# *SECTION 6: Scope & Architecture*

# *11*

## *Intra Enterprise Integration: Methods and Direction*

Edward A. Stohr

Jeffrey V. Nickerson

### *Chapter Summary*

1. Corporations grow, and as they grow the need for integration across the enterprise increases. Integration can be accomplished through various means.
2. Consistent with the theme of this book, the need for alignment – integration is really a business need, and the technology mechanisms won't help without the proper organizational structures, goals, and incentives. Both organizational and technological integration is necessary.
3. Business processes are the vital link between the technical and organizational infrastructures of the organization. Processes are also the mechanism through which most inter-organizational interaction

takes place and are therefore the foundation for increasingly electronic forms of commerce.

4. Enterprise integration doesn't happen naturally. It needs to be planned. Yet the planning cannot be precise, as business processes and facilitating technologies will change, creating different needs and different potential solutions.

5. New mechanisms for integration can be utilized to create a flexible, loosely coupled framework, within which special integration needs can be quickly fulfilled. These new technical mechanisms depend on standards that work both inside and outside the enterprise.

## Introduction

This chapter focuses on enterprise integration at two levels – the systems level and the organizational level. Integration at the systems level requires common standards and data definitions, and some means of synchronizing the communication between different software applications. This is usually what is meant by the recently coined term *Enterprise Application Integration*. However, as pointed out by Markus and others [Markus], systems integration in the software system sense is not, in itself, sufficient to ensure organizational efficiency and effectiveness. Organizations consist of individuals, departments, divisions and functions, which must also be integrated for the organization to be successful. Both integration and coordination have been discussed in the management literature going back to the 1930s [Gulick], and their definitions have changed over time. Today, coordination is the more general term, referring to people-oriented as well as systems-oriented dependencies. Integration is most often used in discussing the linkages between software systems. This chapter uses the two terms interchangeably when talking about the organizational aspects of integration.

This chapter will not focus on a third level of integration, cross-enterprise integration and coordination, since this is the topic of the next chapter in this book. However, any discussion of enterprise integration must recognize that external needs, for example, customer requirements or supply chain efficiency, are increasingly important determinants of organizational effectiveness. Various points in the chapter discuss the boundaries between the systems owned by the organization and those owned by its trading partners and customers.

The chapter begins with a framework that encompasses both system and organizational integration. Section 3 begins the discussion of the systems level of integration. Section 4 explains the major mechanisms and architecture choices. Finally, section 5 provides a summary and some brief comments on future trends in enterprise integration.

## A Framework of Integration Requirements

A useful definition, which applies equally well to both systems integration and organizational coordination is made by Malone and Crowston [Malone, and Crowston] who define coordination as managing dependencies between activities. Figure 1 depicts the range of technical and organizational integration/coordination needs at a high level of granularity. A list of resource and activity dependencies is shown in the left column, common software and human coordination mechanisms in the middle column, and the supporting infrastructure elements in the right column. The resources, mechanisms and infrastructure elements are roughly arranged horizontally according to their sphere of influence. The boundary between technical and organizational integration mechanisms is shown by the dashed lines in the cells in the center of the table. Note that some of the mechanisms play a role at multiple levels in the Table.

It is argued that effective integration/coordination requires attention to elements at all levels on both the horizontal and vertical dimensions in the Table. The integrated architecture exists to support coordination in the use of the material, financial, and human resources of the firm.

<table>
<tr><th></th><th>Resource/<br>Integration Need</th><th>Examples of Integration<br>Mechanisms</th><th colspan="2">Enabling environment<br>/Infrastructure</th></tr>
<tr><td rowspan="2">Organizational Integration</td><td>Organizational Units<br>(Functions/Departments)</td><td>E-mail, collaborative software, lateral teams<br>-------------------------------------<br>Top Management Strategy, budgets, performance metrics</td><td colspan="2" rowspan="2">Organization<br>policies/<br>structure</td></tr>
<tr><td>Decision Makers</td><td>Email, collaborative software, knowledge management systems<br>-------------------------------------<br>Face-to-face meetings, job design, performance metrics</td></tr>
<tr><td rowspan="3">Systems Integration</td><td>Business Processes<br>(both internal & external to the firm)</td><td>Workflow, Collaborative Systems, SCM, CRM, Web Services<br>-------------------------------------<br>Process owners, teams, performance metrics, service level agreements</td><td>Standards</td><td rowspan="3">Systems<br>Architecture</td></tr>
<tr><td>Applications</td><td>Inter-process communication, RPC, Messaging, ERP, Web Services</td><td>Networks</td></tr>
<tr><td>Data</td><td>Data Dictionaries<br>Databases, XML</td><td>Platforms</td></tr>
</table>

**Figure 1: Framework for Business Integration**

The various levels in the figure at which integration is required are briefly described from bottom to top in the figure – from the most concrete to the most abstract. In so doing, it is pointed-out that integration is necessary not only *within* the various layers in Figure 1 but also *between* the layers. Also that, integration in one layer depends on integration at lower levels in the hierarchy. For example, integration at the application level requires a common understanding of the data that is being interchanged, which in turn, implies integration at the data level. The discussion focuses on the role played by the mechanisms in the center of the Table. Where appropriate, there is also reference to the role of the infrastructure elements - standards, architecture, networks, and organization structure - in helping to achieve integration both within and between layers.

**Data Integration**

*Definition:* The *goal* of data integration is to allow organizations to combine, aggregate, and report on data from different sources. Data integration involves both syntactic and semantic considerations. At the syntactic level, software programs must be able to handle data stored on the same or different devices on different media in different formats. At the semantic level, the meaning of all data items must be understood and the same data item must have the same definition across multiple applications both within and outside the firm. To make the integration process worth the effort, the data must be of high quality - timely, accurate and relevant.

*Objectives:* Data integration is desirable for several reasons:

- To provide timely and accurate information for analysis and decision making by both management (e.g., data warehousing applications) and customers (e.g., product catalogs.)
- To provide a single authoritative source of information for use in performance measurement and the audit process
- To facilitate interaction between software programs in order to achieve program and business process integration.

