

Jason Brome's Weblog

An English Technologist in New Jersey...

Copyright 2008

<http://www.jasonbrome.com/blog/>

Unable to connect to package content server

Not the most interesting blog posting title, is it?

Anyway, the reason for the aforementioned title is that I just upgraded at work to a new laptop, a Lenovo (nee IBM) Thinkpad. One of the issues I hit while setting it up is that the ThinkVantage System Update utility repeatedly failed to work, reporting 'Unable to connect to package content server.'

After a quick install and run of [Ethereal](#), the culprit turned out to be the System Update utility, which attempts to download package information from a remote server on port 7618. Our firewall here at work is pretty restrictive when it comes to outbound access on non-standard ports, and therefore it was blocking access to the remote package server, resulting in the error. A tweak of the firewall rules by our local friendly firewall administrator, and I was in business. The laptop is now happily downloading updates as I type.

I have yet to really get to grips with the new machine, but is definitely a big step up from my previous laptop, a Compaq Evo 1GHz, and I am looking forward to a speedier work environment!

http://www.jasonbrome.com/blog/archives/2006/03/24/unable_to_connect_to_package_con.html

Busy Times in San Jose...

The summer has flown past, vacations have gone, and now it is time to get back down to some serious business. Started a new project on Monday, and it looks like I will be spending some quality time out here in San Jose - certainly a change from sunny New Jersey!

Hopefully I will be able to squeeze in a [Mobile Monday](#) or two while I am out here - it has been a few years since I was heavily involved in the mobile space, and I am interested to find out what has developed since that time. Aside from that, with the little spare time that I will have available, I intend to catch up with a few people based over in the Valley. If you're around the area, and want to grab a cup of coffee one weekday evening and talk technology, just drop me an email.

http://www.jasonbrome.com/blog/archives/2005/09/21/busy_times_in_san_jose.html

quickSub 0.3.5 released

With the [recent talk](#) about feed subscribing, I realized that [quickSub](#) was long overdue a new release. So, at long last, quickSub 0.3.5 is now available. The most prominent change is the addition of FeedDemon to the list of newsreaders - this has certainly been the most-requested modification since the last release.

http://www.jasonbrome.com/blog/archives/2005/05/30/quicksb_035_released.html

Caller ID R.I.P.?

Caller ID has been compromised. No longer do you have the assurance that the number displayed on your ringing telephone actually represents the person on the other end of the line.

With the advent of so-called Caller ID spoofing services, anyone with a credit card can initiate outbound calls with the Caller ID of their choice. Aside from the fact that these services easily allow anyone to compromise the integrity of the Caller ID service, they also open up a number of critical security concerns across different voice-based systems.

Logging into my cellphone website account today, I noticed a security alert for the voicemail system. My provider's voicemail system has an option to bypass the requirement to enter your PIN if you're calling from your own cellphone. Hackers have been able to access the voicemail accounts of others through exploiting this 'feature'. If you have a cellphone voicemail account, you really should consider enabling PIN-based authentication.

Think about the other types of systems today that use Caller ID for authentication. Caller ID spoofing will have a big impact on all of these if it is the sole authentication token. Here's another example: Recently I received a replacement debit card, and, when I made the call to enable the card, it confirmed that it had matched my telephone number to the number on file and would not require any further confirmation. This is scary - banking institutions should seriously reconsider the authentication model used for new card enablement.

I'm not completely familiar with the intricacies of the switching network (SS7 et. al.), however I do hope that some steps are taken to restore integrity to the network. Caller ID is a useful feature, but, with the advent of spoofing services, its value has diminished.

http://www.jasonbrome.com/blog/archives/2005/04/29/caller_id_rip.html

NASA's Java PathFinder

Now [this](#) looks interesting:

Java PathFinder (JPF) is a system to verify executable Java bytecode programs. In its basic form, it is a Java Virtual Machine (JVM) that is used as an explicit state software model checker, systematically exploring all potential execution paths of a program to find violations of properties like deadlocks or unhandled exceptions. Other than traditional debuggers, JPF reports the whole execution path that leads to a defect. JPF is especially suitable to find hard-to-test concurrency defects in multithreaded programs. This is NASA's first program to be actively developed and hosted on [SourceForge](#), licensed under the '[NASA Open Source Agreement](#)'.

