ACADEMIA DE STUDII ECONOMICE - Bucureşti
Academy Of Economic Studies - Bucharest

FACULTY OF BUSINESS ADMINISTRATION
(Facultatea de Administrare a Afacerilor cu predare în limbi străine)

Informatics for Business Administration

-

Systems for Enterprise Application Integration (EAI).

By: Professor Vasile AVRAM, PhD

- suport de curs destinat studenţilor de la sectia engleză - master aprofundat-
(lecture notes for 1st year students of english division – Master in Business Administration)
- anul I - Zi -

Bucureşti 2008

Chapter 5: Systems for Application Integration (EAI)

5.1 Definition

It is clear that we now live in a world where one expects everything to be connected, or better, integrated. This, however, is easier said than done. Unfortunately, organizations have had a long history of creating systems that operate in isolation from other systems. The integration is not an entirely new problem. It exists and is as old as Information Technology itself. What is new is the pressure of the necessity to integrate systems: they are too many to be coordinated in isolation as a whole, too costly, and became inefficient. Many organizations have undertaken specific projects in the past where they had to integrate a number of different systems. The challenge faced by organizations today is to find scalable and economical solutions to the problem of large-scale integration (defined by the term enterprise integration – EI – which denotes the need to integrate a large number of systems that may be highly distributed across different parts of the organization). Integration cannot be achieved unless the information technology used by the organization is also integrated.

"The goal of EAI is to integrate and streamline heterogeneous business processes across different applications and business units while allowing employees, decision makers, and business partners to readily access corporate and customer data no matter where it resides. More and more, EAI involves integrating information and processes not only across the enterprise but also beyond organizational walls to encompass business-to-business (B2B) integration supporting large scale value-added supply chains across the enlarged worldwide economy"[1].

EAI can be defined as "the use of software and architectural principles to bring together (integrate) a set of enterprise computer applications".

David S. Linthicum defined EAI as: "The unrestricted sharing of data and business processes among any connected applications and data sources in the enterprise."

Wing Lam and Venky Shankararaman (eds) [WLVS-2007] defines enterprise integration as "[t]he strategic consideration of the process, methods, tools and technologies associated with achieving interoperability between IT applications both within and external to the enterprise to enable collaborative business processes." Enterprise integration is an activity that:

- Is business driven rather than technology driven;
- Coordinates business processes with and across different parts of the enterprise;
- Involves multiple stakeholders;
- Adopts a strategic rather than tactical or localized view.

Enterprise integration is something that affects many different parts of the organization; the problem is not confined to one particular part of the organization. As such, success can only be achieved if the various departments buy into enterprise integration and share the same vision of how to achieve it. Enterprise integration is a long-term initiative that, in some cases, may take years to complete depending upon the number of systems to be integrated and the complexity of the integration challenge. Organizations therefore need to think carefully about how to plan and roll out the enterprise integration initiative.

[1] Semantic Web Technologies and E-Business: Toward the Integrated Virtual Organization and Business Process Automation by A. F. Salam and Jason R. Stevens IGI Publishing © 2007

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

> "Integration: Harmonized technology
>
> Integration is a word that's used throughout the world of IT in many different ways, and with many different meanings. Almost all uses can be boiled down to two meanings.
>
> Most often the word integration means the work done to connect two or more separate things so that they easily work together. Personal finance programs, for example, offer credit card bill integration that allows monthly credit card bills to be automatically loaded into the personal finance program from the Web site of the credit card company. This is a simple integration that allows the information from one source to be made useful in another context. For IT, integration frequently is used to automate manual processes. In personal finance, the integration avoids the need to retype the credit card bill into the personal finance program. Implementing a few simple integrations like this at a company can save millions every year.
>
> The second and closely related sense of the word integration or integrated is the idea of working together in an optimized way. People integration means that the people are working together in a better way through some system than if they weren't integrated at all. Process integration means that processes now work together. For static items like information, integration means that information was brought into a common structure that allows the data to paint a larger picture or tell a larger story.
>
> The goal of information integration is to create one version of the truth, a consistent perspective on all activities and deliverables visible to all."
>
> Source: SAP NetWeaver For Dummies, by Dan Woods and Jeffrey Word, John Wiley & Sons © 2004

5.2 EAI tools and categories

The goal of EAI is to integrate and streamline heterogeneous business processes across different applications and business units. To do that EAI includes a set of tools used to help integrating heterogeneous applications.

EAI tools have three main components (figure 5.1):

1. an integration broker that serves as a hub for intersystem communication (it performs a number of functions such as multi-format translation, transaction management, monitoring, and auditing);
2. a set of adapters that enables different systems to interface with the integration broker (an adapter acts a gateway or wrapper that provides the means by which packaged applications, database applications, legacy systems, and custom applications can connect to the integration broker);
3. an underlying communications infrastructure (such as a reliable high-speed network that enables systems to communicate with each other using a variety of different protocols).

During the time we have been used multiple technologies to develop applications and distributed applications components such as:

- Remote Procedure Call (RPC);
- Common Object Request Broker Architecture (CORBA);
- Distributed Component Object Model (DCOM);

- .NET remote;
- Enterprise Java Beans (EJBs);
- Java Remote Method Invocation (Java RMI).