*Mechanisms*: Simple data integration mechanisms are found in most programming languages, ranging from Cobol to Java. File member libraries in COBOL allow programmers to share data definitions. Class definitions in Java fulfill the same function. At a higher level, data dictionaries also provide a systematic way of integrating information with an emphasis on semantics. However, none of these mechanisms scale well when programs are written in different languages and query many different kinds of databases. As discussed later, the use of Extensible Mark-up Language (XML) Schemas is the currently favored solution for data integration. XML combines data and the description of the data in one place, which greatly simplifies integration [Ibbotson]. In cross-enterprise integration, EDI has served a data integration role, providing a standard format for the exchange of common documents. Even here, the trend is toward XML.

**Data Level - Application Level Integration**

*Definition:* Integration across the data and application layers allows programs to access and understand data stored in heterogeneous files and different databases.

*Mechanisms:* Standards such as ODBC and JDBC allow databases from different vendors to be accessed by a single application.

**Application Integration**

*Definition:* When sets of applications are integrated, each can call on the functions of the others. *Objectives:* The objectives of application integration are:

- To break down complex processing needs into discrete steps performed by relatively independent programs (e.g., a program to perform a file update might be followed by an independent program that produces management reports.)
- To minimize the number of manual steps. In particular, to ensure that data is entered only once.
- To support interactive processing (e.g., provide one-stop shopping for customers by linking diverse applications in a common front end.)

- To facilitate data and information integration

*Mechanisms:* As discussed in more detail later, application integration is now supported by a wide range of middleware products. Initially, such integration was supported by RPC calls, and then by transaction managers, and now by application servers – each of these advances has incorporated the techniques of the previous mechanisms. IBM, BEA, Microsoft, and Tibco all provide many different variants of these mechanisms. However, these systems do not interface with humans directly and so only indirectly support business processes (i.e. through the applications that they integrate.)

At a higher level, ERP systems achieve application integration because ERP vendor-developed applications that perform common business functions are united through a common database (also providing data integration – one of the selling points of ERP systems.) The integration of the functional applications implies also that integration at the next layer, business processes, is achieved. However a major issue with ERP systems is that they fail to bridge the gap between the application and process layers in a flexible fashion. This gives rise to the common complaint that it is easier to fit the organization to the ERP system than it is to adapt the ERP system to serve organizational requirements [Esteves and Pastor].

**Application Level - Business Process Integration**

*Definition:* Integration between the business process and application level occurs whenever an application is used in a business process. Almost all applications of computing in organizations fit this definition in a broad sense including traditional batch processing using legacy systems. The tightest integration between the layers occurs when a process calls an application automatically or when ERP, legacy, or client-server applications that are to be used by a human processor are automatically brought to the user's screen to be used as aids in decision making.

*Mechanisms:* Integration of applications and business processes is achieved to some extent by ERP systems as mentioned above, with the caveat that they are often perceived as costly and inflexible. Alternatively, Workflow Management Systems (WFMS) are a form of middleware that provide integration between the application and process layers. Workflow is oriented toward processes, and hence it is discussed more fully in the next section.

**Business Process Integration/Coordination**

*Definitions:* A business process consists of related sets of activities that are performed by human and software actors according to business rules that may be more or less stringently applied. The connection between the software and human agents that perform a process is well integrated or coordinated when the process is efficient, accurate, and appropriate to the task at hand from a mechanistic, human, and organizational viewpoint. Moreover, the process goals must align with those of the organization as a whole. This is called *within-process integration*.

Processes often cross departmental boundaries and can therefore provide a coordination mechanism in the organization layer as discussed below. This is called *cross-functional coordination*. Processes must also be integrated/coordinated with each other. For example, the order entry process needs to be integrated with accounts receivable and with the back end processes that produce the product or service desired by the customer. This form of integration is called *internal process-to-process integration*.

Finally, e-business demands that the internal processes of the firm be integrated with those of its trading partners and customers. *External process integration* means that the organization is able to connect its internal processes seamlessly with those of its suppliers, intermediaries, and customers. This is the basis of e-commerce. Automated and partially automated process integration is distinguished. Fully automated internal-external process integration has already been achieved to a remarkable extent for the purposes of security, when a firm links in real time to certificate authorities and credit-granting agencies, and in advertising, when an ad server company such as DoubleClick pushes advertisements to consumer Web pages serving both the advertiser and the publishers in its network. As another example, there are two large, overlapping efforts in the financial services industry that seek to automate the entire range of trading-related business processes. One effort is called straight-through processing, emphasizing the goal of full automation. The second is called T+1, emphasizing the reduction in overall trade processing time. Both require that the internal processes of the financial services organization be closely synchronized with those of its customers and business partners.

At an even higher level of external process integration, processes must be coordinated between all the firms in the value chain to achieve improved performance and service. This form of external process integration, which is called *value chain coordination*, is the focus of modern supply chain management.

*Objectives:* Summarizing, the reasons why business process integration is desirable include:

- Coordination between individual human and software actors as they perform the work of the organization (integration within the process)
- Coordination between the efforts of organizational units such as departments and divisions that have different roles to play in the execution of shared processes (cross-functional coordination and internal process-to-process coordination)
- Connecting an organization with its suppliers and customers (external process integration.)
- Optimizing the joint performance of all partners in the value added chain for a good or service (value chain coordination)
- Increasing the efficiency of the organization and its ability to compete in terms of its agility, cost, and service capabilities

*Mechanisms:* Workflow management systems are explicitly designed to support business process automation by moving work between human and software actors according to rules [WFMC, The Workflow Reference Model]. These systems provide visual interfaces for process design, manage process instances, and interface readily with the organization's legacy, client-server, and ERP applications. They also contain simple organizational models (e.g. who reports to whom, who has approval authority for what, who can perform what roles, and who has access rights to what data and what applications.) And they perform a resource coordination function by managing, work assignments and load balancing between actors. In this sense, Workflow contributes to integration at all the levels in Table 1. Historically, workflow has been used mainly for process, cross-functional, and internal coordination. This is changing; the major trend in the workflow area now involves external and to some extent, value chain integration.