Something to add to the must-look-at list!

http://www.jasonbrome.com/blog/archives/2005/04/27/nasas_java_pathfinder.html

Greasemonkey

Today I had my first experience with [Greasemonkey](#). For those of you who have yet to come across it, Greasemonkey is a [Firefox](#) extension that opens up a whole new realm of client-side website personalization opportunities.

Greasemonkey enables the execution of user-written scripts against any webpage. These scripts can interact with the webpage, look-up external content, and literally perform any type of modification.

The first example I tried was Adrian Holovaty's [Chicago Transit Authority map](#) on Google Maps. This script seamlessly adds a third option to Google Maps. Now, in addition to Map and Satellite options in the top right corner, an option for CTA map is added. With the map pointing to Chicago, clicking on CTA map replaces the existing map with the layout of the Chicago Transit tracks. Anyone going to build one for the NYC MTA?

There's already a long list of [canned scripts](#) available from the Greasemonkey script repository. Another interesting script is Jon Udell's [Library Lookup](#). He [modified his existing bookmarklet](#) to rewrite Amazon book pages. Now, while browsing books at Amazon, the Greasemonkey script goes out in the background to your library's online database and checks for availability. Awesome integration!

Now, it makes me think that I should get around to writing a Greasemonkey quickSub script. This would rewrite any links to RSS/Atom feeds so that they included the appropriate URL to enable subscription against a user's desktop or web-based aggregator. Any interest?

<http://www.jasonbrome.com/blog/archives/2005/04/20/greasemonkey.html>

Going SWIMming...

Congratulations to [Daniel](#) and the team at [under\[de\]construction](#) for the v1.0 release of [SWIM](#). SWIM provides a very innovative approach to Content Management, without the complexity of traditional solutions. You can read more about SWIM in the [post over on Daniel's blog](#).

An Open Automotive Applications Platform?

Following on from yesterday's car radio podcasting request, GadgetGuy [brought me down to earth](#) with what we could realistically expect in the near future.

While I do not expect car stereo manufacturers to grok podcasting, it would be nice if they offered me an open applications platform on which I could deploy a podcasting aggregator.

In other words, how about we see the same transition that happened in the mobile phone industry? Nowadays I can go into any wireless phone store and pick up a cellphone that has some form of OS on which I can deploy my own software or (more realistically) deploy software from a software vendor. While I do not expect to see Motorola or Nokia offering a podcasting application for their cellphones in the near future, I could certainly envision some opportunities for the third-party software market.

Give me a car stereo with a built-in hard drive, some form of connectivity (Bluetooth would be better than nothing, WiFi would integrate nicely with the home networking environment and hotspots), and an open platform on which I can deploy my own applications.

This sounds like a prime market for Microsoft. A cursory search turns up Microsoft's [Automotive group](#), with their [Windows Automotive](#) platform. From what I can garner from the site, this is based upon Windows CE. According to the site, its goal is:

"... to deliver adaptable, scalable platforms for connected devices that will enable and enhance applications and services offering flexible solutions for customer needs in the automotive industry."

Sounds great. They also say:

"This technology will enable the industry to give drivers and passengers seamless access to a wide range of Web services as well as smooth functionality among all Windows powered devices."

Wonderful. With Microsoft's other relevant investments in the automotive space ([Streets and Trips](#), for example), it looks like a great space in which they could capitalize. A look at the partners list shows some very interesting developments, like the [Carman i from Nextech](#). A question to [Mr. Scoble](#) - when am I going to be able to go to my local car electronics/big box store and get one of these things installed? And how open is this platform?

A quick aside - with the recent announcements about [Tivo-to-Go](#), wouldn't it be a great app to be able to transfer Tivo-ed programs to the Windows media-enabled entertainment system in my car? (For the benefit of the people in the rear seats, of course!)

Aside from Microsoft, I'm curious about other vendors that are making advances in this space. A few years back a company called [empeg](#) created a Linux-based in-dash mp3 player. It had an integrated hard drive, and a pretty decent UI. The technology (and team) got acquired by SONICblue, who eventually discontinued the car player division. The team behind that product moved onto other efforts. This technology would have been prime for evolving into a next-gen connected in-car system. Other companies, like [PhatNoise](#), have also made advances in terms of offering HD-based digital media capabilities in the car; it would be interesting to see their product roadmap.

Anyway, I'm going to keep on dreaming about what I'd really like to see in my car. Hopefully in the not-too-distance future I'll be able to buy a product that takes the first steps towards it.

