

What's happening with Firebird and InterBase?

as seen by Paul Reeves

Both the Firebird project and Borland have almost simultaneously released new software based on the InterBase database engine. I dealt extensively with the new features of Firebird 1.0 (Fb 1.0) a month or two ago. This time we'll look at what is in InterBase 6.5 (IB 6.5) and do a comparison between the two as we go.

Firstly, though, perhaps we should recap the key features of InterBase generally. Some of you may have only just arrived at the point where you need to consider adopting a new database technology and discussion of new features may be a little arcane if you don't know what InterBase actually does. The rest of you can skip the next section and go straight to the discussion on IB 6.5.

InterBase Core Features - A recap

InterBase is a fully relational database server designed to allow lots of users to work on the same database simultaneously, while not requiring developers to have to do anything special to achieve this. That's right - it is multi-user out of the box.

All activity is transaction based, allowing grouping of statements into coherent blocks. If one statement fails, all the preceding statements can be undone by executing a transaction rollback.

The server uses a fairly vanilla implementation of the SQL standard. By that, I mean it deviates less than most engines from the standard. The use of row-level triggers on tables and a straightforward procedural language for writing stored procedures allows developers to define server side business rules that can be enforced against all applications.

The server itself is lightweight, undemanding of hardware and available on a variety of platforms and processors. The engine and command-line tools install into less than 10Mb disc space, while it will quite happily run on a system with 64Mb RAM. There are, of course, many more features but that is not the point of this article. You can find loads of material on the IBPhoenix and Borland websites discussing existing features.

InterBase 6.5 - the new features

There are several new features, but no showstoppers. Let's take a look.

- **Enhanced metadata security.** This feature adds extra protection to system tables. The new default behaviour is to reduce PUBLIC access to SELECT only. The feature is largely implemented as SQL GRANT/REVOKE statements, so it is unlikely to remain out of Firebird for long. Borland have also included a script that blocks tools doing metadata lookups of the database. By this means it becomes easier to prevent anyone examining the database structure. This is a useful feature for deployment of embedded applications.
- **64-bit file i/o.** This feature is long overdue and is also present in Firebird 1.0. InterBase has always

supported large database sizes by the use of secondary files. However, individual file size was limited to the 32-bit address space. This was largely the case for most underlying operating systems, too. When InterBase 6.0 was on the drawing board back around '98, hard discs were typically around 4 to 10 Gb in size. Very large databases were more or less impossible to create. Since then a lot has changed and many users have accidentally broken the barrier imposed by 32-bit file pointers. This has predictably led to corruption of the header page as the page pointer has wrapped around.

- **Asynchronous cancellation of queries.** This is a nice feature that has often been requested. However, this implementation of it requires a separate thread to do the cancel. Most existing tools won't support it and neither, I suspect, will many existing applications. Hands up all of you who *know* you should have taken the trouble to implement database access via a thread but have never got around to it.

Ad hoc query tools are where this feature is needed most and tool vendors are starting to implement this feature in new versions. Firebird doesn't support this..... yet.

- **ROWS clause in ORDER BY.** This feature is semantically equivalent to Firebird's new FIRST/SKIP statement. However, Borland have chosen to implement it after the ORDER BY section of the select rather than as part of the select itself. I'm really not sure if this is more intuitive than putting it in the select. What do you think? Here is the Borland way:

```
Select *  
from employee  
order by last_name, first_name  
rows 11 to 20
```

And this is the Firebird way:

```
Select first 10 skip 10 *  
from employee  
order by last_name, first_name
```

In this instance, the ROWS implementation is clearer, although neither style requires too much thought.

Before finishing with this subject I should add two things - a) the Borland implementation has other features that I haven't mentioned. They are worth checking out. b) Neither implementation conforms to any SQL standard. Currently there are three ways that I know of for implementing this feature. It is mainly used in web applications to display search results in meaningful chunks.

- **New Config params - CPU affinity, Quanta for sweep, sweep yield time & user time.** Microsoft's implementation of symmetric multi-processing (SMP) just doesn't work well with InterBase. For a start, InterBase only works on one processor. Worse, however, is that whenever InterBase hits 100% CPU usage the O/S decides to switch processors. This actually degrades performance under heavy load. For this reason it is now possible to tie InterBase 6.5 to a single processor. It's not a great long term solution - that is promised in InterBase 7.0 which should fully support SMP - but it does stop the switching and allows the second processor to take the load of InterBase when working on other tasks. This feature has also been implemented in Firebird 1.0.

