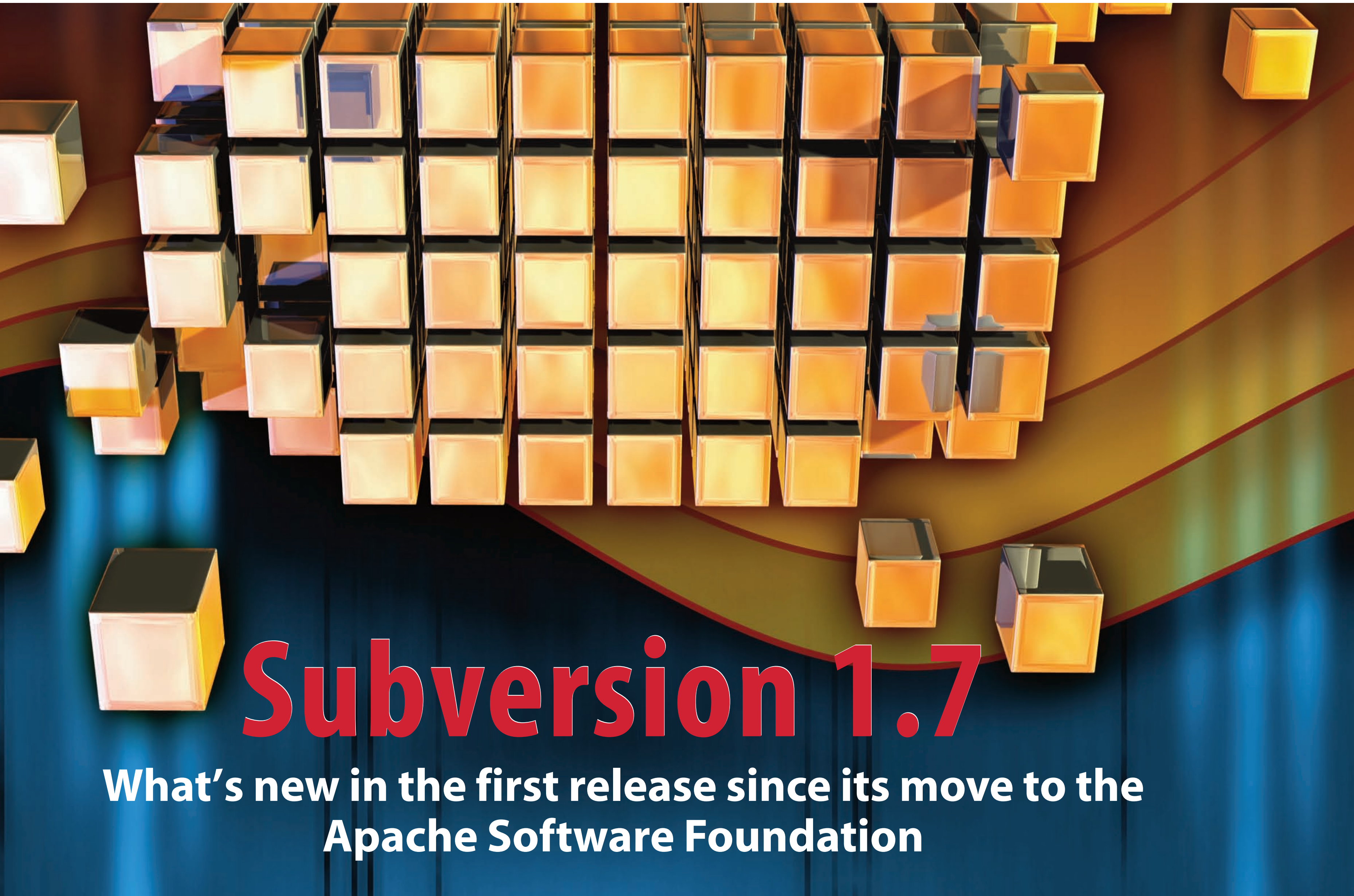
Next

Dr. Dobb's Journal

SEPTEMBER 2011



Subversion 1.7

What's new in the first release since its move to the Apache Software Foundation

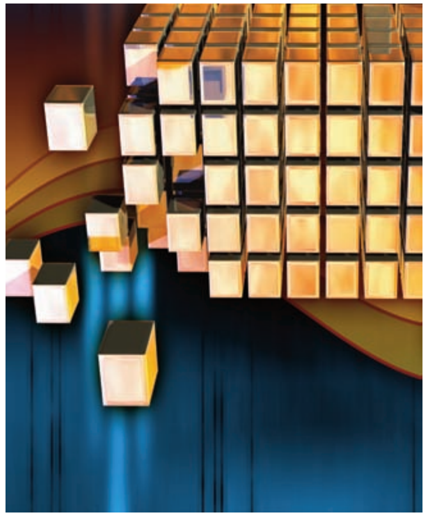
ALSO INSIDE

UBM TechWeb

www.drdobbs.com

Dr. Dobb's Journal

CONTENTS



September 2011

More on DrDobbs.com

Jolt Awards: The Best Books

The six best books every developer should read.

http://drdobbs.com/joltawards/231500080

3 Most Useful Library Features of C++0x

Moving data without copying it, smart pointers, and lambda expressions.

http://drdobbs.com/cpp/231300394

Sloppy Work at Oracle

Andrew Binstock believes poor testing and bad decision-making have marred an important release.

http://drdobbs.com/java/231300060

More Uses for *auto*

This post continues an ongoing discussion of the C++ *auto* language feature by showing some contexts in which it makes programs easier to read.

http://drdobbs.com/blogs/cpp/231300549

[Dude|Sir], That's Not Programming

Every year or so, academic publications in our industry indulge in a bit of mental torpor by running an article on the question of whether computer science is a real science.

http://drdobbs.com/joltawards/231500038

Parallelism As a First Class Citizen in C and C++

According to James Reinders, we need mechanisms for parallelism that have strong space-time guarantees, simple program understanding, and serialization semantics — all things we do not have in C and C++ today.

http://drdobbs.com/go-parallel/blogs/cpp/231500025

IN THIS ISSUE

Mailbag

The capabilities of programmers and the quest for security

[Dude|Sir], That's Not Programming

This editorial (http://drdobbs.com/joltawards/231500038) about what is and is not programming evoked numerous extended and sometimes passionate letters.

This really hit home for me. Domain experts either are instructed or take it upon themselves to write code; when they leave and/or the resulting code becomes too difficult to maintain, that's when a programmer or team of programmers is brought in.

The business people, engineers, and so on object that they need someone who understands the problem domain. I would heartily agree, and I have always argued that a good developer should be somewhat interdisciplinary, immersing him- or herself in whatever industry s/he happens to be working in.

But I would counter that while developers need some understanding of the application domain, they don't need to be expert in order to develop software for it. Also, unless the software fills only a one-time need — and we know from experience that there are very few real one-off applications — businesses need people who understand the art of programming.

But beyond that, there seems to be a curious sort of cognitive dissonance at work here. The assumption is made that the domain experts can easily pick up whatever programming knowledge they need. I find this curious for two reasons: 1) It doesn't seem to occur to anyone that the inverse may be true — i.e., that developers can probably pick up the domain knowledge they need with relative ease. 2) This implies a dismissive attitude toward software development; but if, later on, one or more programmers are brought in to fix or maintain the software, the arrival of the programmers is treated as a sort of "magic bullet" … an interesting contrast, to say the least.

—David Lindsley

Programming Languages empower individuals to help make computers do useful things (useful is obviously in the eyes of the beholder). I agree with most of what you said, but, for example, does C++ just empower "application" development in terms of better abstracting "objects"? Twenty years ago, assembly language programmers were saying, "Eh, C++ programmers aren't real programmers, they are so far from the hardware, they don't even know what registers are in the

IN THIS ISSUE

computer they are targeting," using similar arguments you make in your article. (Similar arguments were made with us C++ programmers vs. the old VB programmers back 15 years ago).