Podcasting: I want an aggregator in my Car Radio

With all the recent talk (and hype) about podcasting, I thought I'd chip in with my two cents.

For me, the killer podcasting application would definitely reside within the automotive environment. This is really the only time that I have to consume such audio content, and, like Russ Beattie mentioned in a [recent post](#), I think it would make a significant difference to my daily commute.

Now, here's my problem: I want listening to podcasts to be a seamless experience. Just like I can turn on my car radio today and listen to an AM or FM station, or play a CD, I want to be able to turn on my radio and hit a button and select the podcast I want to listen to. Or, have a predefined daily playlist of podcasts for my to-work and from-work commutes.

Essentially, I want an aggregator in my car radio.

I've thought about this for a while, and here's my imaginary specification sheet for this 'podcast-enabled' car entertainment system:

Built-in storage - multi Gigabyte hard drive

A hard-drive within the system would be used to store podcast content, as well as other media.

Wi-fi enabled (preferably 802.11g)

When your car is parked at home, or at an available hotspot, the system will be able to commence downloading of new content (or partially downloaded existing content) without requiring any user interaction.

Web-based interface

Aside from the usual in-car User Interfaces, the system will feature an embedded web server. When your car is parked at home, you'll be able to connect to this interface from any browser in your house, and use it to manage playlists, perform an OPML import of podcast feeds, and manage other configuration.

An iTunes-like application could also be provided to support the synchronization of non-aggregated audio and visual contents, such as your personal MP3 library.

Supports both audio and video content

In addition to audio content, video content (MPEG2, MPEG4, Quicktime etc.) could be supported as well. This would especially be for the benefit of those in the back of the car, who could watch new content on the in-car video system. For example, new cartoons could be downloaded daily to entertain children in the back of the car.

Intelligent audio processing

In addition to standard playback of podcasts, I would like to be able to configure some intelligent pre-processing. This would include the ability to time-stretch, speeding up (or slowing down) the podcast without making the presenter sound like a chipmunk, as well as some graceful empty-space management, removing those pregnant pauses. Every podcaster has a different presentation style; I would like to have more control over how I listen to them.

Existing capabilities of a car radio (FM/AM receiver, CD/DVD player)

Along with the rest of the capabilities, there's no reason why it shouldn't still let me listen to standard

terrestrial radio.

So, when can we expect something like this to be brought to market? Soon, I hope. With the advent of technologies like Via's [Mini-ITX](#) and [Nano-ITX](#), a complete PC on a card 12cm by 12cm, it is quite easy to envision a home-grown project to create this next-generation in-car entertainment system.

Technologies like this will pose an alternative, and a challenge, to the satellite radio industry. At some point in the future I could easily envision a wide range of content providers offering free, paid, and sponsored content services. Right now I could sync podcasts not only from a wide range of individuals, but from some of the world's largest broadcasters. And, for me, this is more than enough for my daily commute!

http://www.jasonbrome.com/blog/archives/2005/01/08/podcasting_i_want_an_aggregator_.html

Claiming my feed at Feedster

I just noticed that Feedster are offering some statistics to those who register their feeds. So, here goes...

No Need to Click Here - I'm just claiming my feed at Feedster

http://www.jasonbrome.com/blog/archives/2004/12/06/claiming_my_feed_at_feedster.html

Apache Derby on Mac OS X

Earlier this week I got an email from [John Tangney](#), a user of [nntp//rss](#), reporting a problem running the latest release on Mac OS X. After some investigation, it materializes that the problem is actually related to [Apache Derby](#), the new embedded database within nntp//rss as of v0.5-beta-1.

Derby was failing on the first startup, with the following exception written to derby.log:

```
java.io.FileNotFoundException: /usr/local/nntprss-0.5-beta-1/nntprssdb/log/log1.dat (File exists)
  at java.io.RandomAccessFile.open(Native Method)
  at java.io.RandomAccessFile.(RandomAccessFile.java:204)
  at org.apache.derby.impl.io.DirRandomAccessFile.(DirRandomAccessFile.java)
  at org.apache.derby.impl.io.DirRandomAccessFile4.(DirRandomAccessFile4.java)
...
```

A bit of prodding around in the Derby code showed that it was trying to initialize its subclass of [java.io.RandomAccessFile](#) with the [sync mode 'rws'](#). I had not experienced any difficulties on Windows or Linux with Derby, so I started up a copy on Mac OS X 10.3.6, and, low and behold, got the same exception. A bit of judicious Googling finally turned up this [IBM developerWorks forum post](#). Looking in Apache JIRA, I found this [report \(and extensive conversation\)](#) related to the bug, which basically identifies it as a Mac JVM issue. Unfortunately, to date, there has been no resolution from Apple.

