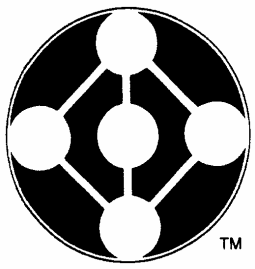

Alternative Technologies

Enterprise Integrity: Convergence

Vol. 6, No. 1

As an independent analyst and strategic planning consultant, I try to keep an eye open for new trends. Among the most interesting trends in the enterprise software market today is that of so-called "convergence." This term has both a marketing and technical usage. In terms of technology, it refers to the convergence of two or more areas of technology that were formerly assumed to be distinct specializations, but which are increasingly being practiced in common. In terms of marketing, convergence refers to the merging of two or more previously distinct market sectors or product categories. It's a little difficult to say which comes first, technology convergence or marketing convergence, but the impact on the consumer and the practitioner is always profound.

Recently, the product categories of application servers and integration suites converged into what some analysts call *application platform suites*. This convergence should not be too surprising. Both categories address common problems: software development, distributed software, component or service integration and reuse, runtime environment management, and so on. Of course, just shoving components of an integration suite and of an application server together doesn't really produce much benefit because the technologies behind these products have typically been driven by disparate conceptual models. Resolving the differences in these conceptual models is an important technological innovation. If (and only if) a common conceptual model underlies the application platform suite's design, consumers benefit by being able to treat all software more uniformly, with development and integration as two aspects of the same activity. The result can be significant improvements in productivity, delivery, staffing, maintenance, and manageability.

Perhaps less obvious is the convergence of ETL (extract, transform, and load) processes and EAI data and message transformation processes. As sources increased in number and data increased in volumes, ETL requirements have transitioned from traditional batch to include more continuous, online, message based processing capabilities. Similarly, many EAI data transformations require some form of staging and synchronization in order to preserve semantic integrity, thereby looking more like ETL capabilities. Perhaps most important is the migration of semantic technologies like data modeling, profiling, cleansing, and ontologies into the EAI domain from their more mature usage in the ETL world. This trend will help bring about the convergence of business intelligence, business activity

monitoring and performance management as both ETL and EAI are driven more and more by the need to manage business processes. And yes, business intelligence activities do belong to business processes.

Another convergence trend exists in the server market. Application server functionality was developed for two key reasons. First, the early simplistic implementations of client/server made it clear that a multi-tier architecture was essential for distributed application scalability. Second, few developers were capable of designing and developing distributed applications that had manageable runtime behavior. With an application server as the deployment environment for an application, application level services such as thread management, scheduling, caching and persistence, transaction management, queuing, connection pooling and management, instance distribution, failover, multi-server coordination, recovery, and the like need not be a detailed consideration of the programmer. These same services are needed by database management systems, Web servers, message brokers, event brokers and process engines, and XML servers. More generally, these services are really not different conceptually from those traditionally supplied by a distributed operating system, albeit at a much higher level of abstraction.

I have sometimes despaired of the nonchalant conflating of various process related technologies and philosophies. Nonetheless, that they are so often confused will drive a kind of ad-hoc convergence among process model driven development, process integration, process automation, and Web services orchestration. As the model-driven development facilities of application platform suites becomes more sophisticated, the convergence of business process models, component orchestration models, and workflow models will mature. The end result of this convergence is quite clearly the kind of business services orchestration that I originally specified as the goal of business process management, fully capable of being specified and directed by business analysts and planners rather than developers.

All this convergence portends that something bigger, something that we can only now just barely glimpse through a glass darkly. I believe a new kind of high level, computer-enabled operating environment for business operations is emerging. This *business operating system* will require both business conductors who control direction, speed, and the acquisition of additional resources, and resources personnel who create, acquire, supply, or replace technology, materials, equipment, and personnel resources in the background. Perhaps, with foresight, it will enhance *enterprise integrity.*