I think we have to be cautious of "where to draw the line." One man's HTML is another man's assembly language. One man's abstraction is another man's implementation detail.

—Mark Franjione
Host Engineering, Inc.

Good software engineering has two prerequisites. First, there must be a certain amount of inherent talent for computers and software. Those with a bit less than others need to try harder, but you must have some degree [of] innate understanding. Given that, the most important requisite is attitude. You must understand that it ain't easy and there are no short cuts. You must care about what you create.

Specific job skills are often important, but never as important as attitude.

—Bryan Kelly

Encrypt Early, Encrypt Often

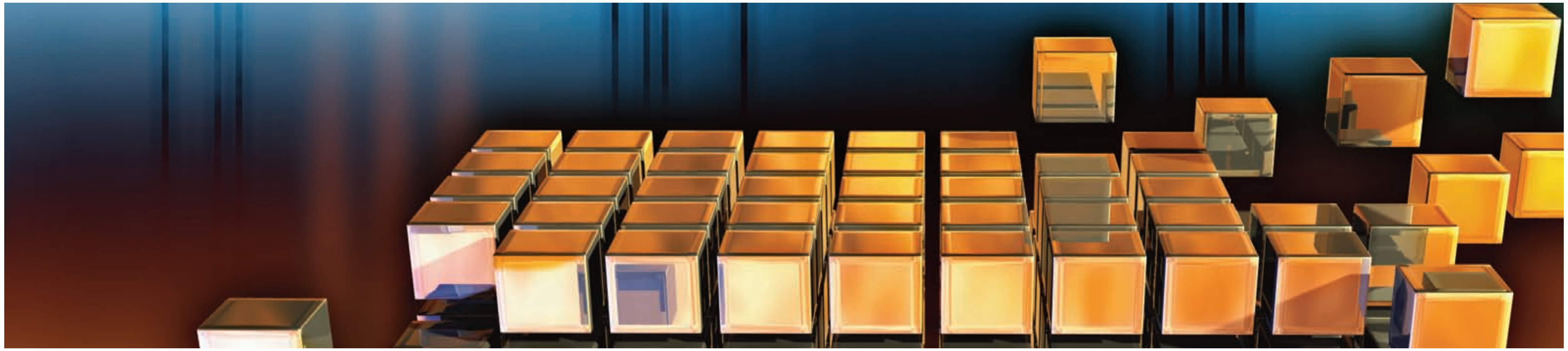
In response to a recent editorial on data security in the cloud (http://drdobbs.com/security/231300517).

No data can be safe, even if they are encrypted, as long as governments think we are all wannabe terrorists. That is, unless we encrypt our data with our own algorithms, but how expensive would that be?

—Fotios Fotinakis

Andrew Binstock replies: "You might be right, but some open-source solutions can be shown to have no back door. I recall that some European OSS projects specifically worked on this because of their fear of the U.S. back doors. But I agree the problem is indeed difficult. It is difficult to be 100% sure, and of course, writing your own is no remedy as it's unlikely you're going to be able to match the strength of today's cryptographic tools."

Have a correction or a thoughtful opinion on *Dr. Dobb's* content? Let us know! Write to Andrew Binstock at alb@drdobbs.com. Letters chosen for publication may be edited for clarity and brevity. All letters become property of *Dr. Dobb's*.



Parallel Microsoft-Style

The actor model of concurrency is gaining favor in Java but remains largely ignored by Microsoft

One area in which Microsoft operating systems have long been on the vanguard is multithreading APIs. Even before multicore processors were the norm, Microsoft offered expansive APIs for managing threads and thread-like execution streams called fibers. The Win32 platform had a wide selection of thread-management functions that could be used with considerable ease. While I can't speak for every operating system, that collection of APIs was considerably greater and more sophisticated than UNIX or Linux counterparts. Pthreads, the now-standard threading implementation on Linux, is a much smaller solution. However, Pthreads does provide a benefit singularly lacking in the Windows implementation: portability. Pthreads originated on UNIX where it still runs on the various variants. A good port on Win32, (http://sourceware.org/pthreads-win32/) exists, in addition to the multiple Linux distributions.

The Microsoft APIs, however, are very much drawn from the traditional approach to threading, which is a complex model of creating threads, managing them individually, and using mutual exclusion to prevent them from interfering with each other. Even though Microsoft's Visual Studio IDE has decent support for working with threads, developers can expect to spend a lot of time in the debugger. In this shared-memory model, the two great debugging challenges are replicating the problem, and once it's replicated, diagnosing it.

The issue of replication has driven more than one developer to the brink of insanity, if not pushed him over the edge entirely. Some errors show up only when certain threads interact in a rare way, with each one handling a specific piece of data. This might show up as a defect report indicating that, occasionally, transactions aren't logged correctly to the database. Trying to recreate what conditions might lead to an event transiently not occurring — when dozens of threads are in various states of operation — can be exceedingly difficult, if not impossible. But even if the situation can be faithfully reproduced, trapping the condition and determining how the threads are crossing each other incorrectly is arduous. In sum, the shared-memory model is no one's idea of

IN THIS ISSUE

programming fun. And as a result, much software that should have been parallelized long ago runs serially because the job is constantly deferred until parallelization is the *only* way to gain the needed performance.

Various attempts have been made to remove this complexity and thereby facilitate parallel development. One of the strongest pushes has come from Intel, which spearheaded OpenMP (http://drdobbs.com/214600012) a decade ago. OpenMP sought to add keywords to the C (via pragmas) and Fortran , so that portions of existing programs could be parallelized. This approach, while limited to only sections of the program, worked well. It was easy to grasp and it cleverly handled the messy thread-management issues behind the scenes. If the code ran correctly on a single thread, chances were excellent that it would still run correctly when parallelized with OpenMP keywords. Microsoft's support of OpenMP made it a natural choice for many applications.

More recently, Intel has come out with a more elaborate solution, Cilk Plus, which follows the OpenMP model and leverages Intel's deep knowledge of threading (acquired from a decade of investment in threading tools). James Reinders of Intel provided an excellent technical overview of the problem and solution as viewed from within the industry in a recent blog post (http://drdobbs.com/go-parallel/blogs/cpp/23150002).

Java and the Actor Alternative

Interestingly, the languages on the JVM are heading towards a different kind of solution, namely actors. If you're not familiar with actors, it's easiest to think of them as functions to which discrete tasks can be sent for execution. The most common operational metaphor is a thread that has an incoming mailbox to which messages are sent by other actors. The actor processes these messages (which can include tasks to execute or data to operate on) and the output is sent to another actor for downstream work.

"For all the extensive threading APIs Microsoft offers, the company has not substantially embraced actors"

Actors avoid the dangers of the shared-memory model because they touch only the data that's sent to them in messages. If they need some external data, they request it from other actors (by sending the request in the form of a message). In addition, the data actors receive is immutable. They don't change the data, rather they copy it and transform the copy. In this way, no two threads are ever contending for access to the same data item, nor are their internal operations interfering with one another. As a result, the nightmares I described earlier disappear almost completely.

The actor model has been used in applications that naturally align with message passing: telephone switches, Web servers, and the like. Increasingly, however, JVM languages are using them for run-of-the-mill applications. Obviously, actors require a different kind of architecture (in most actor applications, everything is done by actors — mixed-model applications are generally not favored.). And in support of this, they are finding support in several recent languages. For example, Scala and Fantom have built in-actor libraries (Scala's enjoys native syntactic support). Groovy has a widely used actor library. And actor frameworks, such as Akka and Killim, that can be used by any JVM language have recently come to market.

