000**. It means that the user address space covers 3GB of addresses starting from address **0x0** and the kernel space covers 1GB of addresses starting from address **0xC0000000** (see Figure 1).

The user can customize the value of **PAGE\_OFFSET** at the kernel compilation time (the **Base Address** name is used when referring to it, as well).

It's very important to know that the physical RAM memory installed in the system is mapped to the addresses reserved to the kernel space (starting at **PAGE\_OFFSET**). Therefore, if Linux is installed on the system with a huge amount of RAM, there may not be enough addressing space to do the mapping. So, some other mechanisms may be required to make the whole RAM visible.

#### About the Author

Mirosław Zakrzewski works for Ericsson Canada in Montreal and is currently on assignment in Sweden. He focuses on developing server platforms for next-generation IP networks.  
[Mirosław.Zakrzewski@Ericsson.com](mailto:Mirosław.Zakrzewski@Ericsson.com)

## Problem Description

Following the model described above, theoretically in the 32-bit addressing, the system can access a maximum 1GB of physical memory, assuming that the base address is set to **0xC0000000**. So there's the question of what to do when there's more than 1GB of memory available, or some kernel areas are already used (leaving less than 1GB of address space for mapping).

To address this issue, recent Intel x86 microprocessors include a feature called "Physical Address Extension" or PAE that adds an extra four bits to the standard 32-bit physical address. Linux kernel version 2.4 took advantage of PAE to support up to 64GB of RAM. However, the linear address is still composed of 32 bits, so there's no permanent mapping of the whole amount of RAM.

Our board was equipped with the PowerPC processor and there was no need for a huge extension of RAM. In our case, only an extra 256MB of RAM visible to the system was sufficient.

In the next paragraphs, I will describe our solution to get the extra 256MB accessible to the system.

The step-by-step process of kernel configuration will be presented in detail as well.

## Solution

When looking at the picture presented in Figure 1, there's one obvious solution. What will happen when the base address is changed from **0xC0000000** to **0xA0000000**? By moving the base address down by **0x20000000**, there will be more address space mapped to the kernel and less to the user space. In this case we'll gain 256MB of memory access. This is exactly what we wanted (having more address space in the kernel to cover the extra RAM).

Figure 2 illustrates the address partition after the base address change. The modifications to the kernel were straightforward and done in no time as you'll see below.

## Kernel Configuration

Before making the kernel modifications, the environment has to be set up for access to the proper cross-compilers and linkers. Remember that the embedded kernel will be built, not on the native environment. In our case the embedded platform was a board, running PowerPC processor and the development platform was based on Solaris (a desktop Linux could be used as well).

- Figure 3 illustrates the development environment, which consists of:
- A Power PC board, where the Linux kernel produced will be executed
  - A development system where the Linux kernel will be configured and built using MontaVista tools like cross-compilers, linkers, debuggers
  - A serial connection between the target board (PowerPC) and the development platform (Solaris, Linux)

The process of building the new kernel consisted of the following steps:

- Set up the environment

All the tools and the kernel source code have to be collected from the MontaVista distribution. The right paths also have to be set up to point to the right directories, etc.