Fortunately Derby has a property, `derby.storage.fileSyncTransactionLog`, which can be set to true to workaround this issue. For example, nntp//rss on Mac OS X should be started with the following command:

```
java -Dderby.storage.fileSyncTransactionLog=true -jar nntprss-start.jar
```

If you're developing a Mac OS X application using Derby, you can also set this property within your code, before initializing the Derby driver. The [System-Wide Properties](#) section of the Derby documentation provides more details.

http://www.jasonbrome.com/blog/archives/2004/12/05/apache_derby_on_mac_os_x.html

Amazon goes Queuing

A colleague of mine passed on a surprise announcement from Amazon: They've just launched the v1.0 beta of their [Simple Queue Service](#). It is an online queuing service with some very simple SOAP **and** REST-style APIs for message queuing.

For the beta, they've limited message size to 4KB, and a maximum of 4000 messages across all queues for a single subscription ID. According to the [FAQ](#), messages are kept for up to 30 days.

The API is quite simple - methods for creating, updating, deleting and querying queues, as well as three methods for queue interaction: Enqueue, Read, and Dequeue.

The messaging read process is somewhat intriguing. A client application issues a 'read' request against the queue. Pending message(s) are returned (you can request more than one message in a read request) in response to the request, and a lock is placed on those messages. This means that successive read requests within the lock period will not get the same series of messages. The client application then explicitly dequeues the messages by issuing a 'dequeue' request against the queue, supplying the QueueEntryId for each message to remove.

One interesting caveat is mentioned in the documentation for the [Read operation](#):

A message may be returned by the Read operation even though it has already been dequeued, and concurrent Read calls may return the same message to multiple readers. This behavior is a result of prioritizing reliable data storage (even in the face of hardware failures), and we expect such events to occur very infrequently.

It seems that SQS is optimized for reliability, but not for once-and-only-once delivery. Applications utilizing SQS would need to ensure that they handle duplicate scenarios. The documentation basically implies that the appropriate solution is to ensure that messages are idempotent; retrieving the same message twice should have the same effect as receiving it once.

Currently security is pretty lax. One only needs to know the Subscriber ID to be able to retrieve a list of queues, and from there it would be possible to retrieve messages from those queues. I'm sure that this is something that Amazon would address as they further enhance the service. A simple password-based mechanism would take the security up a level, and, in the future, certificate-based mechanisms could be employed.

With this foray into more 'Enterprise Level' services, it does make me wonder where Amazon is headed. They've certainly mastered the eCommerce world, and have provided some interesting Web Services focused on that domain. Now it seems that they're venturing out into some more horizontally focused services, and exposing some of their platform-level capabilities. I wonder what's coming next...

http://www.jasonbrome.com/blog/archives/2004/11/04/amazon_goes_queuing.html

Web Services Version Mania

This past week I attended [Oracle's Developer Days in New Jersey](#). The sessions basically covered Oracle's pitch on SOA - although, unsurprisingly, after an initial high level SOA level-set, it seemed to focus more on the presentation, business and persistence level capabilities of their 10g Application Server.

However, at the end of the day, Tyler Jewell from [The Middleware Company](#) gave a brief presentation on their SOA/Web Services perspectives. On one slide he had a phrase that has been in the back of my mind for a number of months - Web Services "Version Mania."

The problem I see is in terms of the evolutionary nature of Web Services. Web Services get defined and published. Consumers begin to use those Web Services. Web Services evolve, add functionality, expose enhanced interfaces. As I think about this topic, the following questions arise:

How do we manage versioning of Web Services interfaces? I may have a number of consumers of MyWebService v1.0. I now have MyWebService v2.0 - how can I deploy this Web Services and transition clients? How do we manage graceful deprecation of previous versions? I want some capability to inform (directly or through a registry) Web Service clients of the availability of an enhanced interface. I do not want to have to support x previous versions of the same service. What tools exist today that provide me the ability to pro-actively push update information to known consumers of a Web Service?

There have been a [number of discussions](#) recently regarding the importance of a registry within any Web Services initiative. Being able to capture, manipulate, query and disseminate Web Services meta-data is a very powerful capability. Versioning and Deprecation information, as well as the functionality to support its management (including change notification), sounds like a good fit as a capability of the registry.

