Enterprise Application Integration in the electronic commerce world

Naveen Erasala, David C. Yen*, T.M. Rajkumar

Department of Decision Sciences and MIS, Richard T. Farmer School of Business Administration, Miami University, Oxford, OH 45056, USA

Received 4 March 2002; received in revised form 7 October 2002; accepted 15 October 2002

Abstract

Electronic commerce is radically changing the competitive landscape. Increasingly, firms need to have stronger ties with customers and suppliers. In this business scenario, it is important for firms to have information systems (IS) that encourage the unhindered flow of information. The paradigm that addresses this need of firms is popularly known as Enterprise Application Integration (EAI). In this paper, we examine EAI in some detail and develop a case for EAI in today’s business environment. We discuss various ways in which EAI can be realized and illustrate how firms can improve gains by implementing the appropriate EAI solution. It concludes with a discussion about the future directions that EAI might take.

Keywords: Enterprise Application Integration (EAI); Electronic commerce; Extensible Markup Language (XML); Legacy systems; Client/server systems

1. Introduction

The advent of the Internet has changed the concept of the traditional market. Computer networks link customers and firms across the world in ways never seen before. Both buyers and sellers offer each other tremendous opportunities. Customers are being offered more choices and means to easily evaluate these choices. They can buy goods over the Internet from the comfort of their homes and have the goods delivered at their doorstep. They can buy from any vendor who can sell on the Internet, irrespective of where the vendor is physically located. All that matters to the global customer is the product offering and the value proposition accompanying it.

The same information technology that has shifted the balance of power to consumers has also offered tremendous cost-saving potential for firms. Firms are in a position to derive improved efficiencies by better supply chain and intra-organizational integration using the new information environment. That environment has opened up a new marketing channel to promote, sell, interact with customers, and provide better customer support. All these promised benefits have resulted in a frenetic race between firms to “get there” first, offer customers a better package, and win them over before the competition does. This heated, competitive landscape has created new strategic imperatives for firms to move towards greater speed, flexibility, and cost reduction.
These imperatives are casting information systems (IS) in an increasingly strategic role. For example, a website that accepts an order without telling the user that the item has a lead-time of 8 weeks is giving poor customer service and may well result in customers going over to competition [29]. This situation represents a threat as well as an opportunity. If the firm can use its information systems effectively, it cannot only provide lead-time information but also differentiate itself by providing other value-added features such as improved customization, order tracking, and shipping intimation. However, in order to achieve this capability, firms need to create an unhindered information flow across the organization. Unfortunately, given the disparate nature of information systems in most firms, this becomes a daunting task.

Over the years, firms have come to depend upon a mixed bag of information systems. For decades, systems have been built to serve a given purpose for a given set of users without sufficient efforts to enable better communication between them. Business processes and information have often been maintained in legacy applications and held in unstructured databases, residing on old technologies or in packaged solutions. These have been extremely complex but have delivered the needs of the business. As new technologies have continued to be introduced, “islands of information” have emerged, each associated with its respective technology and functional stovepipe. These islands, in most cases, could not effectively exchange their contents, thus hindering information flow within the organization. But, given the emerging competitive landscape, it became essential for firms to find ways to circumvent this hurdle. The need of the hour is to be able to share information and business processes without having to make sweeping changes to existing applications and data structures. A promising approach to achieve this is Enterprise Application Integration (EAI).

At the most basic level, EAI represents the unrestricted sharing of information and business processes among all connected information systems in an enterprise [11,13]. It represents an attractive proposition to firms, since it offers them the opportunity to leverage their systems into a seamless chain of processes and present a unified view of their business to customers. When such leveraging and presentation occurs, firms can capitalize on the opportunities offered by electronic commerce because they can efficiently interact with customers and suppliers on a consistent basis.

This paper is divided into five sections. Section 2 presents an overview of EAI. It contains a brief historical perspective illustrating the need for EAI, what it means, and why it is important to firms. Section 3 presents a microanalysis of EAI. It compares the approaches taken by firms and developers towards integration and goes on to identify the characteristics of EAI, at what levels integration can be accomplished, and what methods are being used for EAI. This is followed by a macro analysis (Section 4), which focuses on managerial, developmental and implementation issues. Section 5 discusses the current state of EAI and its future.