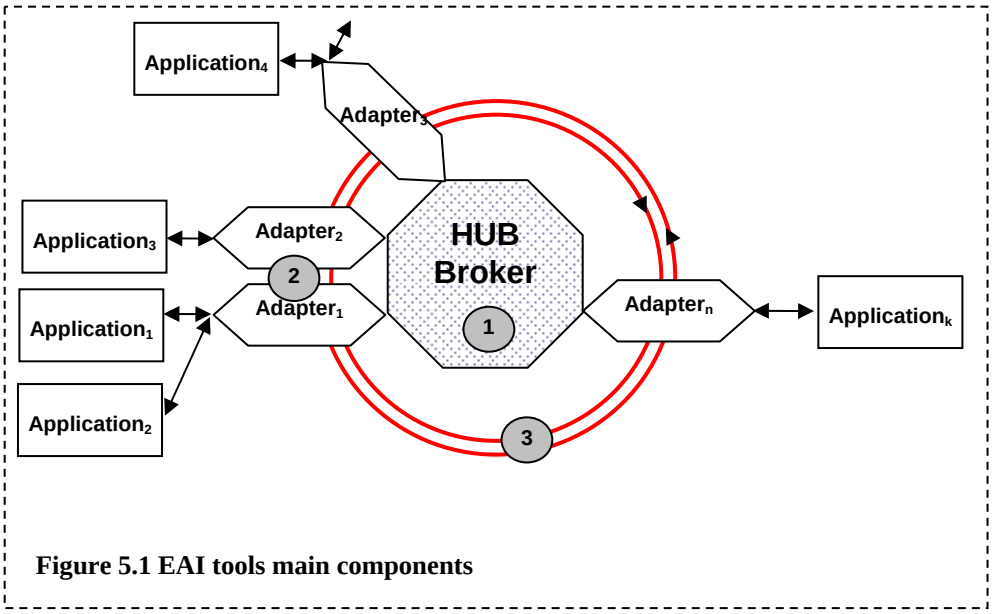


Figure 5.1 EAI tools main components

Most enterprises contain a systemic infrastructure of several heterogeneous systems, such as SCM, CRM, PLM, HR, ERP, databases applications, business portal, legacy systems etc, creating a complex, fuzzy network of interconnected applications, services, and data sources, which is not well documented and expensive to maintain (figure 5.2, left). The goal of EAI is to unify and make cooperating all these systems (figure 5.2, right).

Depending on who is the master of integrated parts the EAI can be categorized as:

- **Intra-organizational**, or application-to-application (**A2A**) integration, that specifies the automated and event-driven exchange of information between heterogeneous enterprise applications and systems operating within an organization or enterprise;
- **Inter-organizational**, or business-to-business (**B2B**) integration, that specifies the automated and event-driven information exchange between various systems of several collaborating organizations and enterprises.

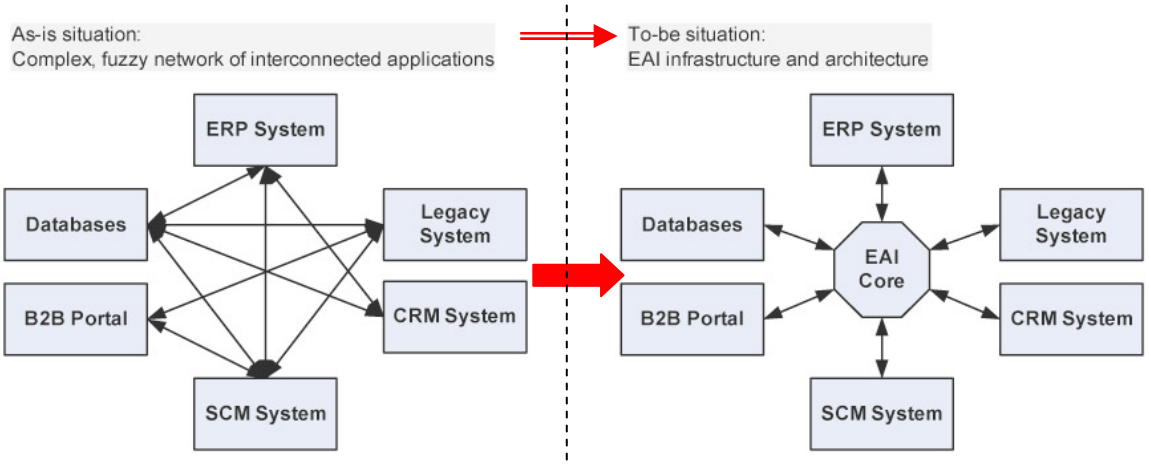


Figure 5.2 The enterprise system environment - With and without an EAI system (Source: Semantic Web Technologies and E-Business: Toward the Integrated Virtual Organization and Business Process Automation by A. F. Salam and Jason R. Stevens IGI Publishing © 2007**)**

EAI can take place at different layers (Bussler, 2003), such as:

- **data-oriented integration**, occurring at the database and data source level, either real time or non-real time, constituting the most widespread form of EAI today;
- **function or method integration**, involving the direct and rigid application-to-application integration of cross-platform applications over a network — it can be achieved using custom code, application program interface (APIs), remote procedure calls (RPCs) or distributed middleware and distributed objects (CORBA, Java RMI, DCOM);
- **user interface integration**, consisting on using a standardized user interface for accessing a group of legacy systems and applications. The new presentation layer is integrated with the existing business logic of the legacy systems or packaged applications; and
- **business process integration**, occurring at the business process level.

5.2.1 Enterprise Application Integration via Web services

The companies require both hardware and software to realize their processing. Instead of buying and installing software programs they can use Internet or private networks where, by paying a subscription, can rent the same functions from application service providers. An Application Service Provider (ASP) is a business that delivers and manages applications and computer services from remote centers to multiple users via the Internet or private network. Today's Internet-driven business environment is changing so rapidly that getting a system up and running in three month instead of six could mean the difference between success and failure so that the ASP is a competitive alternative. The ASP enables also small and medium-sized companies to use applications that they otherwise could not afford. The applications offered in that way generally have a proprietary architecture and functionality and the customization do not produces, in all cases, the desired adaptation to the company needs. To avoid all that impediments a new way of deploying and using services emerges, known as Web services.

Web services defined by W3C Web Services Architecture Working Group as: "A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards."

Web services are software components deliverable over the Internet that enable one application to communicate with another with no translation required. By allowing applications to communicate and share data regardless of operating system, programming language, or client device, Web services can provide significant cost savings, over traditional in-house development. Web services provide a standard means of interoperating between different software applications, running on a variety of platforms and/or frameworks, in other words, heterogeneous in different points.

Web services are self-contained, self-describing, modular applications that can be published, located, and invoked across the Web. They perform functions, which can be anything from simple requests to complicated business processes. Once a Web service is deployed, other applications (and other Web services) can discover and invoke the deployed service. Web services offer a new and evolving paradigm for building distributed web applications. The basic model of using web can be described as human-centric web: a user addresses a query to a web server, via a web browser, and in turn the web server answers to the user by using the same web. The web services allows to move from a human-centric web to an application-centric web, this means the applications will realize direct conversations as easily as the ones realized between web browsers and servers.