For all the extensive threading APIs Microsoft offers, the company has not substantially embraced actors. A good start was made in 2010 when the Redmond giant released Asynchronous Agents Library (http://is.gd/7w4eIV), but this is only a partial actor implementation and it runs only in C++. Due to its infancy, there is no long history of working with this technology — nor any evidence that I can find of support for actors in the .NET languages. Microsoft is still primarily oriented towards the traditional models of parallel programming. The benefit of this approach is raw performance. For the same reasons, Intel's libraries target only C, C++, and Fortran — three languages that are associated with high-speed computation and frequently run as native code.

Application developers who are willing to trade a little performance for ease in writing parallel applications are likely to find Java's embrace of actors an appealing solution (although Java certainly has the traditional primitives for developers who want to follow that path.)

I expect that over time, Microsoft will increase its support for the actor programming model. But if it comes, this support will likely appear later rather than sooner — and will probably be driven by competitive pressures from without rather than internal corporate preference.

— Andrew Binstock is Editor in Chief for Dr. Dobb's *and can be contacted at alb@drdobbs.com.*

Comment

What's New in Subversion 1.7

A completely redesigned working copy library speeds common operations, streamlined communication between clients and servers, and much more

By C. Michael Pilato

The Apache Subversion project recently announced the general availability of the eighth major release of its ubiquitous open source version control system. Subversion 1.7.0 is the first release from the project since its celebrated move into the Apache Software Foundation's famous family of mission-critical software projects. Now, after baking for a couple of years, the oven timer has finally rung. So just what's being served here?

WC-NG

The single largest feature in the 1.7 release isn't even really a feature in and of itself, but is instead a major plumbing overhaul. Subversion ships as a collection of well-defined libraries and APIs, with tooling (such as the "svn" command-line client) to make it all usable. The oldest of those libraries is the so-called working copy (WC) library, which is responsible for tracking enough client-side information about your version-controlled files so that Subversion can easily detect, examine, revert, and publish the changes you make to those files. As Subversion's data model and feature set matured over the past decade, this library grew in a somewhat piecemeal fashion, eventually evolving into a messy chunk of code that was hard to maintain and even harder to cleanly enhance.

Subversion 1.7 delivers a completely redesigned working copy library, universally referred to as "WC-NG." This work is the single defining feature of the release. As is often the case with such code overhauls, the new library implementation flies somewhat under the radar from the user's perspective. But while the original intent was to deliver a new

IN THIS ISSUE

foundation for future improvement, Subversion's developers — and by extension, its users — are already reaping a harvest of immediate, tangible benefits from this work.

The biggest benefit that WC-NG brings is enhanced performance. The original WC library distributed the working copy's administrative metadata throughout the directory hierarchy. Because of this, most Subversion operations had to walk the entire directory tree — often, multiple times — just to gather the required information about, and apply changes to, the working copy. WC-NG instead centralizes all this information in a single database, so most of those slow directory walks have been completely eliminated. On operating systems with less-than-stellar filesystem performance (such as Microsoft Windows), the difference is notable, with some users reporting that their most common operations are faster by an order of magnitude!

Besides the improvements in speed, WC-NG has already allowed the Subversion developers to fix a large number of known defects. There are far too many to list here, but there is one that certainly deserves a mention due to its commonness: case-only renames. Prior to 1.7, Subversion failed when users on case-insensitive platforms (such as, again, Windows) tried to rename a file in such a way that the old name and new name differed only in the letter casing, e.g., from "myfile" to "MyFile." This ancient bug is now fixed — and a slew of additional bugs along with it!

The benefits don't stop with the core Subversion distribution. Providers of third-party Subversion clients and IDE integrations (AnkhSVN, Subclipse, TortoiseSVN, etc.) report that WC-NG's capabilities are already enabling the addition of new features to those tools. It might have been a thankless plumbing overhaul, but it would appear that WC-NG's new pipes terminate at a virtual Fountain of Youth for the Subversion project.

Revised HTTP Protocol

Another important-yet-mostly-invisible enhancement made in Subversion 1.7 is best described not so much by what the software now does, but by what it no longer tries to do. You see, Subversion was born back in a time when the Web-based Distributed Authoring and Versioning (WebDAV) protocol was relatively new. As new protocols tend to do, WebDAV — specifically, the Delta-V extension thereof — made big promises. The possibilities of a standards-driven version control protocol were mind-boggling: can you imagine using your Subversion client to communicate with a ClearCase VOB server? So Subversion set off into the exciting frontier of WebDAV/Delta-V, using the Apache HTTP Server and its reference implementation of WebDAV as the primary building blocks of the first Subversion server.

Unfortunately, Delta-V failed to catch on. Subversion's client/server communication was, as a result, unnecessarily chatty and abstract. So the Subversion developers officially gave up on the dream of interconnectivity and decided to put its HTTP communication protocol on a strict diet. Subversion 1.7 delivers more streamlined communication between Subversion clients and servers while still preserving compatibility across all released Subversion versions to date. Subversion 1.7 also preserves the parts of its WebDAV/Delta-V implementation, which remain beneficial to users — such as the so-called "autoversioning" support, which allows Subversion repositories to be mounted as fully operational remote WebDAV filesystem shares.

Merge Tracking Improvements

Every major release of Subversion (and many minor ones) since version 1.5.0 delivered incremental improvements to Subversion's merge tracking feature, and this release is no exception. Since the 1.6.0 release, Subversion's developers have identified and fixed nearly 80 bugs and enhancements related to Subversion's ability to efficiently and correctly track and manage merges of changes between branches. Many of those fixes have already been released to the public via the subsequent 1.6.x releases, but Subversion 1.7.0 alone provides more than 40 of these improvements.

One of the key improvements is the handling of merge tracking metadata on working copy paths under a merge target (known as "subtree mergeinfo"). No longer does a merge unconditionally update all subtree mergeinfo to describe a merge. In 1.7 only the subtrees affected by the merge are updated. This addresses a common complaint about the amount of "noise" in the form of mergeinfo changes produced by relatively simple merges.

With Subversion's widespread (and growing) adoption in enterprise environments — many of which employ heavily branch-centric development processes — every improvement to this feature is a welcome one.

Improved Difference Marshaling

Experienced Subversion users are familiar with Subversion's built-in "diff" functionality, which displays line-based changes between two files or two versions of the same file. This information is, of course, useful to the humans reading it ("What changes did I make to this file's contents?"), but is also presented in the standardized unidiff format, which can be stored in a patch file, shared with another user, and then parsed by a "patch" program in order to apply those same changes to a different copy of the file. Unfortunately, "patch" is concerned only with content changes to text files. It knows nothing of the other types of changes that can be represented in Subversion working copies: changes to file and directory versioned properties; changes to the directory hierarchy caused by adding, removing, and copying items around; changes to the contents of non-human-readable files; and so on.

Subversion 1.7 offers some improvements to its "diff" functionality, enhancing the output format to represent changes to versioned properties in a more reliably machine-parseable way and offering an output mode that is compatible with the popular git version control system. But those improvements alone don't necessarily help other Subversion users apply someone else's patch files to their own working copies. So Subversion 1.7 also introduces built-in support for parsing its own "diff" output and applying the changes found there to a target working copy. Today the new "svn patch" command can parse and apply content modifications made to human-readable files, modifications to Subversion file properties, and simple file additions and deletions. Future releases of Subversion are expected to deliver the completion of this feature, ultimately allowing the ability to reproduce every type of local change that Subversion allows in its working copies.

... And Then Some

It goes without saying, but good developers can fix a tremendous number of bugs in two years' time. The CHANGES file provided with the source code distribution of Subversion 1.7 lists over 150 improvements to the software, and those are just the ones deemed worthy of mention. And while this article covers the most high-profile features of the release, that coverage is far from exhaustive. Subversion 1.7 offers many more improvements, perhaps one of which is precisely the one you've been waiting for. For more information, consult the project's official release notes online at http://subversion.apache.org/docs/release-notes/1.7.html.