I'm really not sure about the other new config params. One of the beauties of InterBase has always been the lack of things to tweak. Typically, under almost every scenario I can think of, tweaking in this manner usually leads to worse performance than following the defaults. If the designers don't think they have got the sweep quantum right, why do they think you will be able to? Still, it gives us something to play with.

- **Non-padded varchars across network.** Until now, InterBase (and Firebird) have sent varchars padded to their full length across the network. The reason for this is hidden in the depths of InterBase's history. It certainly made sense at the time. Now, however, it doesn't.
- **Large cache sizes no longer degrade performance.** Some work has been done on cache management to prevent performance degradation with large cache sizes. It is hard to tell what difference this is going to make. Borland are really only claiming that large cache sizes will not now perform worse than smaller cache sizes.
- **XML Generation.** I'm yet to be seduced by XML so I don't know whether this is a 'must have' or a yawn. InterBase 6.5 ships with a dll that encapsulates some API calls and returns data wrapped in XML tags. There's no rocket science here and, I dare say if it was really important for a project, a similar bit of functionality could be knocked up in a day.
- **Miscellany.** There is a bunch of other features in InterBase 6.5. The Easysoft ODBC driver has had lots of bug-fixes. Of course, you can get the Easysoft driver without deploying InterBase 6.5, but that is another matter.

InterClient 2.0 has been recompiled and is now called InterClient 2.5. That's not quite how Borland put it, but that is what it amounts to. Some calls within InterClient that have been deprecated by JDK 1.3 have been removed. A handful of functions have been fixed to work properly with Dialect 3 and with 64-bit integers.

IBConsole has had a bunch of fixes applied, but again, you can download this for free if you want it.

InterBase 6.5 - the negatives

Well, these boil down to three, and two of those are really two sides of the same coin. Firstly, InterBase 6.5 is not available as open source. This means that we can't see how any of the new features have been implemented. Not only that but, if there is a bug in there, we can't fix it - we are

back to being wholly reliant upon the closed development model. Considering that many of the bug-fixes implemented by the Firebird project have made their way back into InterBase 6.5, one has to see this as a bad thing. New InterBase features will no longer benefit from Firebird developers fixing the code.

Secondly, and perhaps worse still, you have to pay license fees for InterBase 6.5. Borland really have abandoned open source as a method of software development. The reason for this is probably because the OS model requires a community of developers behind it to make it work and that community is wholly dedicated to Firebird. I suppose we should see this development as a round definitely won by the Firebird project.

The third negative of IB 6.5 is that it is wholly super server. Super server supports higher numbers of concurrent users and provides better admin facilities, but it suffers from one big drawback: it is threaded, using a single process. If the server crashes, it takes every connection with it. By contrast, the original architecture, commonly called classic, uses a separate process for each connection. It can be slower and requires access to a lock file to manage contention but, if a connection goes awol, all other users carry on regardless.

Firebird has more or less implemented classic and super server on every supported platform. There may even be a Win32 version of classic available soon. In the long term, the intention is to entirely deprecate the classic architecture, but to do this requires making Super Server rock solid. We are not there yet and, in the meantime, users seem to like having the choice.

What to use and how to choose?

Borland are demonstrating with InterBase 6.5 that they are still committed to InterBase. The development team is severely reduced from former days, but at least there are full-time developers working on the code. Version 7.0 is apparently under development, and the usual promises are being made that it will fix all our current problems with InterBase, while adding lots of new features. Yes, we are well and truly back to that naive hope, widespread in the software industry, that the next release will be better than the last.

So, what to use? Well, InterBase 6.5 is marginally ahead of Firebird 1.0 on features. If there is a particular feature that you really need, then it may be worth choosing.

Unfortunately, that choice could cost a serious amount of money. InterBase 6.5 distribution is license based and prices start at around £125 per licence for small numbers. This needn't be a problem, especially if the cost can be passed on. It is, after all, the way InterBase has always been sold.

