

The Resource for e-Business and Application Integration

November/December 1999  
www.eaijournal.com

**eAI**  
JOURNAL

www.eaijournal.com

# A Brave New Economy

- Competing at Internet Speed
- Extending EAI Beyond the Enterprise
- MQSeries Standards and Guidelines
- Integration Challenges in B2B Commerce
- Integrating SAP R/3





# enterprise integrity



By DAVID MCGOVERAN

## Integration Agents

Architecturally, their purpose is to layer a host of disparate, native interfaces so as to provide common, compatible interfaces. There are lots of names for them — adapters, connectors, interfaces, wrappers, etc. And therein lies significant confusion. The name chosen by a vendor for that piece of software that lies architecturally between the message-handling middleware and either application software or services (such as DBMSes, directories, and analytic or business intelligence engines) is, in my experience, chosen less for technical than for marketing reasons. This practice often, either intentionally or unintentionally, misleads the consumer — an adapter may not adapt to anything, a connector may require development, wrappers may not encapsulate. In short, the words tend to set expectations that are yet to be realized in practice.

Worse than this semantic confusion is the lack of functional correspondence between any two vendors' products. Show me two adapters (I'll pick on "adapters" without prejudice just to save space) from two vendors for any specific packaged application, and I promise, you will have difficulty understanding all the differences in functionality. It doesn't matter whether these EAI components have similar names or not. It doesn't even help to say that the component is CORBA (or COM) compliant — you just don't know which services are or are not supported without going pretty deep.


The worst of all problems with adapters, connectors, and the like is their downright primitive nature. Those of us who have been working with mission-critical or enterprise applications for a while have become accustomed to certain infrastructure requirements. As designers of such software, we incorporate support for RASS (Reliability, Availability, Serviceability, and Scalability) without giving the decision to do so much consideration. Instead, we spend time trying to achieve higher levels of RASS, most often by selecting better infrastructure components (such as platforms, networks, brokers, application servers, and packaged applications) and using redundancy. Sadly, most "EAI experts" — even those with decades of IT experience — focus only on data-level and message-level aspects of adapters. RASS cannot be addressed solely through such functionality.

All these names can be treated as vendor-specific names for the generic concept. The Enterprise Integration Council (EIC) has proposed the term "integration agent" for this type of component, and I shall use it henceforth. Lets consider some of the properties of an ideal integration agent, in essence providing a reference model for *integration agents*. Clearly, it should encapsulate the functionality being provided by the application or service and, therefore, provide a high level of abstraction and imple-

mentation independence. It should connect simply and without shutting down any other component — plug-and-play integration agents lead to software appliances and so dynamic registration is needed. It should, in fact, adapt — if the native interfaces change, the integration agent should not break and should be able to surface any additional functionality. This means it has to support interrogation. For example, an integration agent should be able to both publish and respond to interrogation about metadata. It should be "boundary aware," meaning that transaction boundaries and timing constraints imposed by the integration environment and the application or service should be respected and not masked. It should be active rather than passive, being able to respond to external events and to publish its own events. Most important, it should enhance RASS, not become yet another autonomous potential point of failure.

For RASS to be supported, an integration agent must be fully instrumented and behave as a distributed agent. It should not die with the application or service. System management interfaces that provide performance and health monitoring are essential and are sometimes supported by EAI vendors, although not of the integration agent itself. Few integration agents can respond actively via system management interfaces. For example, it should be possible to remotely start the integration agent and to stop it gracefully, and to use the integration agent to remotely start and stop the application or service. Online self-test, trace, loopback, and loopforward modes can be extremely advantageous when rolling out a complex EAI project, but I have found none of these features in commercial integration agents.

You can anticipate integration agents that address not only data and messages, but process and business performance metrics as well. Whether as properties of the application or service to be monitored and published or as constraints to which it must respond, it is only through the integration agent that we have hope of such coordination.

Primitive or not in its current realization, the integration agent concept is sound. While their functionality is evolving, be acutely aware of the importance of integration agent selection — after all, they are the glue used to integrate the enterprise. If that glue isn't strong and flexible, the resulting enterprise will lack integrity. 

*David McGoveran is president of Alternative Technologies, Inc. He has more than 20 years' experience with mission-critical applications and has authored numerous technical articles on application integration. E-mail: [mcgoveran@alternativetech.com](mailto:mcgoveran@alternativetech.com).*



Discuss "Enterprise Integrity" with David McGoveran at [www.eaiforum.com](http://www.eaiforum.com).