**Business Process – Decision Making Integration**

*Definition:* Integration between the decision making and business process layers occurs whenever a human operator (or software agent) makes a decision that changes the flow of work through a process. For example, in a helpdesk process, a customer service representative may decide to escalate a problem case by referring it to an expert in another department within the organization.

*Mechanisms:* There are many ways in which this form of integration may be facilitated. One example is to use a workflow system that presents a user-friendly interface and automatically provides the decision maker with access to the tools needed to perform analyses and spot errors. *Informating* a process by providing decision-relevant information on an as-needed basis and feedback to the operators concerning their performance is another example. Embedding expert systems at decision points in the process is another. Finally, knowledge can be imbedded into the two- part process in which process data is collected and analyzed for patterns that are later used to modify the process or inform operators, a process best described as double-loop learning [Argyris]. This level is really the level of the user interface, where the success of a company's systems will often hinge.

**Integration between Decision Makers**

*Definition:* Integration/coordination of decision makers occurs when decision makers are able to share their ideas and knowledge freely and when they coordinate their actions to the benefit of the enterprise. To achieve this integration, strategic decisions often have to be made by upper levels of management in the cooperating organizations. While the classical models of organizational hierarchies imply that leaders are isolated at the top, more current analyses of organizations show that decision makers are in constant contact with each other, and that their interactions are crucial to the success of a company. While the concept of coordination in organizations originally stressed interaction between people, the current ways of thinking about it stress interaction between activities [Malone and Crowston]. This more recent way of thinking has led to much significant work. However, it is at a cost – for the interactions between executives in a company are not simply around tasks. It is around friendships and families as well. The psychological side of integration in an organization should not be underestimated – large and costly projects can fail due to interpersonal friction. It is for this reason that decision-makers choose less structured forms of communication – meetings, dinners, and email – in dealing with their peers. And it is for this reason that companies encourage social interaction among their executives.

*Mechanisms:* The mechanisms of interaction include the simplest ones – face-to-face interactions and the telephone. But other, more asynchronous, methods are also important such as voice mail and email. Computer-based systems have generally provided lean rather than rich information to decision makers. Multimedia approaches can increase the richness and accessibility of information. To facilitate communication that is closer to human communication, richer forms of communication such as multimedia and interactive media are sometimes helpful. Collaborative systems are specifically designed to help human decision makers coordinate their activities, and build shared virtual spaces that allow teams to asynchronously communicate and share information and ideas.

**Decision-Maker - Organizational Integration**

*Definition:* Decision makers are integrated into their organizational units to the extent that their actions are coordinated with each other to achieve departmental or divisional objectives.

*Mechanisms*. Classical organizational theory was very much concerned with this problem [Lawrence and Lorsch]. Ideas such as span of control, delegation of authority, communication of goals from higher levels in the hierarchy, and budgeting were advocated. These mechanisms are still relevant today. The integration of decision makers into departments is also controlled through incentives. Decision makers are motivated very directly to meet the goals of the team they belong to. And they can be motivated to also contribute to the goals of other teams. There is a large literature on incentive schemes in economics [Jensen and Meckling], in management [Eisenhardt] and, currently, the Balanced Scorecard approach of Kaplan and Norton [Kaplan and Norton]. An important development since the advent of tightly coupled value chains is the need to integrate decision making from one organization into that of another. For example P&G manages its customers inventory [McKenney and Clark].

**Organizational: Functional/Departmental Integration**

*Definition:* Through a process of differentiation, the organization is divided into multiple divisions or departments [Lawrence and Lorsch]. Differentiation occurs because each unit of an organization needs to focus on a different set of conditions outside of the firm, and therefore needs to specialize. This specialization leads to differences in attitude of managers, along the four dimensions of goals, time orientation, interpersonal orientation, and structural formality [Eisenhardt]. Which increases the challenge of integration. Yet Lawrence and Lorsch point out that the best performing organizations are both highly differentiated and highly integrated [Lawrence and Lorsch].

Organizational integration requires first that communication is established so that cross-departmental processes are executed smoothly and that departments are informed of the activities of other departments, in connection with the resources they compete for and the processes that they share. Second, departmental integration requires that the goals of the various departments be integrated.

*Objectives:* Organizational integration is desired for the following reasons:

- Departments often depend on each other for inputs (sequential or reciprocal dependence [Thompson])
- Departments often need to cooperate to execute distinct parts of a process
- Integration can mean more efficient sharing of resources and the development of organizational standards
- Functional integration helps support process integration because the functional or departmental managers are better able to coordinate their decisions with respect to process execution.

*Mechanisms:* In the classical view of organizations, coordination and integration are achieved by the controlled delegation of authority and hierarchical command and control structures [Fayol] [Urwick]. However, another reality of organizations is that much of the work is performed through processes in which work is handed off between people that often belong to different departments or divisions. Advocates of the reengineering movement of the 90's advanced the view that the process or horizontal view of organizations was required to overcome the inefficiencies of the vertical organization with its stove pipe mentality and often problematic hand-offs of work between departments. To enable horizontal coordination, reengineering proponents and organizational theorists [Galbraith] advocate the use of process owners, or, at least, process teams with members from each of the functional departments involved in the process. Taking this further, reengineering and other organizational theorists have proposed process-oriented organizational structures in which the functional disciplines are subordinated to the process view [Galbraith]. Adding process owners and cross-functional teams to the organization helps horizontal coordination but requires some kind of matrix management since the team members must balance the demands of their functional (vertical) as well as process (horizontal) coordination. What seems most important is that there are boundary spanners who, like diplomats, speak the specialized languages of all the departments to be integrated [Lawrence].

A major issue in achieving departmental coordination is that vertical chain-of-command; authority must be reconciled with the horizontal process needs of the organization. To address this problem, Rummler and Brache [Rummler and Brache] recommend that the share of resources and the goals of each department be derived from the department’s contributions to the processes that cross its boundaries. Similarly, the goals of each job should be derived from both the process and functional requirements. This approach is markedly different from the usual practice in which functional goals and performance measures are set by the functions in conjunction with top level management without much consideration of horizontal coordination needs and job performance measures are determined by the functions with at most a parochial view of process needs.

