Preface

It is clear that we now live in a world where one expects everything to be connected, or for want of a better word, integrated. For example, when I get onto the Web and do my Internet banking, I expect to be integrated directly with my bank to see exactly how much money I have. Similarly, when I want to book air tickets online, I expect to be integrated into the airline reservation system to see what flights are available, and whether they are fully booked or not. Businesses too are demanding this kind of close integration. A supermarket, for example, that wants to match demand and supply needs to integrate its own stock control system with that of its suppliers so that goods being taken off the shelf are quickly replenished.

Increasingly then, there is a need for organizations to provide services in an integrated fashion. This, however, is easier said than done. Integration cannot be achieved unless the information technology used by the organization is also integrated. Unfortunately, organizations have had a long history of creating systems that operate in isolation from other systems. From conception, many of these systems have been designed to address the specific requirements in a particular functional area such as accounting, or personnel and stock control. This functional orientation, however, has tended to reinforce departmental silos within the organization, resulting in an IT architecture characterized by numerous "islands of applications" that remain quite separate from each other (Sawhney, 2001).

Of course, integration is not an entirely new problem. Many organizations have undertaken specific projects in the past where they had to integrate a number of different systems. The traditional approach to integration, often referred to as point-to-point integration (Linthicum, 2001), has tended to involve crafting custom interfaces between two systems, for example, System A and System B. However, when System A needs to share information with System C, another set of interfaces needs to be created between System A and System C. Similarly, when System B needs to communicate with System C, then another set of interfaces is created. In such an approach to integration, a new set of interfaces is created for each pair of systems that need to communicate. Unfortunately, such a piecemeal approach does not scale well when tens, even hundreds, of individual systems need to be integrated as is often the case in large organizations. Organizations that have attempted to use the point-to-point approach for large-scale integration have typically ended up with tangled webs of integration interfaces in which the high cost of maintaining such a large number of interfaces has become a burden on IT budgets.

In short, the challenge faced by organizations today is to find scalable and economical solutions to the problem of large-scale integration (Sharif, Elliman, Love, & Badii, 2004). This has given rise to the term enterprise integration (EI), which denotes the need to integrate a large number of systems that may be highly distributed across different parts of the organization.

PAST SOLUTIONS

This is not to say that solutions to the challenge of large-scale integration have not been proposed in the past. In the early ’90s, distributed objects and component architectures were mooted as the answer to the challenge of integration (Digre, 1998). This was typified by the Common Object Request Broker Architecture (CORBA), which provided an open standard for distributed systems to communicate (Vinoski, 1997). However, CORBA projects were perceived as technically very complex, requiring significant development effort (Henning, 2006). Furthermore, industry support and standardization efforts for CORBA proved problematic, leading to its slow demise. More recently, enterprise resource planning (ERP) solutions, such as SAP and Peoplesoft, which involve replacing existing systems with a suite of interconnected modular systems from a single vendor, were seen as the answer to systems integration (Davenport, 1998). However, organizations soon realized that no single ERP system could address all the requirements of an organization (Hong & Kim, 2002). In fact, an ERP system often needed to coexist with existing systems, therefore heightening, rather than removing, the need for integration (Themistocleous, Irani, & O’Keefe, 2001).

However, it is not just the scale of integration that is challenging. The integration of IT systems is also severely hampered by the technical complexities of integration (Lam, 2004). For one, a particular system may have been developed on a technology platform that is, at worse, incompatible with the technology platforms used by other systems. In the ’70s and ’80s, for example, many systems were developed on what is now considered legacy mainframe technology, while the ’90s saw the adoption and proliferation of Internet and Web-based technologies. Therefore, most organizations have, over time, ended up with a portfolio of systems based on a diverse mix of technologies. It must also be mentioned that many legacy systems were originally conceived to serve as stand-alone systems with little intent for integration. Such systems, which represent a huge financial investment to the organization, tend to become operationally embroiled within the organization, which makes them difficult to replace with newer, more modern systems (Robertson, 1997).