So, What's In It for You?

If you've read through to this point, you've learned about Subversion 1.7's primary offerings in some technical detail, but may find yourself still in need of the "big picture". And if you've only skimmed subtitles thus far, that's understandable — time is a precious resource. So let's summarize Subversion 1.7. What's in it for you? Why is now the time for you or your organization to embrace Subversion for the first time? Why should you or your organization upgrade your existing Subversion deployments to the new 1.7 version as soon as you can?

- **Protection.** Subversion has always been a passionate guardian of your versioned data. It's a well-established fact that your data is safe with Subversion. But over the years, Subversion's devel-

opers have learned many lessons about how users can sometimes do things that are not in their best interests. Subversion 1.7 brings several changes aimed at preventing — or at least discouraging — users from accidentally complicating their day-to-day lives and workflows. You'll have an even better experience with Subversion overall as the software encourages your adherence to established best practices.

- **Performance**. By and large, the Subversion 1.7 release is about addressing some long-standing complaints about the software's speed, especially on the Windows platform. In today's network-centric world, users understand that there are real costs involved in transferring files to and from a remote server. They "get" that. But they reasonably expect local operations — or the locally handled portions of remote operations — to perform much better than they historically have in Subversion. Subversion 1.7 brings noticeable improvement in this area.
- **Permanence**. Subversion took the version control world by storm before it even reached its 1.0 release. It's everywhere today. But more importantly, it's not going away tomorrow. CollabNet — the company that founded the project over a decade ago and was its sole corporate sponsor for many years — is now joined by other companies who bring additional resources to the developer community and additional benefits to the Subversion user base. The project's move to the Apache Software Foundation has invited the attention of still more developer volunteers. Project activity continues to trend upward. In many aspects, the Subversion project has never been healthier.
- **Potential**. Subversion 1.7 demonstrates the benefits of developers' willingness to revisit some of the software's inner workings. It ain't sexy work, but the results are worth it. And the benefits won't stop here. There are already a number of features planned for future versions that wouldn't be possible without the sorts of plumbing changes previously described. As the project continues to mature, users will see Subversion — and the expanding ecosystem of Subversion-related tools and services — evolving to fill more and more of their needs.

If you're not using version control as part of your development processes, Subversion could very well be the difference between lost assets and the safekeeping of your developers' hard work. If you've already embraced version control in general, but are shelling out big bucks to your version control vendor, Subversion — which is free — could drastically reduce your operational costs. And if Subversion is your current version control solution, Subversion 1.7 promises your best Subversion experience to date, with the necessary groundwork laid for still more improvement in the future. You can download Apache Subversion 1.7 from http://subversion.apache.org/packages.html.

— C. Michael Pilato is a core Subversion developer and the coauthor of **Version Control with Subversion** *(http://is.gd/lJ5bKl). He also authors the C-Mike Run blog at http://cmpilato.blogspot.com/.*

Comment

Automating An Eclipse RCP Build

Building an Eclipse RCP project can be a complex undertaking. Some new tools make it a lot easier.

By Micah Hainline

Since its release, the Eclipse Rich Client Platform (RCP) has suffered from a lack of support for automated builds with tests. The IDE provides tools for building and packaging RCP applications, but they are very different from the PDE/Build tools that perform these tasks in an automated fashion. Also, until recently, testing support for JUnit 4 was not present in the Eclipse Test-Framework. While the situation has markedly improved in Eclipse 3.6, setting up an automated build with tests remains a time-consuming and difficult task.

To make the job easier, I've created a minimal project called "Lightning," which demonstrates how to combine the new automated build tool Buckminster, the Eclipse Test Framework, and the integration testing tool SWTBot. Lightning has been tested on Linux and Windows. (Buckminster on the Mac platform has a few kinks to work out.)

Loading Lightning

Buckminster (http://www.eclipse.org/buckminster/) is a build system for Eclipse products and plug-ins. It can be run from within the IDE (with the proper tooling installed) or from the command line. I use the command-line version in Lightning, so no additional IDE tooling is required. Buckminster can inspect artifacts such as plug-ins and features in order to determine their dependencies.

I launch Buckminster with Apache Ant (http://ant.apache.org/), which manages all the other tools required by Lightning. Ant provides the top-level targets and everything you need to run the build from Eclipse, the command line, or a continuous-integration server like Jenkins.

Since the command-line distribution of Buckminster is hard to install manually, the Lightning build process installs it automatically using the P2 director application from the Buckminster download site (http://www.eclipse.org/buckminster/downloads.html). P2 is the provisioning system in Eclipse, replacing the old update manager.

The first step in building the Lightning project is to download it from github (https://github.com/micahhainline/lightning) and unzip it, or just clone the project using git.

Because this build uses Buckminster, you can actually download and assemble the target platform for the project from the Helios P2 repository automatically — which is called "materializing" in Buckminster parlance. To do so, simply navigate to the "Buckminster.build.feature" directory and run the command `ant materialize.target.platform`, which will materialize the target platform to "<unzip location>/workspace-target-platform." This directory will serve as the target platform for the IDE workspace. Next, import the Lightning plug-ins into a new Eclipse 3.6 workspace and instruct it to use the target platform you just materialized.

Building the Application

You can kick off the full build process by running the command `ant` from the "buckminster.build.feature" directory.

The build process starts by installing P2. This, in turn, installs the command-line version of Buckminster. The next step uses Buckminster to build the Eclipse RCP products.

Two products are created during the build process. The first is the test product, which includes Lightning along with all tests and test dependencies. The second is the production product, which includes only the Lightning application. This is the version that will be shipped to customers when the time comes. If it seems a little strange that two products are needed, just remember that the tests must be run inside an OSGi environment, and this means that they need to be packaged into a fully functional Eclipse RCP application.

When Buckminster is used to build the Lightning products it runs in an Eclipse workspace, just as it would if it were being run from within the IDE. The first step in building the product, therefore, is to set up the workspace and associated target platform. This gives Buckminster a sandbox in which to play. The target platform is imported using a target definition file, which tells Buckminster where to place any plug-ins it later downloads for the target platform. Next, the `import` command executes an import of those plug-ins as defined in a .cquery file (Listing One).

Listing One: The structure of the .cquery file.

```
<cq:componentQuery
xmlns:cq="http://www.eclipse.org/buckminster/CQuery-1.0"
                 resourceMap="target.platform.rmap">
    <cq:rootRequest name="buckminster.build.feature"
        componentType="eclipse.feature"/>
    ...
</cq:componentQuery>
```

The .cquery file points to an Eclipse feature that I have created especially for Buckminster, and also includes a list of places to find the plug-ins included in that feature. This list is stored in an .rmap file, part of which I include in Listing Two.

Listing Two: The .rmap file.

```
<rm:rmap xmlns:rm="http://www.eclipse.org/buckminster/RMap-1.0"
 xmlns:bc="http://www.eclipse.org/buckminster/Common-1.0">
 <rm:searchPath name="helios">
   <rm:provider componentTypes="osgi.bundle,eclipse.feature"
     readerType="p2" source="false" mutable="false">
     <rm:uri format="http://download.eclipse.org/eclipse/
         updates/3.6" />
   </rm:provider>
 </rm:searchPath>
 <rm:searchPath name="local-repository">
   <rm:provider componentTypes="osgi.bundle,eclipse.feature"
     readerType="p2" source="false" mutable="false">
     <rm:uri format="{0}local-repository">
       <bc:propertyRef key="build.feature.dir.url" />
     </rm:uri>
   </rm:provider>
 </rm:searchPath>
 <rm:searchPath name="local-source">
   <rm:provider componentTypes="osgi.bundle,eclipse.feature"
     readerType="local" source="true" mutable="false">
     <rm:uri format="{0}{1}">
       <bc:propertyRef key="product.checkout.url" />
       <bc:propertyRef key="buckminster.component" />
     </rm:uri>
   </rm:provider>
 </rm:searchPath>
 <rm:locator searchPathRef="local-source" failOnError="false" />
```

```
  <rm:locator searchPathRef="local-repository" failOnError="false" />
  <rm:locator searchPathRef="helios" failOnError="false" />
</rm:rmap>
```

"You could set up the build to use the workspace-target-platform created earlier, and the process would probably be a bit faster. But I wanted to demonstrate how Buckminster can assemble a platform on the fly."

