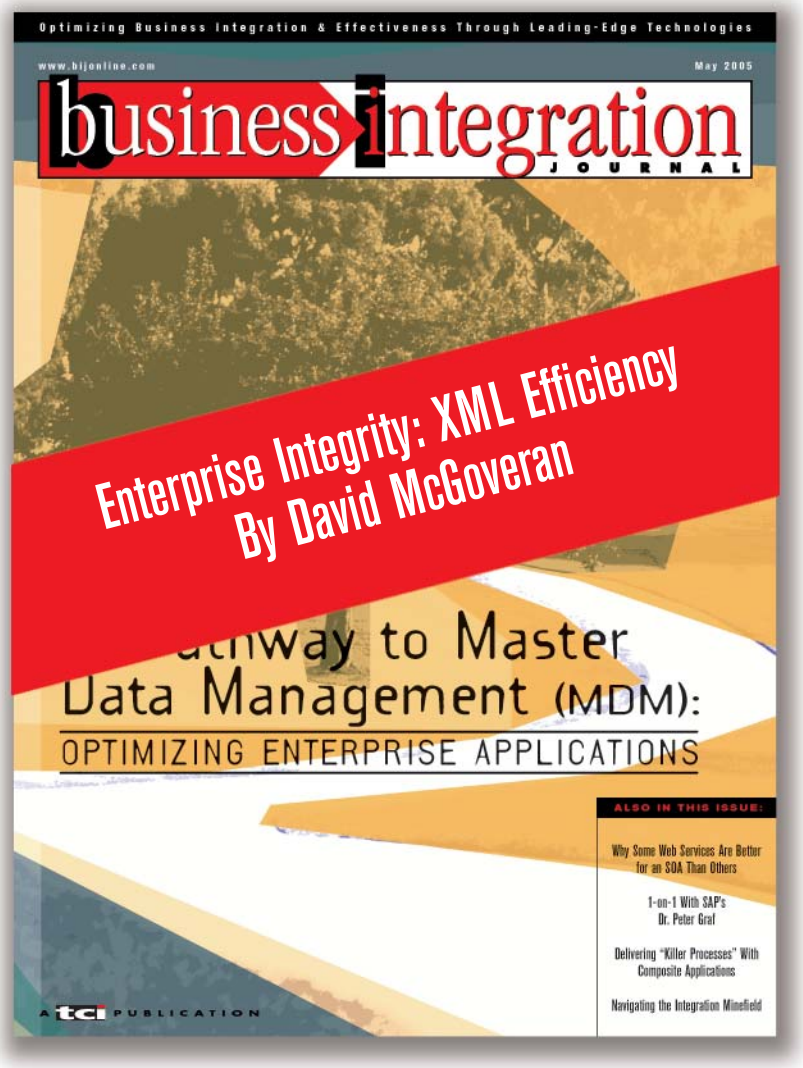
This article appeared in the
May 2005 issue of

business integration JOURNAL



Subscribe instantly at
www.bijonline.com

- Free in the U.S.
- $48 per year in Canada and Mexico
- $96 per year everywhere else

ENTERPRISE INTEGRITY

XML Efficiency

BY DAVID McGOVERAN

Sometimes our industry allows silly arguments to become impediments to productivity. A case in point is the painfully slow progress on XML efficiency. As anyone who has used it in real applications knows, XML is verbose. XML chews up network bandwidth as if that were IT's free lunch. This characteristic makes it of doubtful value for any application in which large amounts of data are exchanged or high transactions rates are necessary. Four important domains readily come to mind: Business-to-Business (B2B), analytics (including Business Activity Monitoring [BAM]), business process instance data, and data integration (including Enterprise Information Integration [EII]). XML compression seems like the obvious solution but, even though the W3C XML 1.0 draft standard was introduced in November 1996, compression just isn't a standard part of the integrator's XML toolkit.

The reason for this state of affairs isn't that the fundamental problem lacks acknowledgement. No reasonable person would argue that XML is efficient. In fact, its space inefficiency is so widely recognized that a number of XML compression technologies have been developed and many are open source. Broadly speaking, there are two reasons XML compression isn't in widespread use. First, there has been considerable resistance to compression by so-called XML purists. Second, the W3C XML Binary standards effort has goals that are more general than solving XML's space inefficiency.

The XML purists argue that compression, or any form of binary representation, is contrary to XML's most fundamental tenet; namely, that XML is human readable. This argument is fine insofar as it goes, but it doesn't go very far. Most XML is far too complex in practice to be reliably read by humans. I don't mean that an XML expert with the proper domain knowledge can't read it in principle, but that the average technical person can't do it for at least two reasons. First, XML tags often become a private language specific to a business or even an application, a natural result of its extensibility. XML standards for verticals industries don't eliminate this problem. For example, FIXML uses abbreviations (to improve efficiency) that the average technician won't understand. Second, contrary to the implication that XML's human readability is paramount, XML standards are designed primarily for machine manipulation. This is especially true of Web Services standards, which are so layered and numerous (with more on the way), that trying to keep their grammar straight, let alone understanding their strengths and weaknesses, can sorely try one's patience. Without machine generation, errors in Web Service XML are very likely. Can we understand Web Service XML? Yes, but no one really wants to understand it in production, given its complexity, volume, and variability. More important than these arguments is the fact that human readability is totally irrelevant in precisely the places where XML space efficiency is most important: transport, storage, and processing. Obviously, no human is going to read XML on the wire, on disk, or while it is being manipulated by machine (e.g., during translation).

Making the issue more complex, W3C XML Binary is concerned with more than compression. The purists' simplistic notion of XML falls apart once it's realized that enterprise data has inherently non-textual characteristics. The XML Binary use cases illustrate the many problems facing XML. Somehow, rich XML documents with binary data, such as graphics, embedded fonts, large amounts of floating point analytical or measurement data, and the like, must be supported. Additionally, they want the binary representation to support lossless bi-directional conversion (round-tripping), content update, and efficient interrogation for content-based routing. Such a combination will take some time to deliver, let alone turn into a standard.

It's time to get practical. Though computing power may follow Moore's Law, network bandwidth capacities can't keep up for at least two reasons. First, requirements are driven by both the processing rate and interconnect complexity (proportional to the square of processor quantity). Second, it's just not possible to upgrade physical networks at the same rate as processors and memory. So, while we're waiting for an ideal solution to the XML Binary quest, let's extensibly solve part of the problem. Integration vendors that use XML should provide a plug-in XML compression and decompression service. This service should be invoked at the discretion of the user whenever XML is put on or taken off the wire (consider disk, too). Whenever the W3C XML Binary standard is released, it should be straightforward to upgrade the service. XML in your *enterprise* can have *integrity*... and efficiency, too. **bij**

About the Author

David McGoveran is president of Alternative Technologies. He has more than 25 years of experience with mission-critical applications and has authored numerous technical articles on application integration.
e-Mail: mcgoveran@bijonline.com
Website: www.alternativetech.com