PROMISING DEVELOPMENTS

Recently, however, there have been some promising developments within the IT industry that, going forward, offer the potential for easing the technical challenge of integrating systems in the future. These developments center on the adoption of Web services (Cerami, 2002) as a standard for open communication over the Internet. In short, Web services enable systems to communicate to other systems over the Internet or, as the case may be, an intranet. For this to happen, systems must expose, as services, the functionality they wish to make available to other systems. A stock system, for example, may expose a “check current stock” service to other systems so that they can check, in real time, the current stock levels of a particular product. One can imagine how this might help buyers and suppliers coordinate the supply chain much more effectively. However there are several barriers, though insignificant, to the adoption of Web services. One of these barriers is the immaturity of the standards (Bloomberg & Schmelze, 2002). The standards relating to Web services are relatively new and continue to evolve at a rapid pace. For obvious reasons, organizations are often loath to make substantial technology investments relating to standards that may be quickly superseded. Other barriers include concerns over performance and reliability. Because Web services rely on the Web, performance and reliability cannot be guaranteed, nor are Web services generally suitable for high-volume transaction processing. For these reasons alone, Web services may not be a suitable technical solution to systems integration in mission-critical

business processing. In addition, while it might be useful to design new systems with Web services in mind, existing systems may need to be substantially reengineered in order to conform to a Web services model. So, while Web services are certainly a promising technical solution to systems integration, they are by no means a complete one.

Another interesting development within the IT industry is that of the service-oriented architecture (SOA). SOA (He, 2003) is something that has piggybacked off the Web services bandwagon, but in the last 3 years or so, has gained a momentum of its own. In an SOA view of the world, systems expose their functionality as a set of services, typically as a set of Web services. It does not matter which technology platform the system sits on or what development language the system has been written in as the services are defined in a way that other systems can readily understand. In fact, other systems do not need to worry or necessarily care about how these services are actually implemented. These services can either be public or published in nature. If services are public, they are typically known to a limited set of other systems, such as in the case of a corporate intranet. If services are published, however, details of the services are registered in a directory from which other systems, including those external to the organization, can discover and use the services. In essence, in an SOA, there is a network of loosely coupled systems where a complex business function may be implemented by calling upon the services offered by other systems.

With SOA, the issue of integration is no longer problematic so long as every system conforms to the SOA model and is able to offer its functionality through a set of defined services. That, unfortunately, is the point where SOA suffers the same adoption problems as Web services, with which it is closely aligned. If organizations could start with a clean sheet again, they would probably develop their systems to conform to an SOA model. In reality, however, organizations need to manage, and live with, at least in the immediate and near future, the huge investments they have already made in existing legacy systems. So SOA and Web services are not the silver bullets that will resolve all IT woes, although some would like to believe otherwise. They clearly offer a pathway, or at a least a direction, for resolving many integration issues, but are not a solution that can be readily implemented by organizations today.

ENTERPRISE APPLICATION INTEGRATION

Fortunately, what might serve as a more complete and holistic solution to enterprise integration are the enterprise application integration (EAI) tools being marketed by integration vendors such as Webmethods, TIBCO, IBM, Microsoft, Seebeyond, BEA, Mercator, and Vitria (McKeen & Smith, 2002). Such EAI tools did not appear overnight but evolved from the message-oriented middleware (MOM) tools that became popular as a means of providing high-volume, reliable communications between systems. In general, EAI tools have three main components. The first is an integration broker that serves as a hub for intersystem communication. The integration broker performs a number of functions such as multiformat translation, transaction management, monitoring, and auditing. The second is a set of adapters that enables different systems to interface with the integration broker. An adapter is essentially a gateway or wrapper that provides the means by which packaged applications (such as SAP), database applications (such as Oracle), legacy systems (such as mainframe), and custom applications (written in Java or another programming language) can connect to the integration broker (Brodie & Stonebraker, 1995). The third component is an underlying communications infrastructure, such as a reliable high-speed network, which enables systems to communicate with each other using a variety of different protocols.