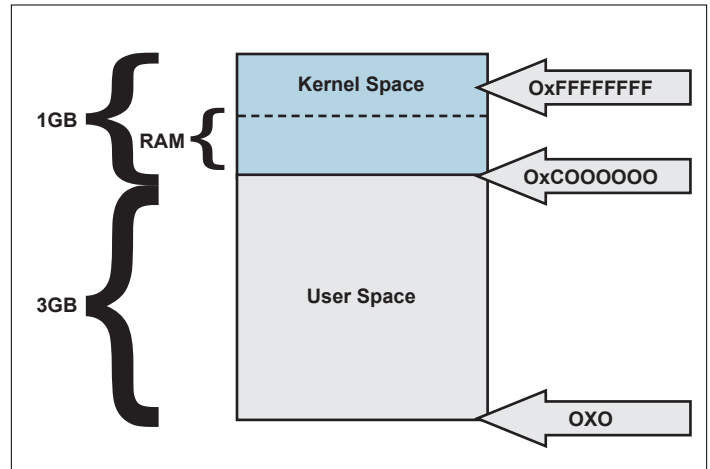


Figure 1: Address space division in Linux

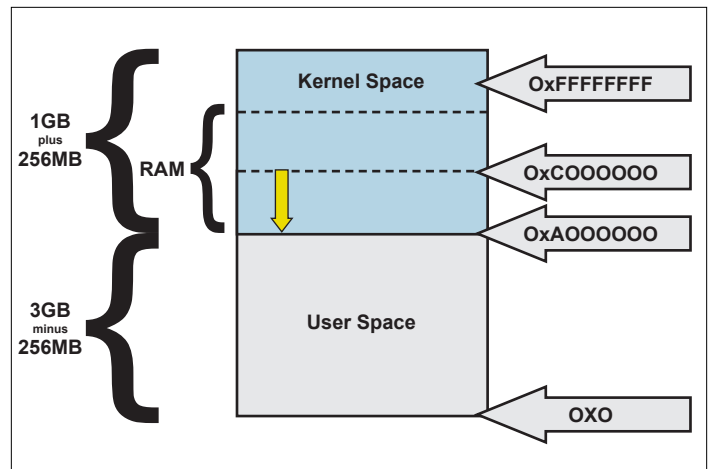


Figure 2: Address space division after base address modification

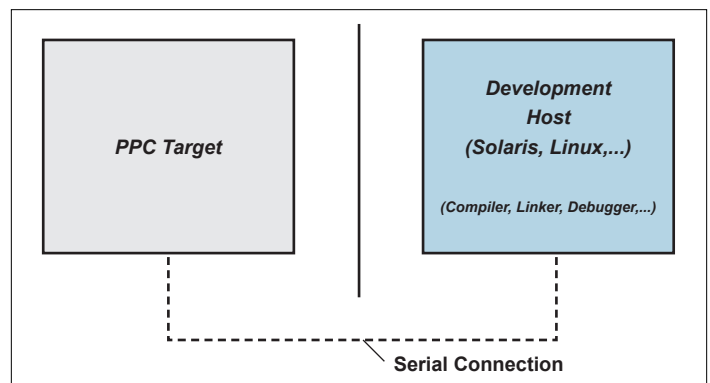


Figure 3: Development environment



Figure 4: Advance setup option



Figure 5: Advance setup option

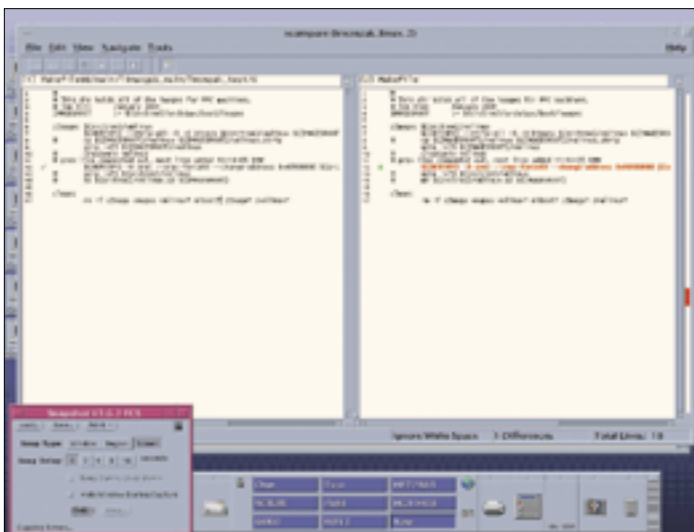


Figure 6: Makefile modifications

- Configure the kernel by executing (from the Linux base directory) `% make menuconfig`. Executing this command brings the menu seen in Figure 4.

Next, the “**Advance setup**” option was chosen to change the default values to the customized values. Figure 5 illustrates the menu of the parameters to be changed.

The following changes were made to this menu:

- “**Set low memory size**” to be modified to **0x40000000** (represents 1GB of RAM memory)

This parameter specifies the size of the memory the system can see. In our case, the original value wasn't set to cover the whole range of 1GB addresses because there was already some address space taken for some other purposes.

- “**Set custom kernel base address**” to be modified to **0xA0000000**


The default base address is set to **0xC0000000**.

The last change left to do was to modify the **Makefile**. Because the base address has been change, we have to adjust the “**srec**” generation accordingly. In this case, modify the address from **0x40500000** to **0x60500000**. (See Figure 6.)

And that's all. What's left is to build the **kernel** and **srec** file by running the “**make**” command from the Linux base directory.

After the new kernel is built and burnt (over the serial connection) into the board, resetting the board will conclude the development process. Now, comes verifying if the kernel can see the whole memory. Before executing any more sophisticated tests the first way to verify the system is to run the “**top**” command. And in fact, the top display shows that the kernel can see all of the available RAM.

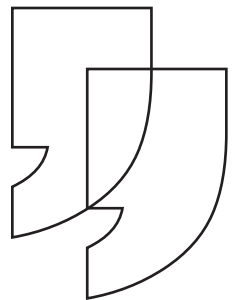
## Conclusion

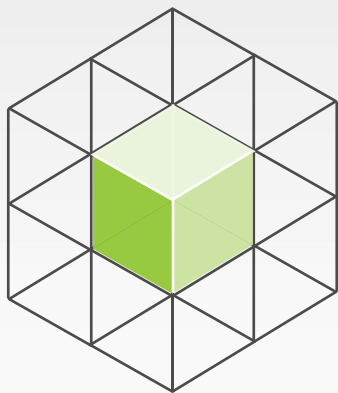
I'm sure that there are many different ways of solving the problem described here. The purpose of this article was to document one possible solution. Usually when a project requires fast results (as in our case) the most efficient way is best. Our example shows how easy the changes were to make following the clear theory behind it. And it shows how powerful open source can be when delivering a solution that requires some platform adjustments. 