The .rmap file lists the locations of the plug-ins that ultimately make up the application. These include public P2 repositories from the Internet, local P2 repositories, and local source folders. The local P2 repositories were created earlier in the build process using the P2 director application and a list of previously built plug-ins. To reduce traffic to the remote P2 repositories, it is a good idea to create mirrors of the P2 repositories on the local network (though this is not required).

Once the .cquery is imported, Buckminster has a fully materialized target platform, a workspace, and the required projects for Lightning. It is now ready to build the projects. Note that you could set up the build to use the workspace-target-platform created earlier, and the process would probably be a bit faster. But I wanted to demonstrate how Buckminster can assemble a target platform on the fly.

Packaging the Project

After the build compiles all of the source code in the workspace with Buckminster, it executes a Buckminster target to package the result. When building an Eclipse feature using Buckminster, one would ordinarily package the result into a P2 repository using the automatically generated "site.p2" target. In this case, however, I need the plug-ins packaged as a product with a launcher. So I have created a different target in a cspex file (Listing Three).

Listing Three: The cspex file.

```
<cspecExtension
xmlns:com="http://www.eclipse.org/buckminster/Common-1.0"

xmlns="http://www.eclipse.org/buckminster/CSpec-1.0">
    <actions>
      ...
        <public name="create.product.test" actor="ant">
            <actorProperties>
                <property key="buildFile" value="product.ant" />
                <property key="targets" value="create.product" />
            </actorProperties>
            <properties>
                <property key="profile" value="Profile" />
                      <property key="iu" value="com.asolutions.
                                        lightning.test_product" />
            </properties>
            <prerequisites alias="repository">
                <attribute name="site.p2" />
            </prerequisites>
            <products alias="destination" base="../../">
                <path path="test-product/" />
            </products>
        </public>
            ...
    </actions>
</cspecExtension>
```

The cspex file allows me to add custom targets to the built-in targets provided by Buckminster. The new target calls site.p2 and then installs the product into the file system. In this case, the build installs the test product with the command `create.product.test`. (This command

IN THIS ISSUE

actually runs the embedded version of Ant that comes with Buckminster to perform the work.)

Running the Tests

After the test product is installed in the file system, the build needs to run the tests. The Eclipse Test Framework includes an Ant file for this, but since it is a little limited, I have replaced that functionality with a custom Ant file. Because the test framework is launched as an Eclipse application and can run only a single test or test suite at a time, a test suite called "AllTests" has to be included for each plug-in fragment.

The test target in Ant launches the test framework from within the test product. It takes the name of the plug-in that contains the tests, and then saves the resulting JUnit test reports. Ant then invokes the SWTBot framework (http://www.eclipse.org/swtbot/), which runs user-interface tests and saves the resulting reports. These test reports can be displayed by the Eclipse IDE. Assuming all tests pass, Ant packages the final product and the build is complete.

An automated build is one of the key units in any professional development effort. With it, developers can have confidence in the product. As a bonus, it allows the use of a continuous integration server (such as Jenkins; http://jenkins-ci.org/) to keep the team on track. With the advances made in Eclipse 3.6 and the Lightning template project, an automated build for Eclipse RCP is finally within the reach of the whole community.

— Micah Hainline is a software engineer at Asynchrony Solutions Inc. (http://blog.asolutions.com). He works on mobile applications, Web technologies, rich client applications, and enterprise architecture.

Comment

From the Vault

Algorithmic Trading

This entry from *Dr. Dobb's* ran in March 2007 and showed how patterns in real-time market data can be recognized and responded to in order to detect trading opportunities and place and manage orders in the market.

— DDJ

By John Bates

A**nyone with a compute**r science background is familiar with the concept of algorithms for tasks such as the searching or sorting of data. However, what about algorithms that specify a sequence of steps to make money in the capital markets? That's exactly what the latest excitement around "algorithmic trading" is all about.

In the algorithmic trading space, an "algorithm" describes a sequence of steps by which patterns in real-time market data can be recognized and responded to in order to detect trading opportunities and place and manage orders in the market. The term "algorithmic trading" has only become commonly used within the financial sector over the past few years — although trading algorithms have been around for longer. Historically, large investment banks have deployed armies of Ph.D.s to custom build trading algorithms. Now, an advanced technology approach called "Complex Event Processing" (CEP) is making it much quicker and easier to build, deploy, and manage trading algorithms, with fewer personnel necessary.

What Do Algorithms Replace?

Before the days of automated algorithms within the financial markets, traders manually carried out the process of building and managing a trading strategy. Sitting at specialized trading stations, with four or eight screens, traders watched as real-time market data changed. By manually tracking analytics and patterns, possibly in a spreadsheet, traders worked out when and where to place orders into the market and then managed these orders to see if they were fulfilled. The trader understood the workings of the algorithm, but each step was a manual process. Now, apart from initiating a particular algorithm, the trader does not have to be involved at all. In most cases, the trader will monitor the algorithms using a graphical dashboard. In fact, a trader can now initiate and manage hundreds, or even thousands, of independent algorithms — as opposed to doing one thing at a time manually. This way, the trader is scaled to become much more productive. However, as will be discussed later in this article, the algorithm doesn't replace the trader. It is the trader and his or her team of quantitative analysts who devise new algorithms and tailor existing ones. What Makes Trading Algorithms Possible?

Trading algorithms have been made possible by the open availability of electronic APIs, to enable connectivity to exchanges and other electronic trading venues. In equities and some futures trading, exchanges provide a centralized venue to buy and sell stocks and futures. Foreign exchange is similar, but there are many more independent electronic venues, rather than a centralized exchange (this is called an over-the-counter, or OTC, market). Streaming data can be received by connecting directly to the trading venue or through an information provider, such as Reuters. The streaming data represents the changing prices and availability of instruments on the venue's order book. It is also possible to send orders into the venue's order book, thus enabling buying and selling at an available price, or registering a bid or offer at a certain limit.

Giving an algorithm access to these multiple APIs enables it to watch the changing market data and place orders when certain desirable levels are met.

What Do Trading Algorithms Do?

There are a variety of algorithms in common use within the financial industry. However, the battle for supremacy in algorithmic trading exists in the creation of new and bespoke algorithms. The aim is to develop the most profitable algorithm at the expense of all others.

The two main parts of a trading algorithm are sequences of steps determining when to trade and how to trade.

Determining when to trade is a decision that revolves around watching the changing market data and detecting opportunities within the

ALGORITHMIC TRADING AND STREAM PROCESSING

by William Hobbib

On Wall Street and other global exchanges, electronic-trading data feeds can generate tens of thousands of messages per second, and latencies of even one second are unacceptable. Consequently, technologies such as "stream processing" have been developed to address the challenges of processing high-volume, real-time data. Stream processing enables parallel processing of a specified series of operations on multiple data streams with high levels of efficiency and performance. It is being implemented on applications ranging from financial trading to computer gaming. StreamBase Systems (www.streambase.com), for instance, provides a Stream Processing Engine leveraged by investment firms and hedge funds in areas like algorithmic/automated trading, risk management, transaction cost analysis, and compliance management. One requirement for streaming applications is that they be capable of storing and accessing current or historical state information, preferably using familiar SQL-type commands. Storage of state data is almost universally desired. In addition, for many situations, events of interest depend partly on real-time data and partly on history. An extension of this requirement comes from firms that want to test a trading algorithm on historical data to see how it performs, then test alternative scenarios. When the algorithm works well on historical data, it can be seamlessly switched over to a live feed without application modification.