Although EAI has, until now, occupied a rather niche market, the growing importance of enterprise integration can only mean that the size of the EAI market will expand. One can also observe a growing

sophistication and maturity in EAI tools. One area of interest, for example, is that of business-process management (BPM). The reason for progress here is because the motivation behind many integration projects is to support business processes that span across different parts of an organization. An e-business transaction, for instance, may begin at the order-entry system, but such transactional information may be passed onto an account-management, payment, logistics, and then eventually a delivery-order system as part of broader business-process flow. Hence, the integration of these systems is driven by business-process needs. Some EAI tools are therefore beginning to include BPM tools that enable the modeling of business processes in a graphical format. These business-process models are subsequently linked to calls or operations that initiate processing within a system. As it turns out, few organizations have their business processes so rigorously defined and mapped out. The introduction of BPM tools therefore provides a timely and pertinent reason for organizations to revisit their business processes.

Another related area of growing sophistication in EAI tools is that of business activity monitoring (BAM). BAM is about monitoring the health of activities within a business process. If, for example, a business process fails for some reason, this will be picked up by the BAM tool and an appropriate alert can be raised. BAM can also be used to identify bottlenecks within a process by tracking throughput and assessing the rate at which the different activities within a business process are successfully completed. Clearly, BAM tools, and for that matter, BPM tools, are particularly well-suited to organizations that have business processes that are, or are inclined to be, heavily automated.

So, EAI tools are themselves evolving, and what one is beginning to see is a closer alignment between technical integration, which relates to how systems talk to each other, and business integration, which relates to why systems need to talk to each other. At the same time, business integration is being facilitated by the increasing standardization within the business integration space and convergence by EAI vendors in working toward these standards. Central to this effort is the business process modeling language (BPML 1.0) standard, developed under the auspices of the Business Process Management Initiative (http://www.BPMI.org), which provides a formal model for describing any executable end-to-end business process. In theory, if all organizations described their business processes in BPML, they would find it much easier to collaborate.

STRATEGY

Aside from technology and process issues, the other important element of enterprise integration is the strategic element. Spending thousands of dollars on EAI tools and teams of individuals modeling business processes does not necessarily mean that organizations will solve their enterprise integration problems. One missing piece from all this, like any other major endeavor, is the importance of strategic direction and senior-management support (Lam, 2005). Enterprise integration is something that affects many different parts of the organization; the problem is not confined to one particular part of the organization. As such, success can only be achieved if the various departments buy into enterprise integration and share the same vision of how to achieve it. This, of course, is easier said than done, and there are several challenges. One of the challenges is the fact that, in a large organization, individual departments may be used to operating autonomously, with minimal interaction and engagement with each other. The notion of working with other departments, or at least coordinating their technology strategies, is something that may appear quite novel. At worst, the endeavor becomes a political battle, where certain divisions are seen to be vying for control, encroaching on the space occupied by others. This, of course, is not something that is peculiar to enterprise integration projects, but is seen any large project that involves different divisions within an organization working in new ways.

Importantly, each department must believe that there is a case for enterprise integration, either financially in terms of reducing the costs of systems integration, or from a business perspective in terms of enabling new business processes or enhancing business performance. Unfortunately, individual departments, by their nature, have a localized view of their integration needs. Getting the bigger picture of an organization's enterprise integration needs is a message that must be communicated to individual departments so that they understand the rationale for a strategic perspective. Of course, if one left each department to its own devices to develop its own solution to its own set of integration problems, there would be much reinvention of the wheel and wasted time and effort. A single strategic integration solution therefore makes much more sense, where an organization can address the integration requirements in different parts of the organization in a cost-effective and controlled manner. It certainly does not make sense for each department to purchase its own EAI tool, for example.

Another thing to bear in mind is that enterprise integration is not something that can take place overnight. Enterprise integration is a long-term initiative that, in some cases, may take years to complete depending upon the number of systems to be integrated and the complexity of the integration challenge. Organizations therefore need to think carefully about how to plan and roll out the enterprise integration initiative. One approach would be to identify the high-priority integration projects within the organization where the urgency or potential business returns from integration are greater. Such successful projects could then serve to reinforce the case for integration and perhaps even provide inspiration for further integration projects. Another more risk-adverse approach would be to identify pilot projects that could serve as the grounds for organizations to build up expertise and knowledge of enterprise integration before tackling larger and more substantial projects. Such a more cautious strategy might suit organizations with little prior experience with enterprise integration. It might also be wise to consider integration projects in parallel with other business improvement projects that, in turn, can help shape the integration project. A good example is business-process reengineering, where it does not make sense to automate a process that is intrinsically inefficient or problematic, but where an opportunity presents itself to make broader organizational changes. In fact, from an organizational perspective, information systems integration involves changes in business processes, and more broadly, a realignment of technology goals with business goals (Themistocleous et al., 2001).