Increasingly, organizations need to integrate not only their own internal departments, but also with other organizations. Supply Chain Management (SCM) software such as that provided by I2 seeks to optimize the logistics within the value added chain. CRM software such as that supplied by Siebel, seeks to integrate the firm with its customers. Finally, electronic exchanges such as Covisint, operated by the major auto suppliers, integrate the buy and sell transactions of many suppliers and buyers.

**Summary**

This section constructed a framework of integration that includes technical as well as organizational factors. The literature on enterprise integration focuses on the data and application layers in the framework together with the standards, architecture, and networking infrastructure alternatives that can integrate these technical elements. On the other hand, the organizational design literature focuses primarily on the decision making and departmental integration layers along with the supporting organizational structure, policies, and strategy. Only recently have reengineering and organizational theorists focused on the middle layer, business processes. Business processes are the vital link between the technical and organizational infrastructures of the organization. Processes are also the mechanism through which most inter-organizational interaction takes place and are therefore the foundation for increasingly electronic forms of commerce.

Proper integration of the elements in Figure 1 should help achieve an agile and efficient organization. However, this is only half the story. A man-machine system is described as if it existed without any purpose and without the resources that are needed to achieve this purpose. To complete the story, assert first, that the organization must have a winning strategy, and second, that in pursuit of this strategy, the integrated technical and organizational components that have been discussed must coordinate and integrate the organization's material, financial, and human resources in pursuit of the organization’s goals. The vast literature of business strategy, management science, finance, and organizational behavior all can be brought to bear on the organizational aspects of enterprise integration. This section of the paper provided only a brief outline of the requirements for an integrated technical and organizational system through which the organization can achieve its ends.

## *Technology Mechanisms of Enterprise Integration*

### Introduction

It has been established that enterprise integration needs to occur at many technical and organizational levels. Some of the methods that can be used at each level have been introduced. This section takes a closer look at the facilitating technologies that can be used to attack the overall problem.

An organization, convinced it needs an architecture that supports integration across the enterprise, can easily state its ideal. The integrated enterprise of the future will tie together all existing applications, so that data from one application can be easily used in another. Business Processes, which may stretch across different applications, will be combinable, so the output from one will be input to another. The integration mechanisms will not create performance bottlenecks. And the integrated enterprise will be easy to change.

This ideal hasn't changed throughout the history of information systems. It is proving hard to achieve, but progress is being made. Four current trends in the quest for the ideal integrated enterprise system are looked at first.

*Overall frameworks are preferred to point solutions*

Enterprises are seeking to solve the integration problem overall, instead of one connection at a time. This is leading enterprises into overall system-designing activities, looking across all major application system. The hope is that, with an overall platform in place, individual integration projects can be accomplished more quickly.

*Loose Coupling is more popular than tight coupling*

Mechanisms for integration are trending toward the loosely coupled. In a tightly coupled environment, a change to one system needs to be coordinated with changes to all connecting systems. In a loosely coupled environment, a change to one system can often be made independently of the connecting systems.

*XML is driving out customized file formats*

The use of the data description language XML is becoming a de facto standard within the enterprise. This is important because XML can describe all of the data flowing between systems in a uniform way; greatly reducing the myriad file formats enterprises support.

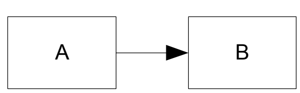
*Industry-wide efforts are driving internal efforts*

The outside is driving the inside – industry standardization efforts are being adopted inside the enterprise. So, for example, a set of data definitions used by financial services companies to communicate with each other can also be used internally to integrate enterprise systems. Once done, internal and external communication can use the same standard set of messages. Both the RosettaNet standards, focused on the manufacturing industry, and the new ISO standard focused on the financial industry are examples of industry models that are finding their way into the enterprise[RosettaNet, ISO, ISO-15022-Securities - Scheme for Messages].

## General Approaches

This section looks first at the three approaches to the problem that have evolved over time. It then discusses the actual mechanisms in use today, both the traditional and the trendy. Finally, it discusses the more systematic approaches to integration that have been proposed in both industry and academia.

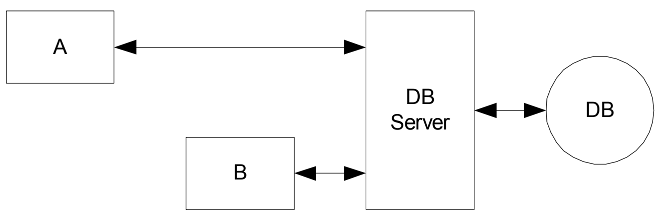
### Approach One: Sequential Integration




The first instances of systems integration involved the movement of a deck of output cards to the input tray of the next program. This kind of sequential integration between programs became more sophisticated with the advent of IBM's Job Control

Language. Using this language, a whole series of computer programs could be linked together. And batch jobs based on this language, involving complex runs of many applications, still do the majority of work in large-scale insurance systems. Sequential integration won't apply to certain problems. An airline reservation system calls for synchronization of many programs running at the same time, and cannot be serialized into a set of batch jobs. More modern approaches to sequential integration, such as the shell system of UNIX, provide a cleaner interface, but work the same way, and share the same limitations.

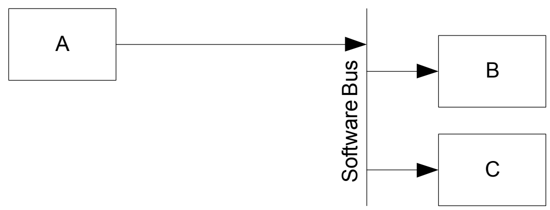
**Approach 2: Databases**




Two distinct applications can share information by alternately reading and writing to the same database server. The applications don't need to know about each other – they just need to share a common understanding of the data and what the data means.

This approach can handle problems sequential integration can't – any transactional system, such as a reservation system, can be implemented with database technology. In an airline system, many application programs will be interested in whether a particular seat is full – all these programs can be coordinated through access to a common database that updates the crucial link between a passenger and a seat.