The [subscription API](#) within UDDI v3 looks like a good potential for the notification channel. Changes, updates or deprecation of Web Services could be dispatched to all known subscribers for the particular Web Service (or group of Web Services).

It's early days for my thought process on this topic, and I'm in the process of doing some more research. If anyone has any pointers to specific approaches to versioning and deprecation, or vendors doing a good job attacking this problem, please leave a comment below. I'm especially interested in any standards-based approaches to addressing aspects of this challenge.

http://www.jasonbrome.com/blog/archives/2004/10/31/web_services_version_mania.html

Web Services Reliable Messaging - An Update

Reliable Messaging holds a special interest for me, having played a significant part in my career (in the manifestation of various technologies such [MQSeries](#), [MSMQ](#), and, most recently, the products of [Envoy](#)) over the past ten years. Therefore it is with great interest that I have been monitoring the progression of the two emergent Web Services Reliable Messaging standards - WS-ReliableMessaging and WS-Reliability.

Just over a year ago I posted an [entry to this blog](#) questioning whether there would be convergence across the two aforementioned specifications. Therefore I thought it was about time to check up on their status, and see whether there had been any significant activity.

Well, it seems that both specifications are still forging ahead, with support from different vendors and, in the case of WS-Reliability, [Oasis](#), a standards organization. The vendors behind WS-ReliableMessaging have held a number of vendor workshops, and WS-Reliability has reached a formal v1.0 release within Oasis. Below, I've summarized the status of both specifications.

The development momentum behind IBM, Microsoft and BEA suggests that WS-ReliableMessaging will definitely capture a significant marketshare. The underlying concept (reliable messaging) behind both these specifications is common, and it would not surprise me to see some of the proponents of WS-Reliability supporting both variants within their product offerings. Certainly with the convergence on common specifications within other [aspects](#) of the WS-* stack, it would be a shame to see disconnected islands of Web Services reliability support across different vendor implementations - let's hope that we do not have to resort to WS-Reliability WS-ReliableMessaging bridges!

I'll probably check back in early next year to see what has developed. With the emergence of new server products from BEA (Quicksilver) and IBM (Jetstream), as well as early offering of Microsoft's Indigo, it will be interesting to see how this area plays out.

Note: The information below is by no means complete. This is an evolving posting - if there are any inaccuracies or additions, please leave a comment and I will update the content.

WS-ReliableMessaging

Latest Spec: [March 2004](#)

Editors: [BEA](#), [IBM](#), [Microsoft](#), [Tibco](#)

Current Implementations:

[Apache Sandesha \(WS-RM on top of Apache Axis\)](#)

[IBM Emerging Technologies Toolkit](#)

Microsoft Enterprise Solutions Framework (early implementation of WS-RM) - see [blog posting](#)
[Systinet Server for Java](#)

WS-Reliability

Latest Spec: [1.1 \(Committee Draft 1.086, 24 August 2004\)](#)

Editors: [Fujitsu](#), [Novell](#), [Oracle](#), [Sun](#)

Committee Members:

[Arjuna Technologies](#), [Booz Allen Hamilton](#), [Choreology](#), [Cyclone Commerce](#), [France Telecom](#), [Fujitsu](#), [Hewlett-Packard](#), [Hitachi](#), [Mitre Corporation](#), [NEC](#), [Nokia](#), [Novell](#), [Oracle](#), [SAP](#), [SeeBeyond](#), [Sun](#), [University](#)

[of Hong Kong](#), [webMethods](#), [WRQ](#)

Current Implementations:

[Fujitsu / Hitachi / NEC RM4GS](#) (Reliable Messaging for Grid Services)

[Oracle Application Server 10g](#) (presently developer editions only, I believe)

http://www.jasonbrome.com/blog/archives/2004/10/30/web_services_reliable_messaging_.html

[nntp//rss v0.5-beta-1](#)

... has just been released. Lots of features that have been lurking in CVS for over a year (Atom parsing, OPML import/export, use of Jakarta Commons HTTP Client), as well as some newer features (Categories, Apache Derby embedded database.) Still working towards a final v0.5 release, but this version is definitely worth a look.

[nntp//rss Project Page](#)

http://www.jasonbrome.com/blog/archives/2004/10/25/nntprss_v05beta1.html