— At the time of publication, Bill was vice president of marketing for StreamBase.

IN THIS ISSUE

market. This is the analytic part of the strategy. As an example, consider a "pairs trading" strategy (see Figure 1). This strategy examines pairs of instruments that are known to be statistically correlated. For example, consider Shell and Exxon. Both are oil stocks and so, to a large degree, are likely to move together. Knowledge of this trend creates an opportunity for profit, as on the occasions when these stocks break correlation for an instant, the trader may buy one and sell the other at a premium. This is what a pairs-trading strategy is all about. Here, the algorithm involves monitoring for any changes in the price of either instrument and then recalculating various analytics to detect a break in correlation. This can be calculated, for example, by identifying that the spread between the two instruments has exceeded certain standard deviations (so-called "Bollinger Bands").

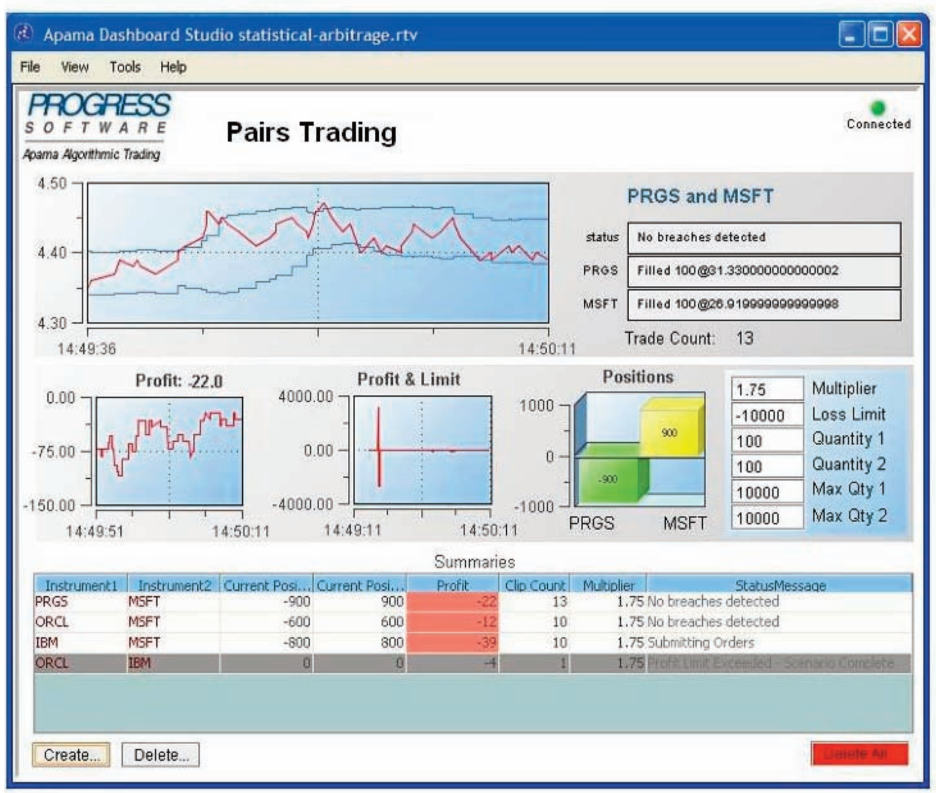


Figure 1: Pairs trading strategy (statistical arbitrage).

Determining how to trade centers on placing and managing orders in the market. As an example, consider a "wave-trading" strategy, which breaks up a large order into smaller orders and places them sequentially into the market over time. The benefit of this is that large orders can get a poor price and can also have a major impact in moving the market overall. Smaller orders are more likely to flow under the market's radar, and subsequently have fewer consequences at a higher level. The wave-trading algorithm simply calculates a number of smaller slices based on trader input and then at prescribed intervals, it places the next wave into the market, barely creating a ripple.

The Algorithmic Arms Race

One risk of using algorithms for trading is that other traders can take advantage of your algorithms if they decipher how they work. Consider a wave-trading algorithm that places large waves into the market every 30 minutes. If a human or algorithm realizes the pattern, they could front-run the orders; that is, they could buy at a discount before the next order causes the price to go up. The more common the trading strategy, the easier it is to reverse engineer. As the uptake of algorithmic trading has increased in recent years, firms are under increased pressure to take measures to ensure their strategies don't fall victim to this practice. During 2006, algorithmic trading entered the mainstream — algorithmic techniques and the technology that powers them are now highly influential in the way that financial instruments, both in exchange and OTC markets, are traded. Prior to this widespread use, firms could gain competitive advantage just by using algorithmic techniques. Today, it is the way in which they use trading algorithms that gives them a competitive edge.

Trading algorithms become interesting — and more effective — when traders combine different techniques in new and complex ways to create unique algorithms that are more difficult to reverse engineer. The markets change everyday and new opportunities continually emerge. It is in the interest of traders to be able to create these new algorithms as quickly as possible, to capitalize on opportunities before their competitors. In this way, algorithmic trading is forming its own "arms race," where the slightest technological advantage can make the difference between substantial profit and loss.

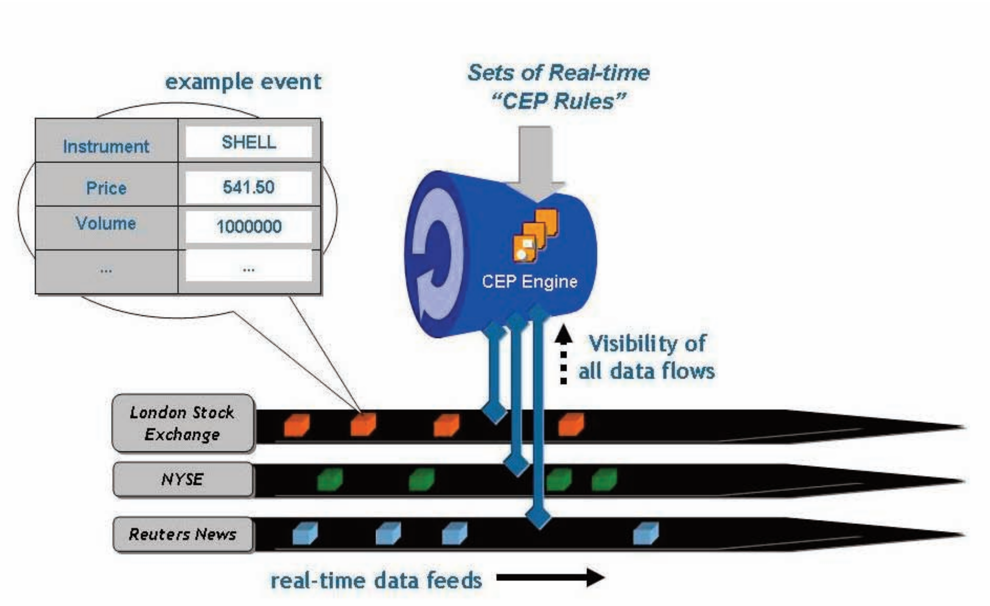


Figure 2: Complex Event Processing (CEP).

