**- The OrbZone - http://www.orbzone.org -**

# Response to 'The Rise and Fall of CORBA by Michi Henning'

**Posted By Dion On 27th June 2006 @ 17:46 In Recent Articles | 3 Comments**

***Original Article by Michi Henning[1]***

**Michi Henning's recent article entitled "The Rise and Fall of CORBA" provided a very interesting insight into the history of the standardisation of CORBA from someone who was a core part of that process. While some of the technical limitations he addresses are correct, others are taken out of context or even from outdated specifications. CORBA is alive and well and applicable today in more industry verticals than it was in its heyday of the 90's. Airline reservations, e-commerce back-ends, telco transactions and financial systems all deliver millions of messages per second powered by CORBA. The truth is that the most vocal CORBA detractors are CORBA competitors, with products that are derivatives of CORBA.**

**It is unfortunate that a lot of the attacks have been aimed at the CORBA standardization process, and by extension, the OMG itself. It is relatively easy to sit alone and develop a proprietary solution that fits exactly what a single company wants to achieve. It's much harder to develop a standard that suits every voting member. Arguments, blockages, concessions and complications are all part of this process. Does that mean the multi-vendor standardization process is flawed? Not at all! In fact, I would argue just the opposite. A standard that has the approval of all interested parties in that space has a far greater chance of acceptance and survival than does the rogue standard created by a single, uncooperative vendor. CORBA is just one example of a consensus-derived standard; the W3C and SCA have also been extremely successful with their efforts as well. And even Sun, who developed and standardized J2EE single-handedly, had to eventually introduce the JCP. The argument is that it's much easier to create a bug-free specification without the hindrance of a consortium. But let's be realistic here – no specification is perfect. Consortium-derived specifications often seem buggier simply because of their increased scrutiny and multi-vendor implementations as opposed to single-vendor specifications which are usually implemented only by that same vendor. And as far as standardization goes, the single-vendor specification still has to gather industry support and public acceptance to become a recognized standard. This means that at some point the specification will be put through the same arguments, concessions, blockages, bug issues and cooperation as the consensus-based effort. CORBA achieved consensus and usage which is why there are so many battle hardened CORBA systems working today. Companies that can co-operate on a standard prove that they are mature enough to be able to co-operate for the benefit of customers. The alternative is domination by one vendor which either has a monopoly and charges excessively or is too small to sustain themselves for the long lifecycles which will be demanded of infrastructure products. Michi Henning put this very robustly on the newsgroup comp.object.corba in January 2002 (click here for link[2]):**

***"I definitely don't want to go back to the days where I have to throw myself in with one particular vendor and be at that vendor's mercy when it comes to upgrading, getting new feature support, or being shielded from that vendor's decision to pull out of this particular market..."***

**Technically speaking, there is much benefit associated with a group-derived standard as well. A resounding example is the bundling of the static stub/skeleton approach with the dynamic DII/DSI approach. Originally submitted as two separate specifications for CORBA, there were arguments for and against each paradigm. In the end no clear-cut decision could be made in favour of one or the other and so both were included in the CORBA specification. It was only years later during the maturity of the various implementations that people realized this was a brilliant decision. Proponents of the stub/skel approach soon realized that some CORBA services were much easier implemented using DII/DSI and vice-versa.**

**It is fair to say now that the success of CORBA has been due to the implementations rather than the standard itself. There are very successful CORBA implementations in both the commercial and open-source markets. IONA for example has over 4500 CORBA deployments across various industry verticals (click here for link[3]).**
**Borland is also a very successful CORBA vendor with its VisiBroker product powering many customers (click here for link[4]). In the open-source market, TAO (and to a lesser extent, MICO and JacORB) provide a successful and mature CORBA offering with many customers. Object Computing Inc. who provide commercial TAO support have posted a great presentation on how CORBA is still delivering[5]. The examples include how CORBA is used when booking a flight, pumping gas, and the most mission-critical, no-downtime-permitted application of them all: delivering German beer![6]**

**Michi has said (pertaining to the CORBA vendors) (click here for link[7]):**

***"The industry's financial collapse drove many software companies out of the market and forced the survivors to refocus their efforts."***