## References

- Daniel P.Bovet and Marco Cesati.  
*Understanding the Linux Kernel.*
- <http://www.mvista.com/>
- <http://www.linuxjournal.com/>

The applications needed more memory than the Linux platform could support; the problem wasn't in the availability of the physical RAM but that the kernel could only see part of it





# 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.

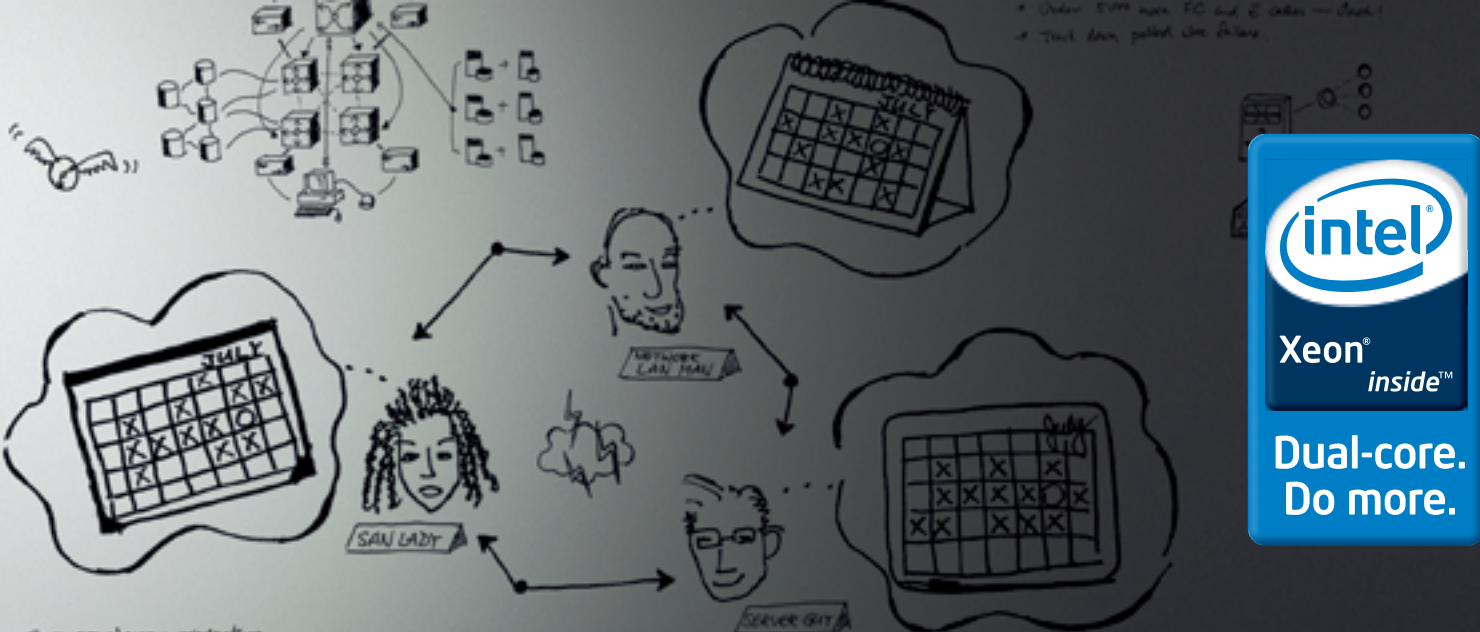
Specific objectives of the Open Management Consortium include:

- 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.



[www.open-management.org](http://www.open-management.org)



YOU HAD THE  
STRENGTH ALL THE TIME.  
YOU JUST NEEDED THE  
RIGHT INFRASTRUCTURE.

### The HP BladeSystem c-Class with Virtual Connect Architecture.

The HP BladeSystem c-Class is a different breed of infrastructure that utilizes a unique, integrated design. It only has to be wired once for easy setup and expandability, speeding up your process and helping you allocate resources. And because the virtual LAN/SAN connections are determined up front, the infrastructure won't require adjustments every time you add or move a server, saving you valuable time and effort.

In addition, the HP ProLiant BL460c server blade — featuring Dual-Core Intel® Xeon® Processors — is versatile enough to support 32- and 64-bit computing environments. So, take a close look at the HP BladeSystem c-Class — it's next generation technology that deals well with change, so you won't have to.



Experience the HP BladeSystem and download the IDC White Paper "Enabling Technology for Blade I/O Virtualization."



Click [YouAlwaysHadIt.com/strength13](http://YouAlwaysHadIt.com/strength13)  
Call 1-866-625-4085  
Visit your local reseller



Dual-Core is a new technology designed to improve performance of multithreaded software products and hardware-aware multitasking operating systems and may require appropriate operating system software for full benefit; check with software provider to determine suitability; not all customers or software applications will necessarily benefit from use of this technology. Requires a separately purchased 64-bit operating system and 64-bit software products to take advantage of the 64-bit processing capabilities of the Dual-Core Intel Xeon Processor. Given the wide range of software applications available, performance of a system including a 64-bit operating system will vary. Intel's numbering is not a measurement of higher performance. Intel, the Intel Logo, Xeon and Xeon Inside are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries. The information contained herein is subject to change without notice. ©2007 Hewlett-Packard Development Company, L.P.