> Web Services have emerged as a solution to problems associated with distributed computing. The previous technologies, primarily Common Object Request Broker Architecture (CORBA) and

Distributed Component Object Model (DCOM), had some limitations. For example, neither has achieved complete platform independence or easy transport over firewalls. Additionally, DCOM is not vendor independent, being a Microsoft product.

A Web Service forms a distributed environment, in which objects can be accessed remotely via standard interfaces. A Web service performs a specific task or a set of tasks, such as credit card processing, production scheduling, security, third-party billing and payment, for example. Web service uses a three-tiered model, defining three actors: service provider, service consumer, and service broker. This allows the Web Service to be a loose relationship, so that if a service provider goes down, the broker can always direct consumers to another one. Similarly, there are many brokers, so consumers can always find an available one. For communication, Web Services use open Web standards as: HyperText Transfer Protocol (HTTP), Extensible Markup Language (XML), Simple Object Access Protocol (SOAP), Web Services Description Language (WSDL), and the Universal Description, Discovery, and Integration (UDDI) project. On the long term we expect that web services allows building the automated Web.

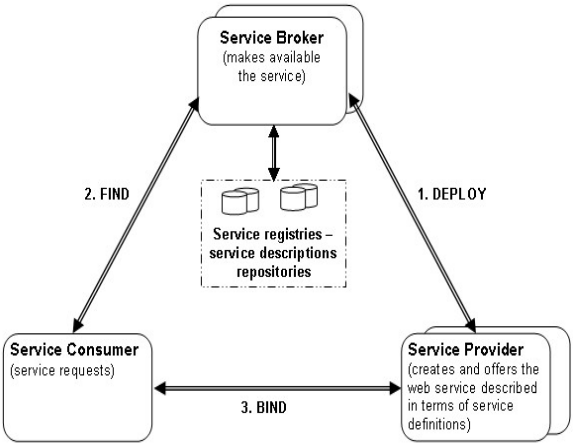


Figure 5.3 Integration using Web services model

For both service consumer and service broker the Web service is available as an interface that describes a collection of operations that are network-accessible through standardized XML messaging. The Web service is described by using a standard, formal XML notation based on SOAP, called service description. The service description is realized by using WSDL (Web Services Description Language) and provides all of the details necessary to interact with the service, including message formats (that details the operations), transport protocols, and location.

The Web service is created, defined and deployed by **service provider** (figure 5.3). A service provider creates a Web service and its service definition and then **publishes** (1) the service in a service registry (or directory) based on the standard Universal Description, Discovery, and Integration (UDDI) specification.

Once a Web service is published, a **service requester** may **find** (2) the service via the UDDI interface. The UDDI registry provides the service requester with a WSDL **service description** and a URI pointing to the service itself. The service requester may then use this information to directly **bind** (3) to the service and invoke it.

5.2.2 Service-Oriented Architecture

Service-Oriented Architecture (SOA) is nothing more than an approach to designing a more innovative computing platform that takes advantage of reusable services to build powerful business processes (figure 5.4). SOA provides for definitions and methods of building an IT infrastructure that makes it possible not only to exchange data between different systems and other data repositories, but to build and extend business processes.

As such, SOA is necessarily tied to both the underlying operating systems as well as the development/programming languages native to an application. But the link is a loosely coupled one; in this way, it is possible to support different operating environments spanning traditional UNIX offerings, Linux, Windows, and even mainframes.

Though not synonymous, Web Services represents the vehicle that makes Enterprise Services possible. They are beginning to become the new standard for inter-application communication.

The idea is that Web Services are "open" rather than tied to a particular type of technology or hardware platform or software vendor.

This platform independence, therefore, makes it possible to communicate between vastly different technological platforms.

Enterprise Services, on the other hand, extend the use of Web Services by combining them into a usable context as they relate to a business process. In this way, they become reusable components that may be shared across partners and organizations to foster real innovation.

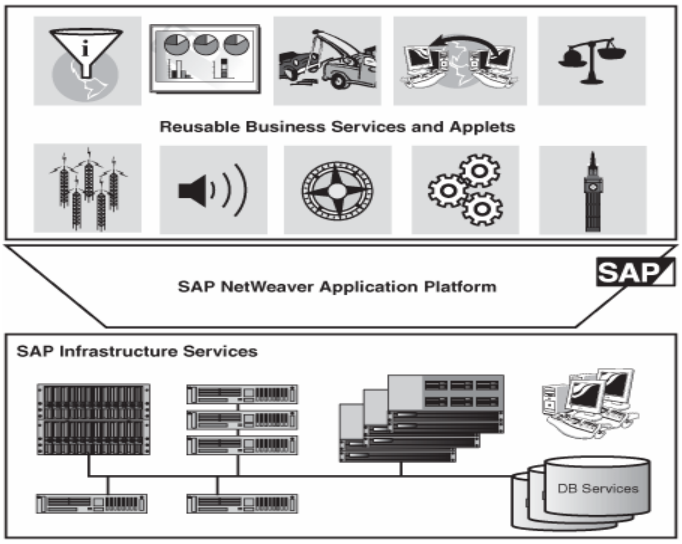


Figure 5.4 SOA allows SAP to be easily extended to take advantage of readily available service-based applications and utilities (Source: Sams Teach Yourself SAP in 24 Hours, Third edition by George W. Anderson, Tim Rhodes, Jeff Davis, John Dobbins, Andreas , August 06 2008)

The Term “SOA”

The following are common examples of SOA definitions. Most of these definitions focus on the technical aspects of architecture, although some include business characteristics. These definitions, which we gathered from multiple sources, are interesting because they illustrate the number of different views or expressions of what an SOA is; however, they all tend to coalesce into a common meaning:

- A business definition: A set of business, process, organizational, governance, and technical methods to reduce or eliminate frustrations with IT and to quantifiably measure the business value of IT while creating an agile business environment for competitive advantage.
- Another business definition (introduced by IBM): A service-oriented architecture provides the flexibility to treat elements of business processes and the underlying IT infrastructure as secure, standardized components (services) that can be reused and combined to address changing business priorities.
- The widest technical (and rather minimalist) definition: An enterprise-wide IT architecture that promotes loose coupling, reuse, and interoperability between systems.
- A moderately complex technical definition: An application architecture in which all functions or services are defined using a description language and have callable interfaces that are called to perform business processes. Each interaction is independent of each and every other interaction and the interconnect protocols of the communicating devices. Because interfaces are platform independent, a client can use the service from any device using any operating system in any language.
- The least common denominator definition: A system architecture in which application functions are built as components (services) that are loosely coupled and well-defined to support interoperability and to improve flexibility and reuse.
- The narrowest definition: SOA is a synonym for solution architectures making use of Web service technologies such as SOAP, WSDL, and UDDI. Here SOA is defined as “any product and project architecture conforming to the W3C Web services architecture (WSA).”

Source: Service-Oriented Architecture Compass: Business Value, Planning, and Enterprise Roadmap by Norbert Bieberstein et al., IBM Press © 2006

SOA can fundamentally change the direction taken by an enterprise to development and deployment of business software systems for competitive advantage. Introducing SOA at the enterprise level cannot reasonably be reduced to a single project because it will affect the way that IT responds to the business requirements. But to say that service-oriented architectures are here to solve all of our technical and business problems would be, of course, a gross exaggeration. There

are numerous other problems that introduce hurdles for interoperability, compatibility, and well-mannered operation among diverse systems. Furthermore, like any other technological system, SOA deployment requires careful business-case evaluation, planning, design, implementation, and management by a properly skilled and talented team to become a successful endeavor.

SOA is considered "the design principle and mechanism for defining business services and computing models, thus effectively aligning business and IT (Figure 5.5; Newcomer & Lomow, 2005)".

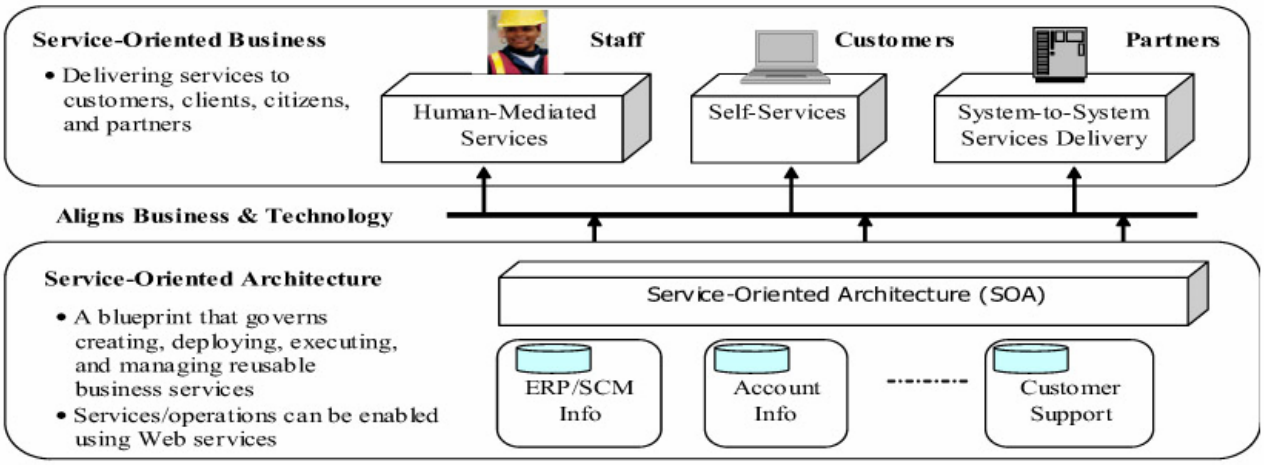


Figure 5.5 SOA – aligning business and IT (Source: Enterprise Service Computing: From Concept to Deployment by Robin G. Qiu IGI Publishing © 2007)

To properly implement service-oriented IT systems complying with SOA, three major levels of abstraction throughout collaborated IT systems are necessary:

- **Business processes:** A business process typically consists of a set of actions or activities executed with specifically defined long-term business goals. A business process usually requires multiple computing services. Service invocations frequently involve business components across the network. Examples of business processes are the initiation of a new employee, the selling of products or services, a project's status, and order-fulfillment information.
- **Services:** A service represents a logical group of low-level computing operations. For example, if customer profiling is defined as a service, then looking up customers from data sources by telephone number, listing customers by name and postal code on the Web, and updating data for new requests represent the associated operations.
- **Operations:** A computing operation represents a single logical unit of computation. In general, the execution of an operation will cause one or more data sets to be read, written, or modified. In a well-defined SOA implementation, operations have a specific, structured interface and return structured responses. An SOA operation can also be composed of other SOA operations for better structures and maintainability.

Source: Service-Oriented Architecture Compass: Business Value, Planning, and Enterprise Roadmap by Norbert Bieberstein et al., IBM Press © 2006

5.2.3 Drivers for enterprise integration

The EAI drivers can be broadly grouped in two categories: **project drivers** and **organizational drivers** (figure 5.6).