To sum up, enterprise integration has become a strategic enabler for many of the business initiatives organizations are implementing or wish to embark on, whether it is supply chain management, customer relationship management, e-business, or simply more efficient ways of business processing. The traditional methods of systems integration have not proved to be scalable, so solutions are needed that can address both the scale and complexity of integration. Enterprise integration, however, is not just about technology integration, it is also about process and business integration, and so may involve a reconceptualization of how organizations work and do business.

ORGANIZATION OF THE BOOK

This book is organized into four main sections, each of which has a number of chapters. The first section is entitled "Managing Enterprise Integration" and contains the following chapters.

Chapter I examines the evolution of enterprise integration and its growing importance amongst organizations. The chapter provides a good overview of the benefits of enterprise integration and some of the challenges associated with its adoption.

Chapter II looks at five critical success factors for enterprise application integration, namely, minimal project expense, optimal reuse of components, complexity reduction, optimal coupling of applications,

and minimal cost of EAI infrastructure. The authors conclude that the success factors are closely integrated, and they develop from this a number of hypotheses.

Chapter III explores how social and organizational aspects can affect ERP projects and their ability to integrate different parts of the enterprise. The chapter draws from the successful case of a multinational ERP implementation in a large international organization and concludes with the recommendation that one should examine the roles of all actors with the power to affect not only the implementation project but also the system being implemented.

Chapter IV discusses experiences in the acquisition of software-intensive systems prior to establishing a contract. One significant challenge is the identification of functional requirements, and the authors contend that business-process modeling is an effective way to explicitly define requirements while creating visibility and consensus among different stakeholders of major IT projects.

The second section is entitled "Business-Process Management" and contains the following chapters.

Chapter V presents an approach to constructing enterprise processes based on the integration of component processes. In this approach, process representations at the logical as well as physical levels are depicted in a hierarchy, and a graphical tool is used to support enterprise process construction.

Chapter VI explores, through a case study, the use of XML (extensible markup language) and Web services in conjunction with an integration broker to provide a B2B (business-to-business) solution. The authors argue that existing B2B solutions, largely based around EDI (electronic data interchange), present a high entry barrier to smaller organizations, and that the use of XML and Web services provides an alternative lower cost B2B solution that may be more suitable.

Chapter VII provides an overview of business-process management. The chapter describes how systems development has changed from being largely implementation oriented to now being analysis oriented, and suggests that business-process management standards and technologies will drive the transition to more service-oriented IT architectures in the future.

Chapter VIII describes the importance of metamodels, particularly in relation to knowledge-intensive service industries. The key guiding principle is to define the process model at the logical level, free from any technical implementation. Business-process integration is then a matter of achieving the best possible overall physical engine to implement that process model from available legacy applications, applied investment opportunities, and expert development resources.

The third section is entitled "Integration Methods and Tools" and contains the following chapters.

Chapter IX presents a methodology for the creation of EAI solutions based on the concepts of software product-line engineering. The overall idea is to view the external applications with which a given system wants to integrate as a family of systems. In this way, the flexibility required by EAI applications can be assured.

Chapter X demonstrates a visual tool for automatic supply chain integration. By clicking on the appropriate tables, functions, and fields, users can specify the data needed for integration. Integration code can then be automatically generated using Web services.

Chapter XI presents an integration framework called DAVINCI aimed at providing mobile workers with mobile software applications to query and update information coming from different heterogeneous databases. The chapter describes the trials developed using the DAVINCI architecture in different European countries.

Chapter XII describes the basic notions of intelligent agents and multiagent systems, and proposes their possible applications to enterprise integration. Agent-based approaches to enterprise application

integration are considered from three points of view: (a) using an agent as a wrapper of an application or service execution, (b) constructing a multiagent organization within which agents are interacting and providing emergent solutions to enterprise problems, and (c) using the agent as an intelligent handler of heterogeneous data resources in an open environment.