2. EAI: an overview

In the early days of computing, information was processed on centralized platforms. As a result, processes and data existed in a homogeneous environment. Integrating applications usually just needed additional coding under these processing circumstances. As technology developed, platforms changed, and new, smaller and more open platforms such as UNIX and Windows NT emerged. Also, new programming paradigms, such as object-oriented, component-based development and packaged applications like enterprise resource planning (ERP) solutions grew in popularity and importance. In a rush to incorporate these new systems, most firms did not show foresight in choosing the right architecture. IT decisions were made at department levels, with each department choosing technologies and solutions based on its own needs and beliefs. As a result, the enterprise as a whole was left with a collection of systems that were extremely difficult to integrate. These technologies, even today, provide some value to the firm, but their value is diminished since they cannot leverage other enterprise applications [11,13].

2.1. EAI drivers

There are three main factors driving the use of EAI by organizations.

E-commerce: A key question facing businesses today is how to seize the opportunities presented by
e-commerce with minimum risk and maximum utilization of current IS investments. EAI offers an answer to this question. E-commerce necessitates integrating business processes such as procurement, sales order processing, customer service, customer support, and supply chain management, and so forth that exist between companies, their partners, and their customers. There is a need for integrating on a business-process level, not just within firms but also between firms that make up the extended enterprise. These cannot occur without the integration of internal systems that provide functionality to web applications. In the end, it often comes down to integration of transactions via the Internet, intranets, or extranets, and it "involves integration of legacy applications and exposing them to easy-to-use front-ends that are Web-based" [25].

Once applications are integrated, EAI enables a firm to improve customer relationships as it can get a better integrated picture of customers. Customers, in turn, perceive the enterprise as an integrated business rather than individual departments. This provides for better customer loyalty in the long run.

Companies achieve advantages in the E-commerce arena by reducing the time-to-market for new service offerings. EAI helps with quicker rollouts of service offerings by taking advantage of the knowledge in existing legacy applications. IT departments do not have to develop/debug existing applications. With EAI, developers simply have to integrate the diverse applications, add web front-ends, and roll out new services.

**Mergers and consolidations:** Observers in recent years have found significant activity of mergers and acquisitions. These mergers either have the business goals of expanding their business coverage and service offerings while reducing costs and streamlining their services by consolidating their activities and eliminating redundancies. Researchers find that applications in merged companies typically run on different platforms and have trouble consolidating these diverse applications and platforms in an inexpensive manner. EAI helps solve their problems [5,7].

**ERP packages:** The rise of ERP package use by companies has also generated the need for EAI. It is common to see companies choose one ERP vendor for financials while choosing another for human resources applications. While the data within individual ERP systems have been integrated, it is not easy to integrate ERP products with other prepackaged applications that are not customized for integration. Integration with existing legacy applications is also hard.

Firms have been wrestling with the idea of integration over the past several years. Forrester Research estimates that 35% of development time is devoted to creating interfaces and points of integration for applications and data sources. This problem is painfully illustrated with traditional client/server systems—what was inexpensive to build was expensive to integrate and difficult to maintain [3]. As firms moved towards business processes rather than departmental applications, the need to integrate intensified further and firms started to spend increasing amounts of revenue to meet these needs. A report by the Aberdeen Group [3] indicates a 100% increase in revenues from extended enterprise processes from 1998 to 2000. The same period shows a 150% growth in revenues from process flows. It is clear, then, that the need for integration is pressing and EAI is not just a buzzword but a solution to a very real problem.

A Conspectus group survey (1999) notes that despite extensive investment in state-of-the-art application software, many firms relied on legacy systems for mission-critical applications. Over a third of the survey respondents considered it necessary to maintain legacy systems rather than to phase in new ones. The problem of integrating with legacy systems appeared as one of the major inhibitors to new IT development. However, it may be that the focus on e-business and enterprise integration is providing organizations with the perfect opportunity to link their legacy systems and state-of-the-art applications within a web-enabled enterprise [22].