**Approach 3: Messages**




Modern systems are defined in terms of their messages – every part of a system is thought of as a separate entity that can either originate or receive a message [Rumbaugh and Booch]. The appeal of this approach is that it works on any scale – from the signals being sent between chip components to the files being interchanged between large computer systems. Sequential integration as well as database integration are described as specialized variants of message passing systems. More importantly, message-oriented middleware can be used as a way of building loosely coupled systems, in which each component does not need to understand everything that the other components are doing.

***The de facto mechanisms of integration***

With the approaches above in mind, the commonly used technical mechanisms of integration across an enterprise can be described.

***Integration using files***

Most companies don't integrate two technology systems until they absolutely need to. For example, if a company creates a new telemarketing sales force, a new order management system associated with them might need to communicate with an existing sales commission system. The simplest way to integrate is to establish a point-to-point interface. The order management system can generate a file every night that is read the next morning by the commission system.

The disadvantage of this mechanism becomes apparent as the number of systems in the enterprise grows. If 15 systems need connections to each other, over 100 files need to be generated. An IT organization can be overwhelmed by the need to update and maintain file interfaces. The incremental cost of each new interface is low, but the combined ongoing cost to the organization is high. And by the time the organization realizes this, the replacement cost of the interfaces may be substantial.

***Remote Procedure Calls***

RPCs are the underlying mechanism of client-server computing. The client calls the server, requesting a response, and waits for the result. The server responds to multiple clients, and provides the responses as fast as it can – sometimes in turn requesting a response from another server, and relaying the answers. RPCs are behind many different integration frameworks, most notably CORBA and COM [Orfali, Harkey and Edwards]. RPCs require tight coupling. The client process needs to know exactly where the server is, and what the server expects. Neither the client nor server can change independently of the other.

***ERP integration***

As discussed earlier, a company can integrate much of their operation by replacing financial, HR, and manufacturing systems with an ERP system provided by one of several large software vendors. All applications will share the same data model, the same databases, and the same interfaces. Reporting across functions or divisions becomes possible. This approach has worked for many companies. It has also failed for others. But ERP is more an approach than a technology, and there are signs that the next generation of ERP systems will take advantage of the new technology mechanisms such as standard application servers. These new ERP systems will be more loosely coupled, and therefore, more amenable to change, and more likely to succeed.

***Integration using consolidated databases***

In its simplest form, a database is shared across a set of applications. The database serves as a shared memory for the entire universe of applications accessing it. While attractive from the logical perspective, many organizations run into problems when they physically implement such systems. For it is easy to optimize a database for a single application, but difficult to optimize for several. And software designers do not like to forfeit performance control to another part of the organization. A single complex transaction from one application can lock up several tables and force a different application to back off, possibly for minutes.

To allow local control of data but still provide integration of information, sophisticated IT organizations use local databases for each application, and then use some other asynchronous technology to update a consolidated database that can be queried by applications that require more global sets of information. If the applications are only querying the database for decision support purposes, rather than operation purposes, then the consolidated stores are called data warehouses or datamarts.

The transaction to the local database is not slowed down. Yet the information is integrated.

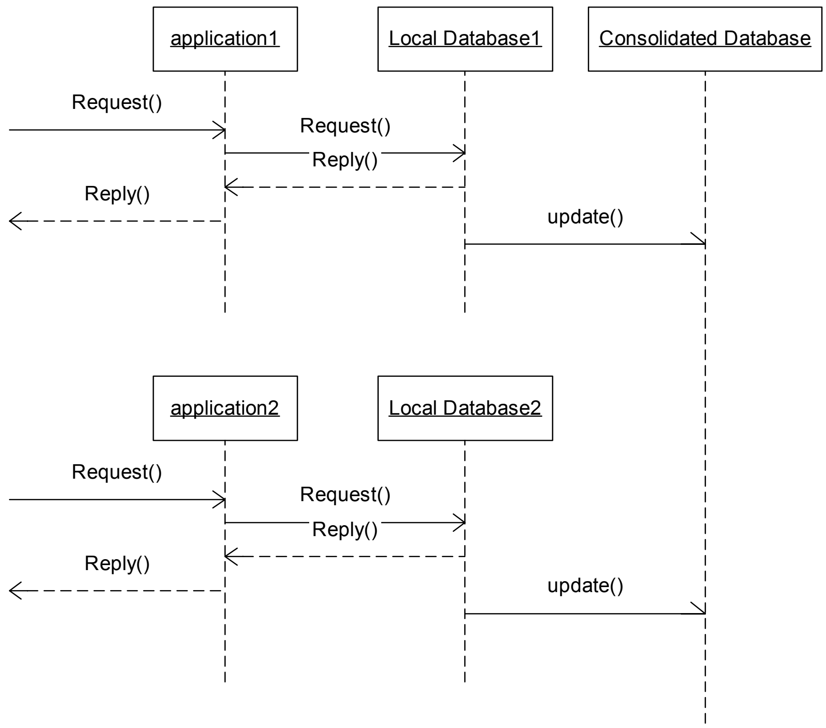



**Figure 2: Database integration**

For this to work, a universal data model needs to be modeled, agreed upon and enforced. In the past, such efforts have proven expensive and time-consuming. It is a difficult exercise – and once accomplished, it takes a great deal of attention to maintain the model in the face of changing business conditions. The current trend is toward pooling the effort of creating a model at the industry level. The industry model can also be used internally.

### *The newer mechanisms*

***Publish/Subscribe***

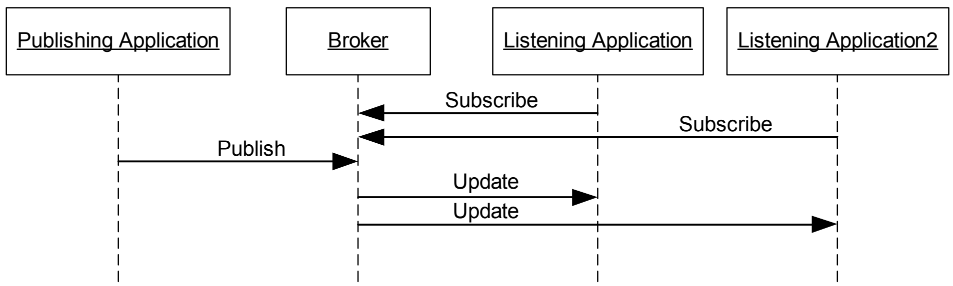



**Figure 3. An application using a publish/subscribe mechanism publishes an event once. Listeners in other applications are always waiting for the event so that they can pull it into the application. In implementations, a broker often operates between the publisher and the listener, handling subscription changes and broadcasting resends if a listener misses an event.**

The problem with most integration mechanisms is that they force one-to-one communications between senders and receivers – each needs to know about the other. The publish/subscribe mechanism decouples the communication, so that the publisher may not even know who is receiving – and is not impacted by an increase in the number of receivers. And the receiver doesn't need to know the source of the information sought – the receiver simply subscribes to events of interest.

