# Does Middleware assume a Perfect Network?

Brian Carpenter
Distinguished Engineer, IBM Zurich Research Laboratory
Trustee, Internet Society
*September 2002*

*Note:* This input to the first SERENATE workshop is purely my own opinion, and does not in any way represent the opinions of IBM or the Internet Society.

Ironically, as I write this note, I am cut off from IBM's internal mail system, along with 30,000 other employees, due to the simultaneous failure of two router cards at one site. So perhaps the answer to the question posed in the title is an unambiguous "yes." In any case it illustrates a common phenomenon – those designing scenarios for middleware and servers are often not the same people as those designing network infrastructure, and middleware may therefore have operational dependencies on the network which are unknown to either side. This phenomenon is echoed by a sentence in Tim Berners-Lee's book about the invention of the Web: *There's a freedom about the Internet: As long as we accept the rules of sending packets around, we can send packets containing anything to anywhere* [Berners-Lee]. In general, those designing complex middleware systems tend to assume that the network is simply there and can be used without further consideration. Distributed systems mechanisms such as DCE, CORBA, Java RMI, Web Services, and Grid, with few exceptions, treat networking as a universally available resource, with perfect performance.

There are exceptions, but they tend to be proprietary (for example, reliable messaging systems or heavyweight transaction processing systems, which contain elaborate procedures for coping with network failures). In the research community, quite a lot of attention has been paid to the interaction between video streaming and the related network transport problems. In Voice over IP work, the strict requirements of the telephone industry have led to close attention to quality of service and reliability, again at the transport level. And of course Web Services, by adopting SOAP over HTTP as its model, has recognised that transparent packet service as described by Berners-Lee is an illusion in a world contaminated by firewalls and network address translators [RFC 2775].

In practice the network is not (and cannot be) perfect. So we have a triple challenge:

- Make the network as nearly perfect as possible. This is not just a matter of "throwing bandwidth at the problem," although that will certainly help. It means reducing complexity at both the technical and administrative level, adding redundancy, and becoming obsessive about operational management. Complexities to be eliminated clearly include multiple technologies or organisations along one path, intrinsically complex technologies (such as ATM or MPLS as opposed to simple IP routing), address translation (thus IPv6 is a must), and the need for firewall traversal rather than using an end-to-end security model.

- Make the network's imperfections explicit and visible. We can't expect middleware designers to take account of network issues if all we give them is an interface that hides those issues. For example, middleware needs straightforward APIs to influence IP QOS, IP security, multihoming, etc., and to learn of outages other than by a transport timeout.
- Convince middleware designers (e.g., the Grid community) to take much more specific account of network layer issues in their designs (for example, exploiting network QOS explicitly, or placing server farms in such a way that a single network failure will not cause massive disruption).

[Berners-Lee] *Weaving the Web*, T. Berners-Lee, M. Fischetti, HarperCollins, 1999, p 208.

[RFC 2775] *Internet Transparency*, B. Carpenter, 2000. Available from http://www.rfc-editor.org