### 2.2. Definitions

There are several definitions of EAI in literature, and although many authors define it distinctly from others, the common theme of integration runs through them all. Green defines EAI as the integration of dissimilar application systems to share information via a common user interface [8]. Linthicum [11,13] takes the process view and defines EAI as the unrestricted sharing of information and business processes among all connected information systems in the enterprise. Taylor [28] feels that EAI is moving IS towards
a platform for supporting electronic commerce. We integrate the different perspectives. We define EAI as follows: the integration of applications that enables information sharing and business processes, both of which result in efficient operations and flexible delivery of business services to the customer. Implementing EAI does not invariably involve discarding current or legacy applications; rather it unlocks the value of these applications and deploys their functionality in a scalable and robust way.

2.3. Characteristics of EAI

At the core of EAI is the need to share and manage information, but the methods of doing this could be complex and varied. The driving focus of any EAI approach is to extend useful information to all and to avoid information islands and bottlenecks. Given the enterprise-wide connectivity that EAI systems seek to accomplish, they may be expected to have certain basic characteristics and criteria. Green [8] has identified five such characteristics:

1. EAI systems are directed towards integration at the business level—they can include all business and data processes.
2. EAI methods re-use and fully distribute all applicable business processes and data.
3. EAI methods involve no real understanding of specific system functions to integrate the applications. This ability to integrate with minimal understanding of specific functions is due to the technology’s focus on the user side of the application, which requires no system knowledge.
4. EAI technology does not require source code or code administration rights to any of the target applications.
5. EAI technology generally requires no changes to the existing hardware infrastructure.

Interactivity between systems can be achieved either by invasive methods, non-invasive methods, or a combination of the two. Invasive methods involve direct interaction with the legacy system, or the target application at a source level. This can be useful if changes and enhancements to the business logic are proposed. But, occasionally, there can be operational issues with the existing application or other integration issues concerned with the legacy system itself. Therefore, this is generally looked on as a higher-risk strategy. Non-invasive methods establish an interaction between systems via a direct terminal links. This approach usually does not cause operational issues and is valuable if the existing legacy application is currently meeting all operational needs. It presents the least operational risk.

3. EAI—a microanalysis

3.1. EAI architecture

Many EAI products provide the functionalities in the layers shown in Fig. 1 [7,14,21]. The communication services layer provides for the transportation of data. This layer may be based on remote procedure calls but is more often based on a message queuing system, which can send the messages synchronously or asynchronously with a store-and-forward capability. IT components are normally connected to the communication services layer using adapters/connector that provide the mapping between the component and the EAI software.

<table>
<thead>
<tr>
<th>Process Management (Business Rules, Business Process)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transformation Services (Data Transformation)</td>
</tr>
<tr>
<td>Distribution Services (Message Routing)</td>
</tr>
<tr>
<td>Communication Services (Data Transport)</td>
</tr>
</tbody>
</table>

Fig. 1. EAI architecture.
The distribution layer is responsible for information distribution (routing the messages) to the various components of the application. It operates in either a request/response (point–point) or publish/subscribe mode.

The input data an IT component may require is not necessarily in the format that the component can handle. The transformation services layer provides for such translation and validation, if any. Inputs required for a component may be also coming from more than one source. It is likely that such input will arrive at different times. The transformation services provide any synchronization that is necessary before sending all the input to the component. Transactional integrity is also maintained during any transformation.

Business rules are important parts of business definitions and govern the business process flow. For example, if a customer is refinancing a house loan and reducing the term from 30 to 15 years, the new payment cannot exceed 120% of the old payment. If it exceeds, then additional processing has to take place. The process management layer provides the functionalities for mapping business rules and supporting the business processes. Essentially, this layer coordinates and controls the transformational services layer to complete the business process. The process management layer provides for the management of complete business processes and enables integration of intra- and inter-organizational processes. The functionalities in this layer enable enterprise workflow. They allow you to assemble and reassemble components quickly, as needed, to create new processes and reduce the time to market for product developments.