This scheme is attractive as a mechanism for enterprise integration. Each major application system publishes events out on a bus. A new account opening, a sale, a supplier shipment notice – all of these would be sent. Then the systems that need particular types of information subscribe. For example, a risk management system might want to see all major sales transactions, as well as stock market quotes. The risk management system could be built without changing the sales or quote systems, which may not even know the risk management system exists [Eugster, Guerraoui and Damm], [Eugster, Guerraoui and Sventek], [Eugster], [Oki]

The Publish/Subscribe mechanism can solve many enterprise integration systems. It is also a complex technology, and it takes skill to integrate it into existing applications. There are many commercial vendors who offer this capability, and there is an emerging standard, Java Messaging Services, that will eventually create an environment for interoperability among most vendor solutions.

***Enterprise Application Integration***

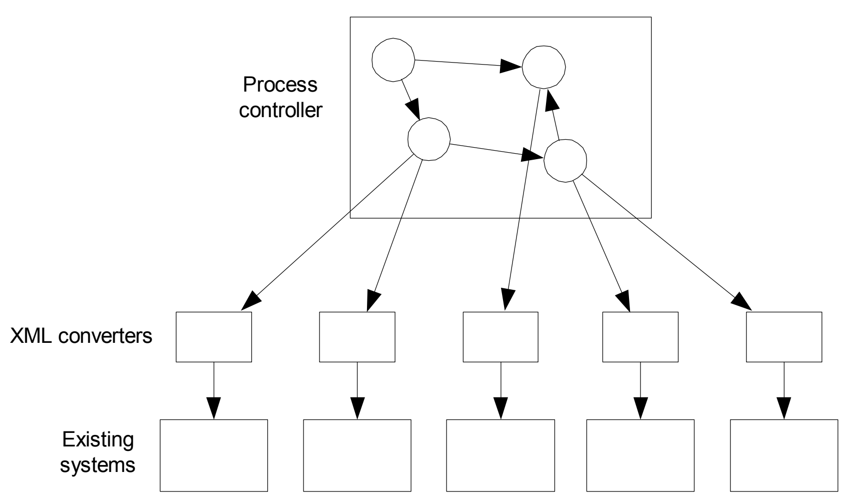



**Figure 4: Enterprise Application Integration systems architecture**

A new infrastructure referred to as *enterprise application integration* ties together processes that reach across large numbers of systems. These systems usually ship with universal data models, mapping all common packaged systems database fields into one format.

Unlike the other mechanisms discussed, EAI systems feature a process, rather than a message, broker. The broker pulls process information into a central control. By working at a process level, it becomes possible to link together disparate systems, specifying the data to be moved, and the compensating transactions to be fired in the event of a failure. Once created, using either a diagram format or a high-level language, the broker will handle the sequential integration and data integration between legacy systems [Johannesson and Perjons]. There are many different ways this is accomplished, but a common method is to send XML messages to converters that will invoke the legacy systems in their preferred data format.

***Web Services***

A new set of standards seeks to create a way for programs to automatically discover other services on the Internet, and then use those services. While the model is oriented toward integration outside the enterprise, it will end up being used inside also, as it provides many of the integration mechanisms that are necessary in a large company.

The standard has three parts. The first is the Simple Object Access Protocol [Mitra], which defines an RPC call. The second is the Web Service Description Language, which defines how a service works – in a way that a computer can understand. The third is the Universal Description, Discovery, and Integration standard [McKee, Ehnebuske and D.R.], which allows a program to automatically discover a service application.

The standards allow for programs to get the services they need – without being tied to a particular server. In the ideal scenario, a program is written that depends on a particular service. At the time the program is run, it queries a directory to find out who supplies the service. The program reads in detail the format of the messages needed, and converts its data into what is expected by the server. The server is called, and an answer is returned.

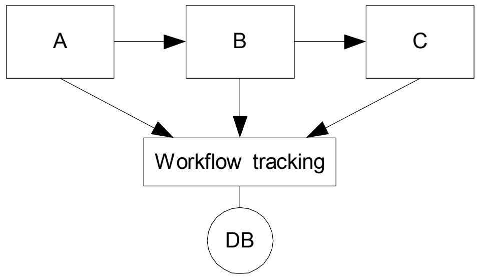
***Agent-based methods***

The more web services there are, the more likely it is that agents will flourish. Agents allow us to delegate a difficult transactional task. The agent can take an instruction from a program, and autonomously locate services, run them, and return results.

Once agents can handle simple transactions, they will be invoked to handle complex negotiating strategies involving price. In an integration environment, an agent can be used to tie together different systems through the accomplishment of a pre-set goal. Agents are tireless and patient, two necessary prerequisites to true integration of activities.

The design of automated agents is a rapidly expanding area of research, not yet commercialized [Riha], [Ottaway and Willis], [Kwon and Lee.], [Klein and Dellarocas], [Cost], [Huang, Houng and Mak], [Shrivastava], [Yan, Maamar, and Shen].

***Workflow***




The preceding approaches to integration focus on the systems aspects - the integration of data and applications. With the exception of ERP systems, they do not involve human interaction. In contrast to the preceding approaches, workflow management systems have the potential to integrate both the systems and organizational aspects of organizations, as discussed earlier.