Algorithm Creation Using Complex Event Processing

As already discussed, key advantages can be gained by shortening the time between algorithm conception and algorithm implementation. Using traditional approaches, like development in C++ or Java, algorithms can take a long period of time to implement. Sometimes it can be weeks or months before algorithms are integrated into the markets, tested, and put into production. To stay ahead of their competitors, firms require an approach to shorten this time to develop, test, and deploy algorithms.

One promising approach is that of Complex Event Processing (CEP). CEP is a new paradigm of computing that allows organizations to quickly respond to data that is continuously changing. CEP allows firms to monitor, analyze, and act on events in milliseconds. This can have a profound impact on the operations of many businesses, as, in today's market, organizations must deal with exploding volumes of fast-moving data, driven by new modes of connectivity and the demands that this connectivity brings.

Traditional data processing is typically database driven and requires you to store and index the data prior to query. That can be time consuming, particularly for certain apps where responsiveness is crucial to effectiveness. CEP allows you to, in effect, determine the queries in advance by setting certain parameters and then "stream" the data through them, so the relevant data may be selected. This makes it possible to monitor, analyze, and act on those rapidly moving events more quickly — without dependence on a database. This provides a much more time-sensitive response to the events— in effect, responding as they happen.

In trading, CEP takes a new approach to constructing algorithms. It is particularly suited to algorithmic trading; however, it is also suited to many other types of event-driven algorithms. CEP algorithms are structured as sets of event-based rules. These rules monitor items in incoming data streams — termed "events" (see Figure 2). Each event represents an update within the system. An example of a CEP rule is: "When the spread between the price of Shell and Exxon exceeds level *x*, then buy Shell and sell Exxon." Using this approach, complex algorithms can be constructed quickly. An example of a complex CEP rule is shown in Figure 3. CEP rules are hosted inside CEP engines, which efficiently monitor and execute rules. CEP engines can be permanently connected to a wide range of trading venues, so algorithms can be injected into the engine and immediately start monitoring real-time data streams, and take real-time actions on remote services.

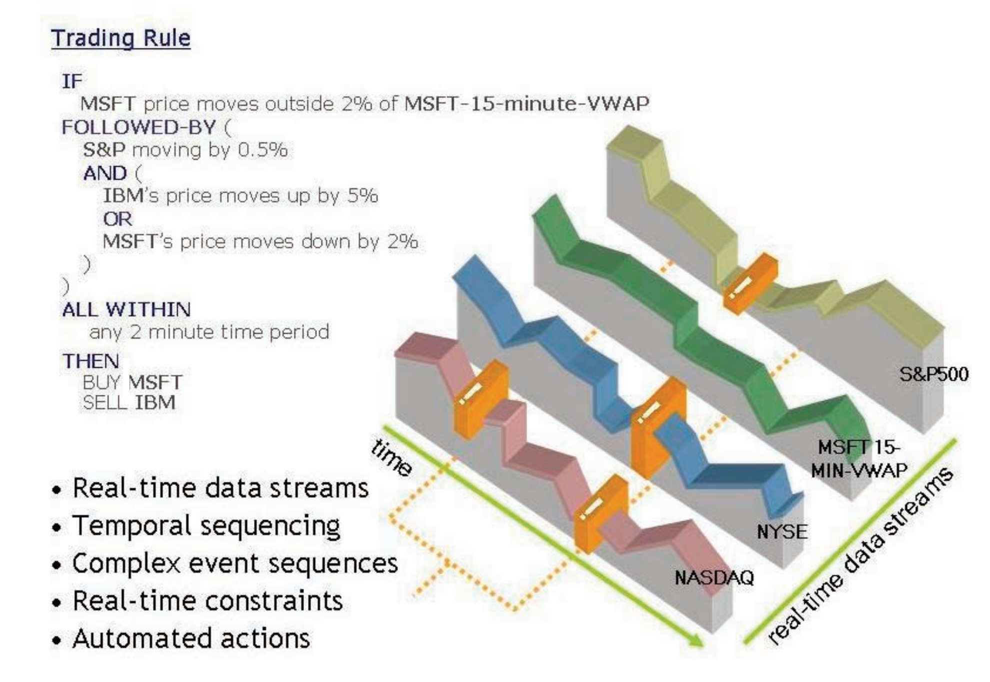


Figure 3: An example of a complex rule and the concepts of CEP.

Graphical Algorithm Modeling

One interesting innovation that is being employed in conjunction with CEP platforms is the ability to implement new algorithms graphically. Graphical programming has always been a challenging area. Using graphical development environments to develop new programs on top of traditional languages, it can take as much time and knowledge as simply typing in the text of the language syntax. However, graphical modeling tools have been very successfully used in conjunction with CEP platforms. Modeling state flow and rules in an event-based system is well suited to graphical abstractions (see Figure 4).

As well as graphically modeling the logic inside their algorithms, today's tools give the traders the ability to visualize, in real time, all runtime activity once their algorithm is running. Real-time "dashboards" can display representations of the changing real-time variables within the algorithms, with automatic alerts when complex conditions or exceptions are detected. Dashboard design studios and runtime rendering frameworks act as a complete design and deployment environment with a wide range of visual objects, including meters, scales, tables, grids, bar and pie charts, along with trend and *x-y* charts — all of which change dynamically as events occur in real time (Figure 1 shows an example of a deployed dashboard). Elements are accessible through a design palette from which the objects can be selected, placed on a visual canvas, and parameterized. This capability removes the reliance on the technical

IN THIS ISSUE

development team traditionally required for the creation and adaptation of trading strategies.

The Future

One question that is occupying the minds of many with an interest in algorithmic trading is: "Will this ultimately replace the trader?" The answer is no — for now. Algorithms have expanded the capabilities of the trader, making each trader much more productive. It still falls to humans to devise new algorithms by analyzing, with computer help, opportunities in the changing market.

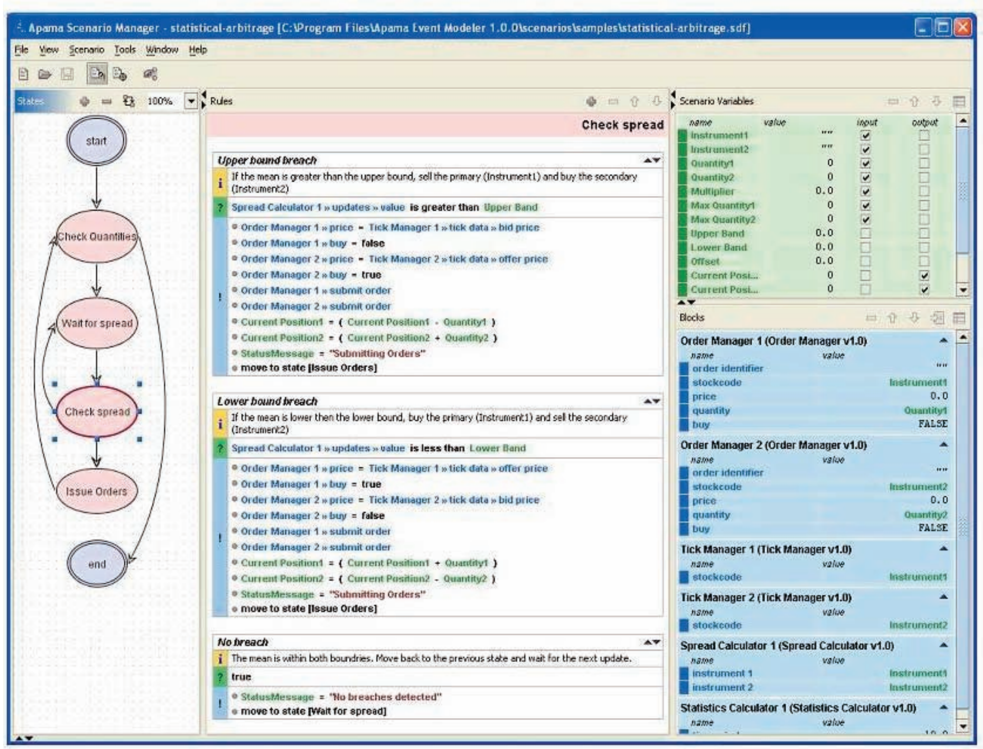


Figure 4: Graphical algorithm modeling using CEP.