3.2. The relationship with middleware

The traditional approach to integration has been through program code that can alter a pair of systems so that they can communicate with each other. Popularly known as middleware, this approach can only allow one-to-one connectivity and it is mostly unaware of other similar transactions taking place within the same IT environment. Middleware is essentially technology- and application-dependent and provides little or no visibility of business processes [3,11,13]. Middleware is traditionally viewed as being composed of only the first two layers (Fig. 1). Middleware involves no knowledge of process management functions [21,30]. This gives rise to problems or restrictions actually caused by the integration method itself. Any attempt to link additional systems through middleware quickly became a complex tangle of links with no central control and negligible management. EAI, on the other hand, is not a ‘point-to-point’ solution but one that focuses on the holistic needs of the enterprise business process. By focusing on a many-to-many solution, EAI creates a common way for both business processes and data to speak to one another across applications [11,13]. With a hub-and-spoke technology, users need to specify the interfaces between applications and EAI software, rather than between each application pair. Fig. 2 [19] illustrates this with an E-commerce example. Once an order comes in via the web, phone, mail, or in person, customer information is captured and distributed to different applications. When the order is validated (credit, items, etc.), it is sent through an order fulfillment process. The order fulfillment process may schedule the order for manufacture, fill it from inventory, or forward it to the warehouse for fulfillment. Fulfillment returns status and shipment entries to an order entry system and to a customer service/call center that need to know about outstanding orders. Each application has an interface with the EAI software and not with other applications. Fig. 3 shows the same application using traditional middleware and its one-to-one linkages. The complexity of such systems and the benefits of EAI’s “hub and spoke” paradigm become immediately apparent.

However, middleware has several advantages. It can hide the complexities of the source and target systems, thereby enabling developers to concentrate more on information sharing than on low-level interfacing. In a sense, the connective capability of middleware makes it the ideal solution for moving information between applications. But it is not of much use in the traditional point-to-point form (e.g. Remote Procedure Calls and Message Oriented Middleware) for reasons mentioned above.

New forms of middleware have evolved recently. Linthicum [11,13] identifies several different types of middleware; each designed to solve a specific set of integration problems. He argues that eventually, middleware capable of establishing a many-to-many
connection will become the engine of EAI. Solutions developed using such middleware can typically provide several desirable features such as secure and reliable message transfer, data transformation and broadcasting, and communications. Such a solution can also act as a transaction monitor and safely carry out parallel database updates across multiple systems.

Another compelling reason for using middleware is that the solution can effectively address E-commerce needs. Middleware products such as application servers can act as hosts for groups of components, and when a certain number of clients are accessing the components, they are capable of automatically shifting the workload on to alternative hardware—a process known as load balancing. In addition, if an individual application server fails, others can take over. Thus, the right middleware can grow with the organization delivering both scalability and reliability.

3.3. Integration with existing systems/applications

Depending on the level at which integration is desired, several ways to achieve integration have been envisaged (Fig. 4). Some of these are described below [15,20].

Screen scraping: This uses emulation software to map terminal session screens from a mainframe application and place it into an appropriate user interface technology such as HTML. This approach has the advantage of representing the ‘green screen’ in legacy systems in a much more presentable user interface technology and provides an opportunity for re-engineering the user interface.
Interface redesign: This is a much more complex and invasive approach, which involves encapsulating the legacy code with a new application programming interface (entry point) where additional functionality can be added and may change some user and system processes. This, too, can be expensive but will allow the original business logic to be re-used and provides better integration at the procedural level.

Object integration: The legacy code is wrapped with an interface encapsulating the application’s functionality. The wrapper translates the application’s inputs and outputs via its methods. The EAI software can then allow these methods to be used by other applications and hide all the details of the actual application. A better but more costly option is to translate the legacy code to component-based languages such as Component Object Model (COM), Common Object Request Broker Architecture (CORBA), or Enterprise JavaBeans.