Some of **project drivers** are represented by:

- **e-Business** - integrating legacy IT systems containing core business functionality with front-end Web-based systems such as application servers and portals;

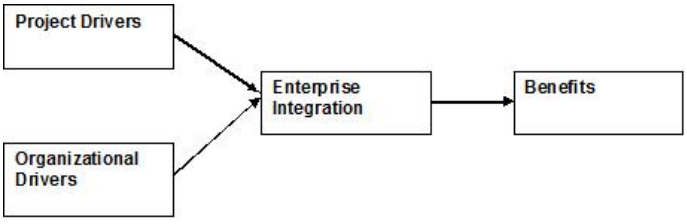


Figure 5.6 EAI drivers

- **Supply chain management** - integrating disparate IT systems for order management, manufacturing resource planning, scheduling, and logistics with direct exchange of real-time information between buyers and suppliers to optimize the supply chain and streamline workflow;
- **ERP expansion** - the integration of ERP systems that provide core functionality with more specialized IT systems used within the organization;
- **Customer relationships management** - building a single, consolidated view of the customer by integrating customer data that are distributed across different IT applications (integration of customer databases, call-centre, and Web-based customer service systems);
- **E-Government** - integrating legacy back-end government systems with front-end Web-based systems and/or enabling intra- and inter-government agency workflows and data exchange;
- **Customer self-service** - enabling customers to perform services traditionally performed by staff by integrating front-end customer service applications with back-end IT systems. Integrating customer service IT applications running over multiple channels (Web, digital TV, mobile) using customer service workflow events to trigger IT system services and processing;
- **Customer expectations** - meeting customer expectations for immediate access to real-time services and information that require the integration of IT systems. Achieving straight-through processing (STP) for increased efficiency, particularly for those operating in the financial-services sector;
- **Knowledge management** - real-time search and access to knowledge content that is distributed across multiple knowledge sources. The management of knowledge assets within the enterprise;
- **Business intelligence** - the collation of data from a variety of IT applications and data sources into a data warehouse. The mining of data from multiple sources to identify patterns and trends;
- **IT systems consolidation**.

Some of the **organizational drivers** are represented by:

- **Corporate mergers, acquisitions and partnership** - consolidating or merging duplicate or overlapping IT systems that perform similar functionality;
- **Industry regulations and deregulations** – forcing monolithic IT systems that supported a spectrum of business services to be broken up into smaller, more manageable IT systems that are loosely integrated. Government and industry regulations that force systems to be integrated.
- **Globalization** - The integration and consolidation of data from IT systems that reside in different countries and regions. The enablement of business processes and workflows that cut across multiple organizational and regional units;

Some of the **benefits** that can be obtained by integration are represented by:

- Greater business responsiveness;
- Improved customer service and support;
- Business process automation;
- Shorter processing cycle;
- Reduced processing errors;
- Process visibility;
- Lower transactions costs;
- Improved supply-chain management ;
- Closer business partnerships;
- Rapid introduction of new business services;
- Future integration path;
- Lower IT maintenance costs.

5.3 EAI – layers and architectures

5.3.1 EAI - a layered approach

In a layered approach each successive level of integration represents what might be considered a more advanced form of integration, as shown in figure 5.7

- **Presentation integration:** The aggregation of data from multiple IT systems within a single view. A Web portal that aggregates and displays a customer's stock portfolio taken from one IT system and bank account balance from another IT system can be considered integration at the presentation level. However, there is no direct communication between the individual IT systems.
- **Data integration:** The synchronization of data held in different databases. For example, if two different databases hold the same customer's address details, a change of address in one database should also by synchronized in the other database. If this synchronization takes place only after a period of time, a certain amount of data freshness will be lost.
- **Application integration:** Where applications make some of the functionality directly accessible to other applications. Popular packaged applications, for example, such as SAP and PeopleSoft, often expose their functionality through well-defined application programming interfaces (APIs).

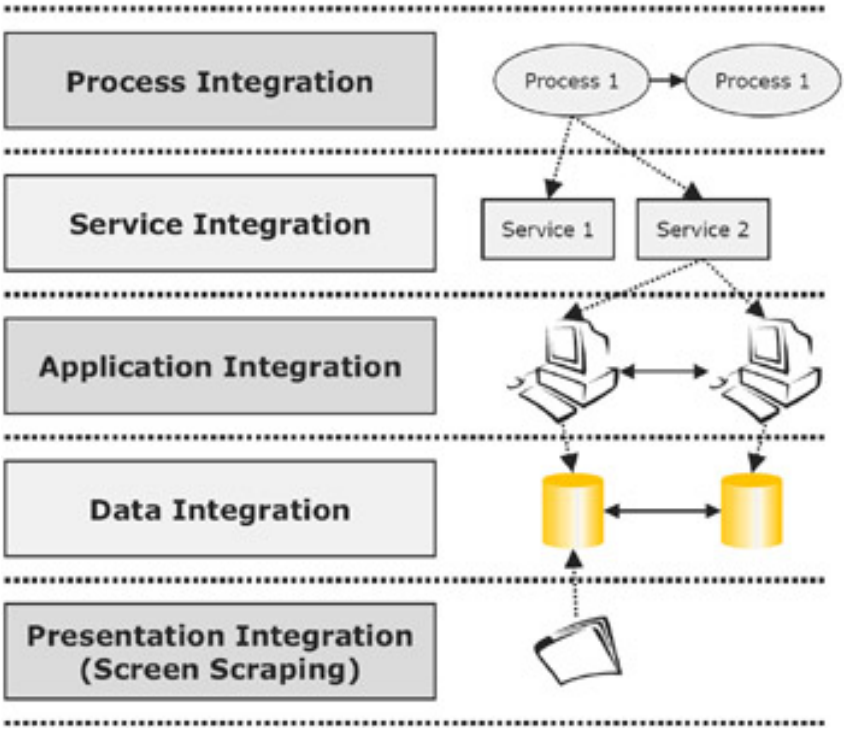


Figure 5.7 The layers of integration

- **Service integration:** A common set of reusable services that are made available to other applications. For example, a service to obtain a customer's status may be created that can be called by any other application within the organization. Such services are usually built from a set of specific application functions.
- **Process integration:** The definition of business-process or workflow models from which reusable services are called. Process integration is particularly relevant in collaborative contexts, such as B2B, where there business processes drive interactions and transactions between business partners.

5.3.2 Integration Architectures

We do not have yet a standard for integration architectures. What we have is a set of tools together with a set of standards for languages and protocols used for integration purposes. Here are introduced two approaches that categorize the integration architectural types.

I. BPBB

Wing Lam and Venky Shankararaman (eds) introduces in [WLVS-2007] four basic kinds of integration architecture:

1. batch integration;
2. point-to-point integration;
3. broker-based integration;
4. business-process integration.