Chapter XIII describes an attempt to introduce semantics to workflow-based composition. A composition framework is presented based on a hybrid solution that merges the benefits of the practicality of use and adoption popularity of workflow-based composition with the advantage of using semantic descriptions to aid both service developers and composers in the composition process and facilitate the dynamic integration of Web services into it.

The fourth and final section is entitled "Enterprise-Integration Case Studies" and contains the following chapters.

Chapter XIV examines common EI challenges and outlines approaches for combining EI and process improvement based on the process improvement experiences of banks in several different countries. Common EI-related process improvement challenges are poor usability within the user desktop environment, a lack of network-based services, and data collection and management limitations. How EI affects each of these areas is addressed, highlighting specific examples of how these issues present themselves in system environments.

Chapter XV explores the gradual evolution of SOA through various phases and highlights some of the approaches and best practices that have evolved out of real-world implementations in regional retail banks. Starting from a step-by-step approach to embrace SOA, the chapter details some typical challenges that creep up as the usage of an SOA platform becomes more and more mature. Also, certain tips and techniques that will help maximize the benefits of enterprise-wide SOA are discussed.

Chapter XVI describes a case study of application integration in a Korean bank. The case examines the integration technology employed and the adoption of the technology in a pilot project. The chapter highlights some of the managerial implications of application integration and discusses the broader organizational impact of application integration.

Chapter XVII uses the example of a retail business information system to illustrate the concept of service-oriented development and integration in a number of different scenarios. The chapter shows how using a service-oriented architecture allows businesses to build on top of legacy applications or construct new applications in order to take advantage of the power of Web services.

Chapter XVIII discusses the adoption and potential usages of RFID (radio frequency identification) in the case of enterprise integration projects such as supply chain management. Through electronic tagging, RFID can enable stocks to be monitored at a level that was previously not practical or cost effective.

REFERENCES

Bloomberg, J., & Schmelze, R. (2002). *The pros and cons of Web services* (ZapThink Report). Retrieved from http://www.zapthink.com/report.html?id=ZTR-WS102

Cerami, E. (2002). *Web services essentials.* Sebastopol, CA: O'Reilly.

Davenport, T. A. (1998). Putting the enterprise into the enterprise system. *Harvard Business Review, 76*(4), 121-131.

Digre, T. (1998). Business object component architecture. *IEEE Software, 15*(5), 60-69.

He, H. (2003). *What is service-oriented architecture.* Retrieved from http://www.xml.com/pub/a/ws/2003/09/30/soa.html

Henning, M. (2006). The rise and fall of CORBA. *ACM Queue, 4*(5). Retrieved from http://acmqueue.com/modules.php?name=Content&pa=showpage&pid=396&page=1

Hong, K. K., & Kim, Y. G. (2002). The critical success factors for ERP implementation: An organizational fit perspective. *Information & Management, 40*, 25-40.

Lam, W. (2004). Technical risk management for enterprise integration projects. *Communications of the Association for Information Systems, 13*, 290-315.

Lam, W. (2005). Exploring success factors in enterprise application integration: A case-driven analysis. *European Journal of Information Systems, 14*(2), 175-187.

Linthicum, D. (2001). *B2B application integration.* Reading, MA: Addison Wesley.

McKeen, J. D., & Smith, H. A. (2002). New developments in practice II: Enterprise application integration. *Communications of the Association for Information Systems, 8*, 451-466.

Robertson, P. (1997). Integrating legacy systems with modern corporate applications. *Communications of the ACM, 40*(5).

Sawhney, M. (2001). Don't homogenize, synchronize. *Harvard Business Review*.

Sharif, A. M., Elliman, T., Love, P. E. D., & Badii, A. (2004). Integrating the IS with the enterprise: Key EAI research challenges. *The Journal of Enterprise Information Management, 17*(2), 164-170.

Themistocleous, M., Irani, Z., & O'Keefe, R. (2001). ERP and application integration. *Business Process Management Journal, 7*(3).

Vinoski, S. (1997). CORBA: Integrating diverse applications within distributed heterogeneous environments. *IEEE Communications Magazine, 35*(2), 46-55.