Legacy migration: This involves capturing the business logic in the legacy application and redesigning it, adding functionalities to it, or deploying it on a new platform. This is a task undertaken in cases where the legacy system is critical to the business process and where an off-the-shelf replacement is not available. Legacy analysis techniques such as software asset management (SAM) or knowledge mining are used extensively in this approach, since they significantly decrease the risk and expense. In addition, the process of transforming the legacy source code to modern languages and architecture makes the task viable in terms of both speed and cost.
3.4. The role of XML in Enterprise Application Integration

The key to any EAI mechanism is the ability to control and distribute information throughout the enterprise and to effectively manage the control and distribution. This warrants the use of a single, commonly accepted data standard for sharing information irrespective of the source or destination of data. One of the currently available solutions offering these characteristics is the Extensible Markup Language (XML).

XML is a simplified subset of Standardized General Markup Language (SGML), the language from which HTML is derived. It differs from HTML in its ability to handle context and dynamic behavior information. While HTML can only display information, XML can also associate a context with the information and handle it accordingly. XML has the ability to associate a schema with its document that allows any application or database to refer to the schema and use the data in the XML document correctly.

Though originally created for the web, XML is quickly finding a place in the EAI world because of its ability to provide a common data exchange format. This feature becomes even more valuable in the age of supply chain integration where information needs to be exchanged between firms. Since XML is text based, it can be easily understood and translated between all platforms. This may restrict the types of data that can be encapsulated using XML, but the advantage of simplicity far outweighs the shortcomings. Another danger is that since the XML standard is in the process of evolution, vendors are looking to recast it using their set of proprietary extensions. This may enable several forms of XML to emerge and defeats the purpose of integration.

Though XML was primarily developed for the Internet, it is good not only for the transmission of data from server to browser, but also for the passing of data from application to application and from machine to machine. Since XML is database-neutral, it can be of great help for interconnecting heterogeneous databases.

Among the other features of XML that make it well suited for EAI and E-commerce are [23]:

- Since XML advocates structured data, any XML-aware application can extract exactly the information it needs.
- XML and HTML can be combined, with each performing the portion of the task that it does best, thus offering stronger web-capabilities.
- The syntax of an XML document instance describes the relationships among the various XML elements, and thus no prior knowledge of sender application is necessary.
- XML enables client-side data manipulation without requesting data from the server each time. Similarly, the same data can be presented differently, say, as a subset, depending on the user’s role with respect to the data (e.g. accounting department sees more details than the purchaser).
- Display of data is independent of structure since XML style sheets can describe how to render the same data on different devices.

Given these useful features, it is clear that the advantages of XML far outweigh its disadvantages, thus making it a preferred solution for EAI in the E-commerce age.
In summary, the major advantage that XML can offer is that XML-based data exchange lets different systems inside and outside enterprise exchange information more freely with existing enterprise systems. However, XML has its own limitations in terms of system performance and security. XML-based data exchange requires the systems to convert a chunk of data into an XML document and needs much more bandwidth [11,13]. For example, A customer record may be only 512 KB in its binary state, but in XML it becomes a text file encapsulating both text and schema information, and typically grows to four times the binary file’s size. In addition, it takes additional time for the XML parser to extract the data. Therefore, there is potential that moving the same information via XML will degrade network performance. To prevent the potential system underperformance, it is imperative to rebuild the application from a synchronous to an asynchronous design [1].

Security is another major concern for XML-based data exchange. XML is an unsecured message, and sensitive information intercepted on the network is easily read. The most widely used security technology-encryption will protect most of the data, and we can encrypt the entire XML document using standard technology. However, the requirement in XML is to keep the data together in its entirety and control the viewing of the subsets of the document, for example, a vendor may need to know the customer address and name, but not the details of the credit card information. Similarly, the bank needs to know the credit card information, but not the customer’s purchases [16]. Standards for encrypting portions of the XML documents are currently being specified by the world wide web’s XML encryption workgroup. In addition, adding security measures will further affect network performance, with larger chunks of information moving from system to system and being encrypted and decrypted on the transit [11,13].