**This statement however just isn't true and amounts to nothing more than an obituary announcement antemortem. There are successful CORBA vendors behind many successful and mature CORBA products. And today it is the various vendor websites and websites such as OrbZone.org that provide the ongoing, relevant CORBA knowledge. The OMG still maintains the various specification documents as well as issue tracking but the maturity of CORBA has caused the knowledge to become more distributed and solution focused.**

**The attacks on CORBA aren't strictly limited to the standardization process - some of the attacks are technical in nature. Unfortunately most of these attacks are not based on fact. For example, Michi stated (click here for link[8]):**

***"No language mappings exist for C# and Visual Basic, and CORBA has completely ignored .NET."***

**The CORBA community has not ignored .NET. The Espresso[9] product from J-Integra allows a client written in any .NET language (C#, VB, etc) to call a CORBA Server. IONA's Artix Connect[10] product provides similar functionality. There are even open-source solutions – IIOP.NET[11] pops to mind.**

**Another statement from Michi (click here for link[12]):**

***"The on-the-wire encoding of CORBA contains a large amount of redundancy, but the protocol does not support compression. This leads to poor performance over wide-area networks."***

**Again, this is simply not true. CORBA's standard protocol and encoding are extremely performant and far faster than competing standards such as Web Service's SOAP. Low-level socket coding will still maintain a performance advantage but with the extra cost to write and maintain. Proprietary protocols might save a few bytes (yes, I said bytes as this is the amount of redundancy we are talking here) but integration of systems is as important as highway systems or telephone systems are to countries so companies are not going to leave themselves exposed by using a proprietary protocol from a small company. Who would trade an interoperable, mature, multi-vendor, standardized protocol for a proprietary one to save a few bytes? Probably the same company that would abandon modern object-oriented computing practices in favour of assembly language. To improve their implementation of the standard, IONA developed ZIOP[13], a protocol extension for GIOP, which offers zip compression to messages. While not yet standardized, it demonstrates both how CORBA vendors are still able to compete with proprietary protocols and still strive to adapt the CORBA standard in response to new business needs.**

**A common criticism made about CORBA pertains to its perceived complexity. This is an easy attack to make given that the core CORBA specification tops out at over 1100 pages. But it is important to remember that the core specification is aimed as much as the implementation vendors as the end CORBA users. It is disingenuous to say that CORBA is complex simply based on the specifications. Michi said (click here for link[14]):**

***"The most obvious technical problem is CORBA's complexity —specifically, the complexity of its APIs. Many of CORBA's APIs are far larger than necessary. For example, CORBA's object adapter requires more than 200 lines of interface definitions, even though the same functionality can be provided in about 30 lines—the other 170 lines contribute nothing to functionality, but severely complicate program interactions with the CORBA runtime."***

**This is a rather glib judgement of CORBA based on a single API within a specification that mainly targets the CORBA vendors and developers of open source implementations rather than CORBA end users. The truth is that CORBA is only as complex as it needs to be. CORBA is the only distributed computing platform that can handle highly complex and scalable systems because it does not remove choices from developers (as J2EE does). If you require detailed control of object and servant life cycles, or need to tightly restrain the memory footprint of an application, CORBA provides the necessary tools to achieve this. Most of the vendors have provided good documentation demonstrating the ease at which CORBA can be deployed and users looking for tutorials on CORBA programming should not be looking towards a reference specification. I wouldn't look at the ISO C++ specification to learn C++ nor would I look at the Java VM specification to learn Java. Michi himself even addresses his own concern on comp.object.corba in 2001 (click here for link[15]):**

***"I think that at least part of the reputation for complexity in CORBA stems from the desire to pretend that no network exists and from the wish to be able to write a distributed application just as if it were a non-distributed one. I'm afraid that this will remain a pipe dream for many more years. Distribution adds complexity, no matter how sophisticated a platform you use."***

**And a final example from Michi (click here for link[16]):**

*"Security. CORBA's unencrypted traffic is subject to eavesdropping and man-in-the-middle attacks, and it requires a port to be opened in the corporate firewall for each service. This conflicts with the reality of corporate security policies. (Incidentally, this shortcoming of CORBA was a major factor in the rise of SOAP. Not having to open a port in the corporate firewall and sending everything via port 80 was seen as a major advantage, despite the naïveté of that idea.) The OMG made several attempts at specifying security and firewall traversal for CORBA, but they were abandoned as a result of technical shortcomings and lack of interest from firewall vendors."*