Each kind of integration architecture represents differing levels of sophistication, as illustrated in figure 5.8.

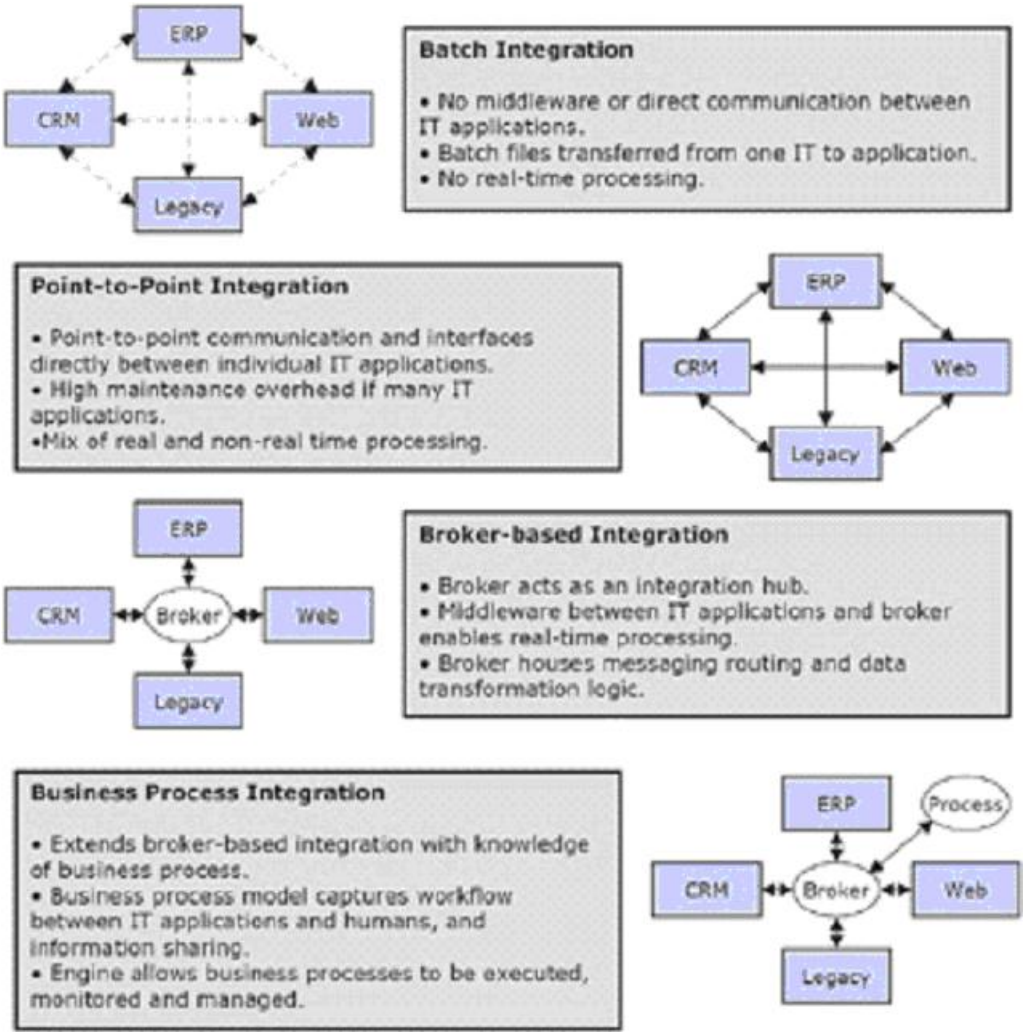


Figure 5.8 The four integration architectures

Each kind of integration architecture has its own capabilities and limitations.

Batch integration architecture:

- IT applications communicate asynchronously with each other
- External files are used to transfer data from one IT application to another
- There is no business requirement for real-time processing

A batch integration architecture is often suitable for organizations that perform high-volume back-end processing that is able to take place overnight, weekly, or on a scheduled basis.

Point-to-point integration architecture:

- IT applications communicate with other IT applications through interfaces
- Communication supported by these interfaces may be real time or asynchronous
- The number of interfaces to be built rapidly grows as the number of IT applications increases

A point-to-point integration architecture is most suited when the number of IT applications to be integrated is small. However, as the number of IT applications grows, the number of interfaces to be maintained becomes problematic, and a broker-based integration architecture becomes more appropriate.

Broker-based integration architecture:

- An integration broker acts as a hub for connecting IT systems. A good analogy is to view an integration broker as a major airport that serves as a hub for connecting flights to smaller cities.
- Can only be implemented through special tools having as the main functions of the integration broker the following:
  - **Message routing:** Routing data from one IT system to another, for example, routing banking transactions taken from the call centre to the legacy banking system
  - **Schema translation:** Translating data given in one format or schema definition into data in another schema definition, for example, Siebel data fields mapped onto CICS records
  - **Business-rules processing:** Applying business rules based on the content of the data or messages, for example, raising a warning flag if the account balance is less than zero
  - **Message splitting and combining:** Splitting or combining messages
  - **Transaction management:** Managing a sequence of messages as a single transaction

A broker-based integration architecture supports real-time integration. Such tools provide the basic integration broker and messaging infrastructure, but adapters must be purchased to interface with the kind of IT systems that reside within the organization.

Business-process integration architecture: builds upon the broker-based integration architecture by adding the concept of business processes that trigger the relevant messages to be sent to individual IT systems. For example, when a banking customer opens up a new account, the business process may involve doing a background check, creating an account if the background check is positive, and then sending a notification to the customer. Each step in the business process involves communication with different IT systems.

II. PHEE

In [BC-2008] the basic integration architectures are the following:

- **Point-to-Point**
- **Hub-and-Spoke**
- **Enterprise Message Bus**
- **Enterprise Service Bus**

In the following paragraphs is a brief description of each one of these architectures.

Point-to-Point

EAI has traditionally been done using *point-to-point* integration solutions meaning defining an integration solution for a pair of applications (with only two end points to be integrated). The adapters (protocol and/or format adapters, or transformers) can be build at one or to either end. This is the easiest way to integrate, as long as the volume of integration is low. The advantage is that between these two points, we have tight coupling, since both ends have knowledge about their peers.