3.5. The relationship with Business–Business E-commerce

Process engineering streamlines and automates processes to improve business efficiency. EAI focuses on streamlining the processes within the company, integrating diverse applications such as ERP, sales force automation, customer relationship management (CRM), legacy applications, etc. Business–Business (B2B) E-commerce focuses on streamlining not only the internal applications but includes external entities such as customers and suppliers in the process. Many EAI techniques, while applicable within one company, do not necessarily translate across companies. For example, while a company might choose a messaging technology to integrate its internal applications, it is not reasonable to expect its trading partners to choose the identical messaging technology to integrate their applications [27]. This certainly becomes aggravated with multiple trading partners. B2B frameworks have evolved to solve this need.

Businesses conduct E-commerce transactions using standards such as electronic data interchange (EDI) or XML, which define data formats for document exchange in B2B frameworks. Since 1970s, EDI has become one of the major technologies for Business-to-Business communication. It has indeed become the dominant technology for the largest companies but has been adopted by less than 5% of all companies. Business in general perceives the cost of EDI as outweighing the benefits due to several reasons such as: high EDI software cost, difficult implementation process due to the single universal standard supported by EDI, expensive proprietary network required by EDI, etc. [2]. On the other hand, the basic XML based communication is less difficult and costly to implement and support than EDI. However, the size of an XML message may raise a problem—an XML message may be 10 times faster than EDI message, which may not be a concern for the small companies but may be a big issue for large companies. In addition, XML based communication may not be as secure as EDI. The comparison between the two technologies is listed in Fig. 5 [17].

B2B frameworks are generic templates that enable businesses to communicate efficiently over the Internet [26]. B2B frameworks not only work with other businesses but must efficiently integrate with existing enterprise applications. Like EAI, they define a generic interface rather than a point–point interface and provide the inter-operability mechanisms between different business entities (Fig. 6). B2B frameworks tend to work at the process management layer of the EAI architecture.
B2B applications focus on business semantics, and the vocabulary used between the partners must be coordinated and well defined. RosettaNet, a specific B2B framework uses industry standard dictionaries to coordinate semantics wherever possible. Consider the case where a product for sale (with an item number, description, and price) is to be made available to other partners. B2B frameworks specify a series of XML exchanges that define how the data is shared and coordinated. In the case of RosettaNet, a Partner Interface Process (PIP) is used to define the dialogue (application-to-application XML content) or XML document interchange required between the partners. The RosettaNet framework also specifies the domain-specific business exchanges (in XML) that are required for the specific business processes. This combination of domain-specific, business-process XML data is tied to a state-aware application (e.g. a Java application, or through an EAI software) that delivers the XML content and reacts to the XML content that it receives. The application performs updates to business systems as a consequence of the exchange, and not by the trading partner invoking the application directly [18].

4. EAI—a macro analysis

In today’s interconnected business world, as firms continue to seek better ways to strengthen their links with suppliers and customers, integration takes on an increasingly strategic role. Firms have to deal with the reality of implementing and maintaining EAI systems. This task has several managerial, financial, technical, and personnel issues that need to be addressed.
Some of the issues involved with implementing and maintaining an EAI solution can be classified under the following headings [29].

4.1. Approaching EAI

Enterprise Application Integration should not be approached as a single “big bang” project. It is a collection of individual integration projects that individually give a payback in a specific area of a business and collectively give returns that are greater than the individual parts. According to the Meta Group, most Global 2000 organizations have over 40 enterprise applications. Although firms may not want to integrate every application, they need an integration platform that allows them to continue to integrate more applications at a lower cost. Custom point-to-point integration is not feasible for more than two or three applications.

Most firms find success through laying down an initial integration platform and then implementing integration between a few systems. Once this is working, more projects can be initiated, building upon the success, platform, and skills created in the initial project.

4.2. Designing EAI solutions

There are a few basic steps that must be followed in designing EAI solutions [14]. Define and catalog business processes, business events, IT components, process rules and messages, and message content.

Business processes: Business processes define project scope and help identify the processes that need support in the enterprise. They will also help identify the application integration requirements.

Business events: The events initiate business processes. Both the format and content of events should be defined.

IT components: A list of IT components that business processes depend on will enable the identification of any component that is missing or must be developed. Any functionality that is required but missing in these components must also be identified.