**Again, this attack lacks merit. The OMG has a standardized security specification along with another firewall proxy specification. Many vendors even build on the OMG specifications to provide customer tailored security solutions that include:**

**Sophisticated, mechanism-neutral API based on CORBASEC Level 2 interfaces.**

**Support for the Object Management Group (OMG) Common Secure Interoperability specification, version 2 Level 0 (CSIv2) includes username/password authentication, identity propagation control, and a single sign-on CORBA login service.**

**Separate Key Distribution Mechanism (KDM) component. The ORB can distribute pass-phrases to automatically launch server applications. The server uses these pass-phrases to decrypt the relevant private key. KDM communications are fully TLS secure (encryption, privacy and integrity are guaranteed).**

**An extensive X.509 C++ parsing API is supported, providing a complete IDL wrapping of X.509v3 certificates, and including X.509v3 extension support**

**PKCS#12 container format support.**

**CORBA is used in verticals such as telecommunication, aerospace, military/defense, manufacturing, and automotive – all of which have the highest levels of security concerns and requirements.**
**CORBA's many implementations seamlessly interoperate across network, language, CPU, and operating system boundaries. No other platform in the history of computing has offered this flexibility, and no other platform has achieved anything like this degree of acceptance and market penetration.**
**Want some more measures of CORBA success?**
**CORBA runs on a vast number of hardware platforms, including PCs, mainframes, handheld PDAs, single-board embedded systems, airframes, and supercomputers.**

**CORBA is available for virtually every operating system in existence. To mention just a few of them: Windows 95/98/ME/NT4/2000/Server 2003/XP, any number of UNIX and Linux versions, VxWorks, QNX/Neutrino, LynxOs, Chorus, pSos, OS/390, z/OS, Tandem, MS-DOS, and Windows 3.1 (Yes, CORBA is available for more Microsoft platforms than either DCOM or .NET.)**

**CORBA is available from a large number of vendors, and several Open Source implementations are available as well. CORBA is highly extensible and open to integration with any number of legacy systems, thanks to its very wide platform and programming language support.**

**CORBA has even made its way into subsequent technologies. For example, J2EE's architecture relies on CORBA to work: Application servers rely on CORBA's IIOP for RMI invocations, and require the Object Reference Template specification. The Java Transaction Service is intimately tied in with CORBA's Object Transaction Service (OTS). In fact, a CORBA ORB ships as part of every copy of the JDK.**

**CORBA has not fallen. It has risen to the tasks for that which it was designed. The hype of the dot com boom is just now becoming a reality: e-commerce is now accepted, knowledge sharing is more robust, and people are looking to the internet as a place of business. CORBA was ahead of its time in allowing internet driven business to integrate long before consumers were ready to jump onboard. And while the marketing hype has quieted, CORBA continues to power modern business and is now more relevant than ever before.**


- **[1] http://www.acmqueue.com/modules.php?name=Content&pa=printer_friendly&pid=396&page=1**
- **[2] http://groups.google.ca/group/comp.object.corba/msg/4e47f55cd2c2caad?dmode=source**
- **[3] http://www.iona.com/info/aboutus/customers/welcome.htm**
- **[4] http://www.borland.com/us/customers/profiles/view_by_product_category.jsp**
- **[5] http://www.omg.org/cgi-bin/doc?omg/2006-04-01**
- **[6] http://www.holsten.de/**
- **[7] http://www.acmqueue.com/modules.php?name=Content&pa=showpage&pid=396**
- **[8] http://www.acmqueue.com/modules.php?name=Content&pa=showpage&pid=396&page=3**
- **[9] http:///j-integra.intrinsyc.com/products/espresso/**
- **[10] http://www.iona.com/products/artix/welcome.htm**
- **[11] http://iiop-net.sourceforge.net/**
- **[12] http://www.acmqueue.com/modules.php?name=Content&pa=showpage&pid=396&page=3**
- **[13] http://www.iona.com/support/docs/orbix/6.2/develop/corba_pguide/java/compression2.html**
- **[14] http://www.acmqueue.com/modules.php?name=Content&pa=showpage&pid=396&page=2**
- **[15] http://groups.google.ca/group/comp.object.corba/msg/a5880b0e01b3f04b?dmode=source**
- **[16] http://www.acmqueue.com/modules.php?name=Content&pa=showpage&pid=396&page=2**


---