Hub-and-Spoke

Hub-and-spoke architecture is also called the **message broker** and is similar to point-to-point architecture in that it provides a centralized **hub** (broker) to which all applications are connected. When multiple applications connect in this manner, we get the typical hub-and-spoke architecture. Another distinguishing feature of the hub-and-spoke architecture is that each application connects with the central hub through lightweight connectors. The **lightweight connectors** facilitates for application integration with minimum or no changes to the existing applications.

Message transformation and routing takes place within the hub. This architecture is a major improvement to the point-to-point solution since it reduces the number of connections required for integration. Since applications do not connect to other applications directly, they can be removed from the integration topology by removing from the hub. This insulation reduces disruption in the integration setup.

There is a major drawback with the hub-and-spoke architecture. Due to the centralized nature of the hub, it is a single point of failure. If the hub fails, the entire integration topology fails. A solution to this problem is to cluster the hub. A **cluster** is a logical grouping of multiple instances running simultaneously and working together. But clustering is not the right solution to the problem of single point of failure. This is due to the fact that the very point in having a hub-and-spoke architecture is to have a single point of control. This drawback may be offset to some extent by the fact that most of the clustering solutions provide central management and monitoring facilities, which will provide some form of centralized control.

Enterprise Message Bus

While the hub-and-spoke architecture makes use of lightweight connectors for applications to integrate together through a central hub, many a times the integrating applications need to interact in a decoupled fashion, so that they can be easily added or removed without affecting the others. An *enterprise message bus* provides a common communication infrastructure, which acts as a platform-neutral and language-neutral adapter between applications.

This communication infrastructure may include a message router and/or Publish-Subscribe channels. So applications interact with each other through the message bus with the help of request-response queues. If a consumer application wants to invoke a particular service on a different provider application, it places an appropriately formatted request message on the request queue for that service. It then listens for the response message on the service's reply queue. The provider application listens for requests on the service's request queue, performs the service, and then sends (if) any response to the service's reply queue.

We normally use vendor products like IBM's Websphere MQ (WMQ) and Microsoft MQ (MSMQ), which are the best class message queue solution to integrate applications in the message

bus topology. Sometimes the applications have to use adapter which handle scenarios, such as invoking CICS transactions, for example. Such an adapter may provide connectivity between the applications and the message bus using proprietary bus APIs, and application APIs. The Message bus also requires a common command structure representing the different possible operations on the bus. These command sets invoke bus-level primitives which includes listening to an address, reading bytes from an address, and writing bytes to an address.

Enterprise Service Bus Integration

The service bus approach to integration makes use of technology stacks to provide a bus for application integration. Different applications will not communicate directly with each other for integration; instead they communicate through this middleware Service Oriented Architecture (SOA) backbone. The most distinguishing feature of the ESB architecture is the distributed nature of the integration topology. Most ESB solutions are based on Web Services Description Language (WSDL) technologies, and they use Extensible Markup Language (XML) formats for message translation and transformation.

ESB is a collection of middleware services which provides integration capabilities. These middleware services sit in the heart of the ESB architecture upon which applications place messages to be routed and transformed. Similar to the hub-and-spoke architecture, in ESB architecture too, applications connect to the ESB through abstract, **intelligent connectors**. Connectors are abstract in the sense that they only define the transport binding protocols and service interface, not the real implementation details. They are intelligent because they have logic built-in along with the ESB to selectively bind to services at run time. This capability enhances agility for applications by allowing late binding of services and deferred service choice.

The above qualities of ESB provides for a true open enterprise approach. As we have discussed above, ESB is neither a product nor a separate technology; instead, ESB is a platform-level concept, a set of tools, and a design philosophy. What this means is, if we just buy a vendor product and install it in our IT infrastructure, we cannot say that we have ESB-based integration architecture. Instead ESB-based integration solutions are to be designed and built in the "ESB way". Tools and products help us to do this.

A list of major features and functionalities supported by an ESB will help us to understand the architecture better, which are listed as follows:

- Addressing and routing
- Synchronous and asynchronous style
- Multiple transport and protocol bindings
- Content transformation and translation
- Business process orchestration
- Event processing
- Adapters to multiple platforms
- Integration of design, implementation, and deployment tools
- QOS features like transactions, security, and persistence
- Auditing, logging, and metering
- Management and monitoring

5.3.3 Enterprise Integration Projects

The enterprise integration projects generally fall into one of the following project categories:

1. Enterprise application integration (EAI) - concerned with integrating the IT applications that reside within the organization (for example, the integration of the customer-accounts IT system with the order-management IT system);
2. Business-to-business integration (B2Bi) - concerned with integrating an organization's IT system with those of its business partners or suppliers such as in an extended supply chain;
3. Web integration - concerned with integrating an organization's IT applications with Web applications to provide a Web channel.

Table 5.1 realizes a comparison of the three enterprise integration projects - EAI, Web integration, and B2B projects – considering the project scope, drivers, and key challenges.

Table 5.1 EAI, Web integration, and B2B projects categories comparison

	EAI	B2Bi	Web Integration
Scope of Integration	Intra-organization: IT systems within the organization	Inter-organization: IT systems between different organizations	Intra-organization: Existing IT systems with new Web-based front-end applications
Typical Project Drivers	Operational efficiency, customer relationship management, business-process automation	Supply chain management, B2B commerce	E-business, Web-channel services
Key Challenges	Business-process management, workflow	Document and data exchange standards	Online security, transaction management