Process rules and messages: Business rules define the sequence of invocation of IT components. This will help identify the sequence of event flows between applications and the EAI software. The messages that are being passed by these events must also be identified.

Message content: For each message, data requirements must be identified for every component that the message provides information for. The aggregated message must be analyzed to extract common definitions whenever possible.

4.3. Implementation architecture

According to a comparative assessment conducted by Doculabs of six EAI vendors, the integration servers are built around two major physical architectural approaches: a hub-and-spoke model and network bus model [24].

A hub-and-spoke model is akin to a networking star topology. The star topology is one of the most widely used structures for a communication system. The star network was used because it was easy to control, the software is not complex and the transaction flow is simple. Fault isolation is also relatively undemanding in a star network because the line can be isolated to identify the problem [7]. The major advantages of hub-and-spoke model hereby generate from the same logics. In a hub-and-spoke approach, the hub integration server acts as a message broker to control communication, data translation, and process interaction among the connected systems, which as a result minimizes point-to-point integration. If each of \( n \) given systems needs a two-way connection to the other, a total of \( n*(n-1) \) connections are required. On the other hand, in the hub and spoke architecture, a central hub manages the connection to each application, resulting in \( n \) connections rather than \( n*(n-1) \) and thus, a significantly lower cost of integration and maintenance. In addition, from management perspective, the hub-and-spoke approach simplifies administration and maintenance.

However, lack of scalability is the major problem with spoke-and-hub model that requires that all information consumed from source applications process within a single hub server. Linthicum [12] rated this model as resource-constrained since eventually the number of connected systems and the information traffic will saturate the available resources of the server and lead to reduced performance. As the star topology in the networking configuration, the poten-
tial bottlenecks and a single point of failure are the major challenges for the hub-and-spoke model.

Two major innovations have been made on the traditional hub-and-spoke architectures: federated static and federated dynamic structure [12]. The federated static structure allows applications and data sources to connect to a single hub statically and multiple hubs are connected and can exchange information with each other. The hubs do not share the processing loads dynamically. So, while providing better scalability than the single hub-and-spoke architecture, the federated static architecture is not flexible. The Federated dynamic model, in contrast to static, provides hubs with the ability to share the processing load dynamically among all available integration servers. Although it is considered an ideal solution to meet all the challenges faced by the traditional hub-and-spoke model, there are no EAI vendors that support this architecture due to its complexity and sophistication.

The physical implementation of the bus model involves placing adapters with each integrated system or application, which then uses the bus backbone for interaction with the integration server and the other interconnected systems [24]. Compared with hub-and-spoke model, the bus model solves the problem of scalability and offers better performance. However, the bus model is more complex and more difficult to administer as the entire system grows.

4.4. Web Services and EAI

Web Services technology is the compilation of technologies (XML, Simple Object Access Protocol (SOAP), Web Services Description Language (WSDL), and Universal Discovery, Description, and Inventory (UDDI)) that allow users to develop, catalog, and publish business services for delivery and use on the Web [10].

Web Services provide a distributed computing technology for revealing the business services of applications on the Internet or intranet using standard XML protocols and formats. The use of standard XML protocols makes Web Services platform, language, and vendor independent, and an ideal candidate for use in EAI solutions. Web Services eliminate the interoperability issues of existing solutions, such as CORBA and DCOM, by leveraging open Internet standards—Web Services Description Language (WSDL—to describe), Universal Description, Discovery, and Integration (UDDI—to advertise and syndicate), Simple Object Access Protocol (SOAP—to communicate) and Web Services Flow Language (WSFL—to define work flows, not yet a W3C standard) [9].

Butler [4] refers to web services as the catalyst in the EAI to provide a reliable, secure, and safe infrastructure. Although EAI adapters and connectors are still needed to integrate with legacy data sources, in the future, all applications with built-in Web Service/XML/SOAP capability will reduce the need for intrusive third-party adapters and connectors. Compared to the traditional EAI technology, web services are simpler, cheaper, based on open standards and more efficient, dynamic, and flexible [9]. In the near future, web services will remain EAI ‘Lite’ technology, enabling a rapid, non-complex integration solution. Web Services promotes and provides the plumbing for a peer-to-peer service network with little requirements for each to interoperate except that they must support XML. No complex servers are required for implementing Web Services. Adapters and connectors do not need to be purchased, and the business logic may remain resilient due to the flexible communication mechanism (XML). Web Services, for EAI ‘Lite’ situations, also results in shorter long-term maintenance costs due to their simplicity [10].