Algorithmic trading technology will only begin to replace humans if algorithms are actually devised, developed, tuned, and managed by other algorithms. There are already some techniques being deployed to this end.

One approach is the automatic selection of an appropriate algorithm to use in a particular circumstance, based on events occurring in the market at that point.

Another approach is the use of "genetic" algorithms, whereby a large number (potentially thousands) of variants of an algorithm are created — each with slightly different operating parameters. Each variant can be fed with real market data, but rather than actually trading, can calculate the profit or loss it would be making if it was live in the market. Thus, the most profitable algorithm variants can be swapped live into the market on a continuing basis.

In all of these approaches, Complex Event Processing offers a compelling platform for the creation and management of trading algorithms. The promise of CEP is in providing a powerful platform to enable even the nonprogrammer to encode an event-based algorithm. This year, we will see increased adoption of this approach.

Algorithmic trading is just the first of many exciting applications of CEP — in the financial markets, use in risk management and compliance are the obvious next steps. As we move into the future, CEP will continue to revolutionize trading on the capital markets as we know it.

— At the time of publication, John was the founder and Vice President of Apama Products, Progress Software.

Comment

IN THIS ISSUE

This Month on DrDobbs.com

Interesting items posted on www.drdobbs.com over the past month that you may have missed

LANGUAGE OF THE MONTH: GOSU

A statically typed language with user-defined types? Functions that can be defined anywhere in the class path and applied to all downstream objects? Indeed! Gosu delivers unusual and powerful features.
http://drdobbs.com/open-source/231001429

ASP.NET VS. ASP.NET MVC

Increasingly, companies are choosing ASP.NET MVC for green field development and for significant revisions of existing sites. But does ASP.NET MVC really bring concrete benefits that can be justified?
http://drdobbs.com/windows/231002811

THE FINE ART OF IGNORING STAKEHOLDERS

Stakeholder participation is a cornerstone of Agile. Scott Ambler's survey shows all methodologies involve users equally poorly.
http://drdobbs.com/architecture-and-design/231002934

LINUX FILE SYSTEM MONITORING

Al Williams is struck by how embedded Linux development is often more akin to dealing with a Linux server and not a Linux desktop.
http://drdobbs.com/blogs/embedded-systems/231300313

INTEL VECTOR MATH LIBRARY

The Intel Vector Math Library, also known as VML, is an integral part of the Intel Math Kernel Library. VML provides vector implementations of certain computationally intensive core mathematical functions. Thus, you can use high-level VML vector functions when you want to take full advantage of most modern SIMD instructions without having to write specific SIMD calls.
http://drdobbs.com/go-parallel/blogs/tools/231400052

CREATING YOUR OWN DOMAIN-SPECIFIC LANGUAGE

Users of technical software often want to extend a package's capabilities by writing scripts. To do this, developers need to provide DSL capabilities. This is the first installment in a four-part series showing how to put together an industrial-grade DSL without all the agony of writing a whole new language.
http://drdobbs.com/tools/231002656

JAVA SE 7: COULD IT BE?

Eric Bruno thinks he must be dreaming — Java SE 7 has finally, officially, been released. In this blog, he outlines what the first release of the Java platform under Oracle’s stewardship delivers.
http://drdobbs.com/blogs/java/231002849

IN THIS ISSUE

Dr. Dobb's

Previous Next

IN THIS ISSUE

Dr. Dobb's Business Contacts

DR. DOBB'S

Sales Director, Michele Hurabiell
(415) 378-3540, mhurabiell@techweb.com

Account Executive, Shaina Guttman
(212) 600-3106, sguttman@techweb.com

INFORMATIONWEEK BUSINESS TECHNOLOGY NETWORK

CMO, Scott Vaughan
(949) 223-3662, svaughan@techweb.com

VP of Group Sales, InformationWeek Business Technology Network, Martha Schwartz
(212) 600-3015, mschwartz@techweb.com

Sales Assistant, Group Sales, Kelly Glass
(212) 600-3327, kglass@techweb.com

Publisher's Assistant, Esther Rodriguez
(949) 223-3656, erodriguez@techweb.com

SALES CONTACTS—WEST

Western U.S. (Pacific and Mountain states) and Western Canada (British Columbia, Alberta)

Inside Sales Manager, Vesna Beso
(415) 947-6104, vbeso@techweb.com

Sales Assistant, Ian Doyle
(415) 947-6105, idoyle@techweb.com

Strategic Accounts

Account Director, Sandra Kupiec
(415) 947-6922, skupiec@techweb.com

Account Manager, Shoshana Freisinger
(415) 947-6349, sfreisinger@techweb.com

Sales Assistant, Matthew Cohen-Meyer
(415) 947-6214, mmeyer@techweb.com

SALES CONTACTS—EAST

Midwest, South, Northeast U.S. and Eastern Canada (Saskatchewan, Ontario, Quebec, New Brunswick)

District Manager, Jenny Hanna
(516) 562-5116, jhanna@techweb.com

District Manager, Michael Greenhut
(516) 562-5044, mgreenhut@techweb.com

Account Manager, Cori Gordon
(516) 562-5181, cgordon@techweb.com

Inside Sales Manager East, Ray Capitelli
(212) 600-3045, rcapitelli@techweb.com

Sales Assistant, Elyse Cowen
(212) 600-3051, ecowen@techweb.com

Strategic Accounts

District Manager, Mary Hyland
(516) 562-5120, mhyland@techweb.com

Account Manager, Tara Bradeen
(212) 600-3387, tbradeen@techweb.com

Account Manager, Jennifer Gambino
(516) 562-5651, jgambino@techweb.com

Sales Assistant, Kathleen Jurina
(212) 600-3170, kjurina@techweb.com

Dr. Dobb's

Sales Director, Michele Hurabiell
(415) 378-3540, mhurabiell@techweb.com

Account Executive, Shaina Guttman
(212) 600 3106, sguttman@techweb.com

MARKETING

VP, Marketing, Winnie Ng-Schuchman
(631) 406-6507, wng@techweb.com

Marketing Manager, Monique Luttrell
(949) 223-3609, mluttrell@techweb.com

AUDIENCE DEVELOPMENT

Director, Karen McAleer
(516) 562-7833, kmcaleer@techweb.com

BUSINESS OFFICE

General Manager, Marian Dujmovits

United Business Media LLC
600 Community Drive
Manhasset, N.Y. 11030 (516) 562-5000

UBM TECHWEB

Tony L. Uphoff CEO
John Dennehy CFO
David Michael CIO
Bob Evans Sr. VP and Global CIO Director
Joseph Braue Sr. VP, Light Reading Communications Network
Scott Vaughan CMO
Ed Grossman Executive Vice President, InformationWeek Business Technology Network
John Ecke VP and Group Publisher, Financial Technology Network, InformationWeek Government, InformationWeek Healthcare
Martha Schwartz VP, Group Sales, InformationWeek Business Technology Network
Beth Rivera Senior VP, Human Resources
David Berlind Chief Content Officer, TechWeb, and Editor in Chief, TechWeb.com
Fritz Nelson VP, Editorial Director, InformationWeek Business Technology Network, and Executive Producer, TechWeb TV

UNITED BUSINESS MEDIA LLC

Pat Nohilly Sr. VP, Strategic Development and Business Admin.
Marie Myers Sr. VP, Manufacturing

UBM TechWeb