When Borland's commitment to InterBase was reasonably beyond doubt, the license looked like quite a good deal. However, since December 1999 that commitment has appeared questionable to say the least. If I was a manager with an upward career path, I doubt that I would be recommending corporate adoption of InterBase. Likewise, if I was a VAR looking to invest resources in the long-term development of vertical market applications, I would want

evidence that InterBase will still be there for me in three years time. As a consultant, I would hesitate to risk my reputation by recommending a product that just may not be around in eighteen months from now.

The InterBase division used to generate over US\$10 million revenue with around a 40% margin. At the time, it had little serious competition within its own niche. That business has largely gone, and it is now competing with its alter ego, Firebird.

Firebird is obviously picking up users that InterBase never could - small applications that would never, ever generate an income for a software vendor. We can discount that business. And Firebird, as a young project, will not be considered seriously by many corporate users until it has more of a track record.

That still leaves a large sector that could go either way on Firebird or InterBase. Here, price is likely to be a deciding factor. Borland have already accepted they cannot compete on the pure open source level. They are going to find it increasingly difficult to compete with Firebird on price. The sector that adopts Firebird is likely to make the difference between Borland breaking even on InterBase and making a profit out of it.

Borland are currently committed to ongoing full-time development, which is a luxury that the Firebird project doesn't have. But then, the Firebird project has virtually zero costs, so it is reasonable to foresee that it could go on enhancing the code indefinitely. This is not an option that Borland has.

Users who are concerned about the long-term prospects for Firebird do have the option of contributing their time to its development, or paying for development in some way. The easiest and cheapest way to achieve this is by purchasing a CD from IBPhoenix. The profits from this are ploughed back into further development of Firebird.

Squid'll Fix It

If you want the TTreeView's right mouse key to select in a way similar to Explorer then setting the RightClickSelect is not enough. Although this does set the Selected property it does not set the Selections[0] value correctly. To get around this just put the following code in the treeview's OnMouseUp event:

```
if Sender is TTreeView then begin
  TTreeView(Sender).Selected :=
  TTreeView(Sender).Selected;
end;
```

Adrian Rye

Delphi Developers
Conference 2002

See you there!

Firebird News

The code base has been moved over to C++ and memory management has been completely overhauled. Serious work on Firebird 2.0 is just about to start. In the meantime, the good bits of InterBase 6.5 will make their way into Firebird in the coming months. In the next article, I will look more seriously at what the Firebird project plans for version 2.0 and how developers can get involved.

Conclusion

Overall, Firebird 1.0 is looking a better long-term bet than InterBase 6.5. The latter has the edge on features, but is heavily defeated on price. The security blanket that a large multi-national corporate usually provides is missing in this case. The aforementioned corporate has already tried to pinch the blanket. Meanwhile, Firebird has a strong commitment from developers and users across the world, so it seems likely that it will stay the course longer than Borland.



Paul Reeves is currently looking after European Technical Support for IBPhoenix, so he has to declare an interest in the success of Firebird. When not doing support, he is one of the many working to make Firebird a better InterBase. He can be contacted on preeves@ibphoenix.com.



Firebird 1.0 Is Now Available on CD

Get the most comprehensive, up-to-date
Firebird resources available.

IBPhoenix exists to further the development of the open source relational database known as Firebird. IBPhoenix generates its income through CD sales and the provision of support services for Firebird. If you wish to support this open source project, get your copy of the CD today. More details about IBPhoenix can be found at: www.ibphoenix.com

CD Contents:

- All the Firebird 1.0 Binaries ready to install
- (Windows, Linux, MacOS, Solaris, HP-UX, FreeBSD etc)
- Browsable snapshot of the source code at 1.0 Release
- Searchable knowledgebase
- Large collection of documentation and how to's
- Third party tools and drivers
- Firebird 1.0 Documentation Set and User Manuals.

Pricing

Corporate: £175.00
Developer: £105.00
Educational: £ 70.00
Or consider a yearly subscription
(4 CD's for the price of 3):
Corporate Subscription: £525.00
Developer Subscription: £315.00
Educational Subscription: £210.00

To Order

Please send your order (stating what type of CD you require), your preferred payment method, and your shipping address to:

E-mail: orders@ibphoenix.com
Telephone: 01844 354465