5. Conclusion

As EAI evolves and basic connectivity goals are achieved, firms will want to obtain more leverage from their EAI investments. Business issues will continue to become more complex and expectations from EAI will accordingly become more demanding. Some of these trends are already becoming obvious. The emergence of the extended enterprise and stronger involvement with suppliers and customers is one such trend. Business processes in such settings tend to run across different firms and thus dictate the need for greater flexibility from EAI. Davydov [6] talks about the evolution of EAI into IAC (inter-enterprise application cooperation) over the next few years.

Integration of multiple disparate sources: When dealing with needs of emerging enterprises, systems
developers face a bewildering conglomerate of programming languages, interfaces, APIs, and all kinds of other technologies that have to be integrated, in most cases, simultaneously. Moreover, the integration must occur at different levels involving multifaceted assembly from disparate sources. As much as EAI vendors are trying to convince us otherwise, existing products do not have the translation and other mediation-type capabilities needed to enable such forms of interoperation and interaction.

Plug-and-play interoperability: Organizations of the future, engaged in cross-enterprise operations, require two forms of interoperability—tightly integrating with partners while simultaneously retaining the flexibility to drop or add potential partners. Therefore, it must be possible to rapidly reconfigure application-to-application links to have such links functioning between partners only for the time required to complete specific arrangements. This will need significant additional effort.

Cross-enterprise collaboration: A key requirement for managing the work across the firm of the future is to have a dominant interface that controls processes and access to information across systems (providing global workflow integration capability). More specifically, EAI should enable an application model where cross-enterprise management of business processes is separate from the application interface.

Integration of decision-support and business intelligence systems: Finally, current EAI products are focused almost exclusively on issues of integrating transaction-oriented systems. But it is necessary to combine the decision-making capabilities of participating partners in order to monitor all the elements of the firm. In other words, it is not enough to integrate operational processes and data. EAI products must enable the integration of inter-enterprise decisions.

Given the bright outlook for the future, EAI is poised to grow and change to accommodate new business needs in the short- to moderate-term future. However, future EAI development must provide the right architecture. Firms must choose technologies on the basis of value, delivered individually and in unison with other technologies. Choosing “trendy” technologies just because they are fashionable will seriously hamper integration efforts. When designing new applications, firms must exercise care to ensure that they build in mechanisms to allow easy exchange of information. This will allow new applications to be easily integrated at the desired level.

The ultimate goal of EAI is to bind all the information systems in firms in such a way that any application can access any other application without delay. This “Zero Latency Enterprise” ideal is far from actualization and will most likely be the goal future EAI solutions will strive for.

References


David C. Yen is a professor of MIS and chair of the Department of Decision Sciences and Management Information Systems at Miami University. He received his PhD in MIS and Master of Sciences in Computer Science from the University of Nebraska. Professor Yen is active in research, he has published two books and over 70 articles which have appeared in Communication of the ACM, Information and Management, International Journal of Information Management, Journal of Computer Information Systems, Interface, Telematics and Informatics, Computer Standards and Interfaces, Internet Research among others. He was also one of the co-recipients for a number of grants such a Cleveland Foundation (1987–1988), GE Foundation (1989), and Microsoft Foundation (1996–1997).

T.M. Rajkumar is an associate professor of MIS at Miami University. He received his PhD in MIS for Texas Tech University. He is the co-author of a book on operating systems. His current research interests include outsourcing, troubled project management, E-commerce, and impression formation.

Naveen Erasala received an MBA degree with a concentration in Management Information Systems on May 2001 from Miami University at Oxford, OH and he is currently working in the information systems area. His research interests include data communication, Internet, and electronic commerce.