### *Systematic Approaches*

In contrast to the methods above, academic researchers have suggested systematic technical approaches to performing enterprise integration. From the standpoint of a software architect, integration has little to do with the specifics of a business. Instead, it is an issue of how to link things. First, Mary Shaw at Carnegie Mellon University pointed out that connectors between systems were often an afterthought – but to do integration properly give the connectors first-class status [Shaw]. Then Chris Dellarocas at MIT pointed out that perhaps there is a need to give first-class status to component interdependencies, as interdependency is a higher-level concept than connection [Dellarocas] .

Both these observations reflect the technical perspective that it is too difficult, and probably fruitless, to try to anticipate all the potential interconnections between business functions – that instead, it makes more sense to accept that functioning software components will be combined in ways that cannot be predicted. The problem becomes one of making assumptions about a component as explicit as possible so that compatible functions can be combined without further investigation and testing.

Earlier in the chapter the coordination science work done at MIT was discussed.[Malone and Crowston] Their work seeks to systematically map business problems to a set of coordination mechanisms.

As this applies to software systems, resources are broadly described to include CPU time, data, operating systems resources, and other hardware. Activities are either resource producers or resource consumers. Figure 5 shows all six permutations of producers and consumer interaction. The top row shows three common forms of coordination – in *sharing*, two processes might need to coordinate use of a finite resource such as a printer. In *flow*, one process waits for another to produce a result. In *fit*, two processes must combine their outputs into a common whole, as when software developers integrate their work into a final application. In the bottom row, the left diagram shows the circumstance in which a task reads from multiple resources. The right diagram shows a task producing multiple resources. And in the center, a task reads from one resource and produces another. When this diagram is joined with the one above it, the combination suggests sequential flow across an unlimited number of tasks.

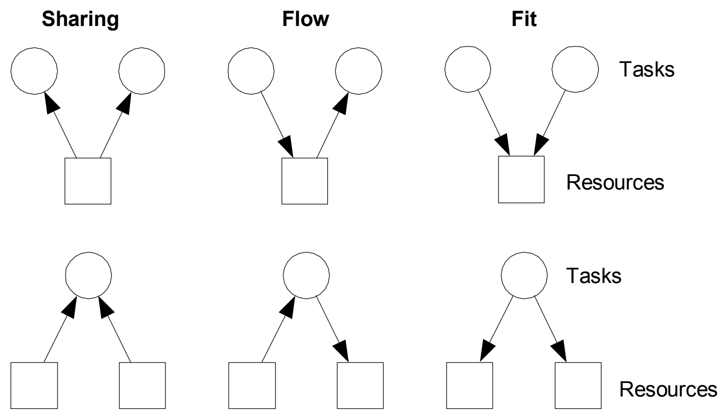



**Figure 5: Combined from Crowston [39] and Malone [40], showing all the permutations of tasks and resources. The middle top diagram shows a task producing a resource which is consumed by another task.**

Malone and others in the MIT Center for Coordination Studies have codified dependencies and their associated coordination techniques in a process handbook [Malone]. It is a bottom-up approach – by cataloging a set of business processes and their associated coordination mechanisms, the hope is that the practices of an organization can continually be refined. In contrast, there is a top-down effort to create an international integration standard called the Generalized Enterprise Reference Architecture and Methodology (GERAM)[Chalmeta, Campos, and Grangel].

It is far from clear that any single generalized architecture can work. It is also not clear if a process handbook can ever be complete. To date, neither the academic theory nor the industry architectures has had as much direct impact on enterprises as vendor products and internal IT teams have had. Yet ideas on architecture, no matter what the source, have a way of appearing in software products. It is expected that integration frameworks may grow to encompass many different technical and organizational mechanisms, applied according to the nature of the specific integration problem.

### *The Process of Integrating Across the Enterprise*

Enterprise integration doesn't happen naturally. It needs to be planned. Yet the planning cannot be precise, as business processes and facilitating technologies will change, creating different needs and different potential solutions.

The reference architecture efforts mentioned define methodologies for building integrated architectures. There are few surprises in the sequences advocated – all the methods consist of a sequence of discovery, analysis, design, implementation, and maintenance, as in the development of an individual application.

The metaphor *enterprise integration in city planning is suggested*. In city planning, it is important to develop the street plan, but not to focus on the individual buildings. The size of the block is important; so is the diameter of the pipes supplying water. But a city planner who tries to specify the size of each building's windows will not be serving the community. City planners try to anticipate, but not over fit. They know there are many alternative futures for a city.

An enterprise integration plan will have a similar structure, putting in place a framework that encourages integration by defining the standard mechanisms for connection between processes and applications. Once the standards are set, and the processes are defined, the planning activity has fulfilled its function. Individual systems can be built imaginatively within the defined framework.

City planning, and enterprise integration, is easier if nothing has been built yet. But in the case of enterprise integration, there is almost always a large and complex infrastructure in place. Those who have engaged in retrofitting an existing environment have often found there is resistance to the integration efforts. As discussed, the resistance is due to human issues such as compensation and recognition.

Enterprise integration will not work unless all layers of Figure 1, both organization and technical, are considered. On the organizational side, the problem of integration is exacerbated by the limits of human cognition, by behavioral issues, by the difficulty of aligning the goals of individuals with organizational units, and by the relentless need for faster change. There is a long history of research in organization design and many approaches have been advocated. But distributed, networked organizations act and interact in a world that moves much faster than before. Exploding information availability presents new challenges to managers as well as to systems architects.

Facilitating technologies can help both groups. New mechanisms for integration can be utilized to create a flexible, loosely coupled framework, within which special integration needs can be quickly fulfilled. These new technical mechanisms depend on standards that work both inside and outside the enterprise. Which presents interesting opportunities for those engaged in cross-enterprise integration, the subject of the next chapter.

## Notes


Argyris, C., *Good Communication That Blocks Learning*, in *Harvard Business Review*. 1994. p. 77-85.

Chalmeta, R., C. Campos, and R. Grangel, *References architectures for enterprise integration.* Journal of Systems and Software, 2001. **57**(3): p. 175-191.

Cost, R.S., et al. *An agent-based infrastructure for enterprise integration*. in *First International Symposium on Agent Systems and Applications/Third International Symposium on Mobile Agents*. 1999. Los Alamitos, CA, USA: IEEE Comput. Soc.

Crowston, K., A Taxonomy of Organizational Dependencies and Coordination Mechanisms, http://ccs.mit.edu/papers/CCSWP174.html

Dellarocas, C., *A Coordination Perspective on Software Architecture: Towards a Design Handbook for Integrating Software Components*. 1996, MIT.

Eisenhardt, K.M., *Agency Theory: An Assessment and Review Academy of Management Review.* Academy of Management Review, 1989. **14**(1): p. 57-74.

Esteves, J. and J. Pastor, Enterprise Resource Planning Systems Research: An Annotated Bibliography. Communications of the Association for Information Systems

Eugster, P.T., R. Guerraoui, and C. Damm. *On Objects and Events*. in *OOPSLA*. 2001.

Eugster, P.T., R. Guerraoui, and J. Sventek. *Distributed Asynchronous Collections: Abstractions for Publish/Subscribe Interaction*. in *ECOOP*. 2000.

Eugster, P., et al., *The Many Faces of Publish/Subscribe*. 2001, EPFL.

Fayol, H., *Industrial and General Administration*. 1925, Paris: Dunod.

Galbraith, J.R., *Designing Organizations: An Executive Briefing on Strategy, Structure, and Process*. 1995, San Francisco: Jossey-Bass.

Gulick, L., *Notes on the Theory of Organization*, in *Papers on the Science of Administration*, L. Gulick and L. Urwick, Editors. 1937, Institute of Public Administration: New York. p. 3-13.

Huang, G.Q., J. Houng, and K.L. Mak, *Agent-based workflow management in collaborative product development on the Internet.* Computer Aided Design, 2000. **32**(2): p. 133-44.

Ibbotson, J., ed. *XML Protocol Usage Scenarios*. 2001, W3C.

Jensen, M.C. and W.H. Meckling, *Theory of the Firm: Managerial Behavior, Agency Costs, and Ownership Structure.* J. Financial Economics, 1973. **3**(3): p. 305-360.

Johannesson, P. and E. Perjons, *Design principles for process modelling in enterprise application integration.* Information Systems, 2001. **26**(3): p. 165-184.ISO, ISO-15022 - Securities - Scheme for Messages, http://www.iso.org/

Kaplan, R.S. and D.P. Norton, *The Balanced Scorecard: Measures that Drive Performance.* Harvard Business Review, 1992.

Klein, M. and C. Dellarocas. *Exception handling in agent systems*. in *Proceedings of theInterantional Conference on Autonomous Agents*. 1999. New York, NY: ACM.

Kwon, O.B. and J.J. Lee, *A multi-agent intelligent system for efficient ERP maintenance.* Expert Systems with Applications, 2001. **21**(4): p. 191-202.

Lawrence, P.R. and J.W. Lorsch, *Organization and environment; managing differentiation and integration*. 1967, Boston,: Division of Research Graduate School of Business Administration Harvard University.

Malone, T.W. and K. Crowston, *Interdisciplinary study of coordination.* ACM Computing Surveys, 1994. **26**(1): p. 87-119.

Malone, T.W., et al., *Tools for inventing organizations: Toward a handbook of organizational processes.* Management Science, 1999. **45**(3): p. 425-443.

Markus, M.L., *Paradigm Shifts - E-Business and Business/Systems Integration.* Communications of AIS, 2000. **4**(10).

McKenney, J.L. and T.H. Clark, *Procter & Gamble: Improving Consumer Value Through Process Redesign.* Harvard Business School Case 195126, 1995.

**McKee, B., D. Ehnebuske, and D.R. ed., UDDI Version 2.0 API Specification, http://www.uddi.org/pubs/ProgrammersAPI-V2.00-Open-20010608.pdf**

Mitra, N., ed. *SOAP Version 1.2*. 2001, W3C.

Oki, B., et al. *The information bus-- an architecture for exensible distributed systems.* in *Proceedings of the Fourteenth ACM Symposium on Operating Systems Principles*. 1993: ACM.

Orfali, R., D. Harkey, and J. Edwards, *The essential distributed objects survival guide*. 1996, New York: J. Wiley.

Ottaway, T. and G. Willis. *An adaptive workflow management system utilizing agent technology*. in *Proceedings of National Annual Meeting to the Decision Sciences*. 1997. Atlanta, GA, USA: Decision Sci. Inst.

Riha, A., et al. *Adoption of an agent-based production planning technology in the manufacturing industry*. in *12th International Workshop on Database and Expert Systems Applications*. 2001. Los Alamitos, CA, USA: IEEE Comput. Soc.

RosettaNet, http://www.rosettanet.org/

Rummler, G.A. and A.P. Brache, *Improving performance : how to manage the white space on the organization chart*. Jossey-Bass management series. 1990, San Francisco: Jossey-Bass.

Rumbaugh, J., I. Jacobson, and G. Booch, *The unified modeling language reference manual*. The Addison-Wesley object technology series. 1999, Reading, Mass.: Addison-Wesley.

Shrivastava, S.K., et al. *A workflow and agent based platform for service provisioning*. in *Proceedings Fourth International Enterprise Distributed Object Computing Conference. EDOC 2000*. 2000. Los Alamitos, CA, USA: IEEE Comput. Soc.

Shaw, M., *Procedure Calls are the Assembly Language of Software Interconnection: Connectors Deserve First-Class Status*, in *Studies of Software Design, Proceedings of a 1993 Workshop, Lecture Notes in Computer Science No. 1078*, D.A. Lamb, Editor. 1996, Springer-Verlag.

Thompson, J.D., *Organizations in Action: Social Science Bases of Administrative Theory*. 1967, New York: McGraw-Hill.

Urwick, L.F., *Scientific Principles and Organization*. 1938, New York: American Management Association.

WFMC, The Workflow Reference Model, Version 0.6., http://www.wfmc.org/

Yan, Y., Z. Maamar, and W. Shen. *Integration of workflow and agent technology for business process management*. in *Proceedings of the Sixth International Conference on Computer Supported Cooperative Work in Design*. 2001. Ottawa, Ont., Canada: NRC Res. Press.