5.3.4 Systems for EAI

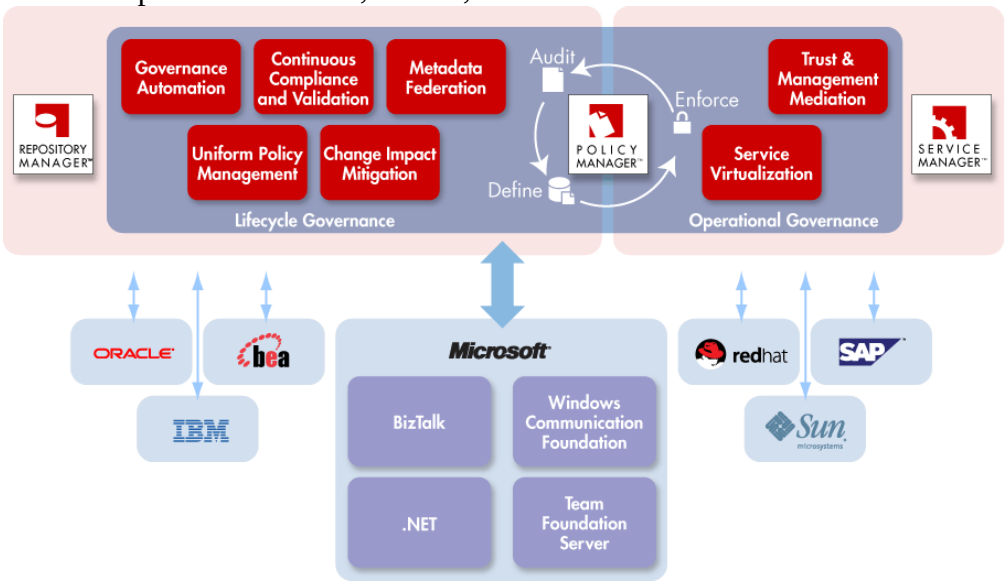
IBM® WebSphere® Enterprise Service Bus provides Web services connectivity and JMS™ messaging (http://www.ibm.com/soa)

It applies Web services connectivity and JMS (Java Messaging System) messaging, improving flexibility through the adoption of service-oriented interfaces.

- Provides a smart approach to SOA, delivering a standards-based connectivity and integration solution that allows you to create and deploy interactions quickly and easily between applications and services, with a reduced number and complexity of interfaces.
- Offers easy-to-use tools that require minimal programming skills and is simple to install, configure, build and manage.
- Supports hundreds of ISV (Independent Software Vendor) solutions via WebSphere Adapters.
- Provides leadership in SOA standards for service composition, mediation, and hosting.
- Increases business agility and flexibility extending easily to a Federated ESB model.
- Re-configures dynamically to meet changing business processing loads. Provides easy interactions with any JMS and HTTP applications.
- Integrates seamlessly with the WebSphere platform as well as products within the IBM SOA.
  Foundation, such as IBM Tivoli Composite Application Manager for SOA.
- Operating systems supported: AIX, HP Unix, i family, Linux, Sun Solaris, Windows, z/OS.

Microsoft & SOA Software (http://www.soa.com/index.php/section/Microsoft/)

SOA Software products provide a comprehensive Integrated SOA governance solution including a high-performance, scalable SOA security, mediation, and management (Service Manager), and a comprehensive policy management system (Policy Manager). SOA Software's Service Manager and Policy Manager products enhance Microsoft's Governance capabilities on both IIS and Biztalk platforms. Policy Manager provides an integrated Service Management, Registry/Repository and Policy management solution, allowing Microsoft's field engineers to respond to a customer's end-to-end SOA Governance requirements. The solution integrates with BizTalk as a BizTalk pipeline and provides full service lifecycle management along with features such as design-time compliance testing, service discovery via a UDDI v2/v3 interface, service provisioning, WS-S security and service monitoring regardless of protocol.
SOA Software's Policy Manager and Service Manager products are compatible with the following Microsoft products: IIS 5.0 , IIS 6.0, BizTalk 2006



SOA Software's products integrate seamlessly with Microsoft BizTalk to ensure the security, reliability, interoperability and governance of the services it provides and consumes. The SOA Software products that make up the solution for BizTalk include:
Policy Manager™ ; SOA Software Service Manager Network Director ; SOA Software Agent for Microsoft BizTalk

Pervasive Data integration tools for a Range of Needs (http://www.pervasive.com)

Pervasive Data Integrator — a proven, flexible data integration tool for aggregating data from various sources into data warehouses, data marts or other target systems on a continuous, event-driven, or regularly scheduled basis.
Pervasive Data Profiler — This data profiling solution eliminates the complexities, risk, and expense associated with manually checking data. Versatile and easy to use, it improves IT productivity, eliminates costly rework, and increases ROI around applications and business processes.
Pervasive Integration Manager — A browser-based management console providing single-point administration, performance monitoring and job scheduling of integration processes running on remote Integration Engines.
Oracle Adapters — Pervasive Data Adapters for Oracle Application Server enable rapid access to heterogeneous data and offer the unique capability to process non-relational and semi-structured content that is buried throughout most companies. Our adapters eliminate the complexity of low-level interface coding, while reducing cost, time, and risk.

Pervasive MessageStore — This is a data collection framework that allows companies to receive, integrate, store, audit and retrieve data sent in message formats such as XML or EDI, and in industry standards such as HIPAA, HL7, or ACORD. It provides companies in healthcare, insurance, and other industries with a high value, low TCO way to become compliant with these industry standards without custom coding, having to learn the complexities of the standards themselves, or becoming certified in those standards.
Pervasive Metadata Studio — Pervasive Metadata Studio automatically captures extensive design-time and production metadata in a single repository for powerful analysis of integration processes, designs and inter-relationships. Business and technical end users can search on any data element, any function/rule, source or target connection, map or process design for impact analysis, lineage and reports for team collaboration.

Sun ONE Integration Server EAI Edition 3.1(http://www.sun.com)

The Sun ONE Integration Server EAI Edition, is a software product designed for enterprises that need to integrate packaged, custom, legacy, and new Java applications. The product makes it possible for companies to integrate core business processes with multiple applications running on multiple operating systems across multiple communication protocols within the enterprise. Companies benefit from the automation of business processes across distributed heterogeneous information systems with increased productivity and efficiency.
At a Glance: Provides integration for Web services through SOAP messaging, which can export a service definition via WSDL to UDDI or other SOAP clients.

Key Features:

- Standards-based design
- Web services through SOAP support
- Business process management (BPM) capabilities
- Multiplatform support, including mainframes

Sun ONE Integration Server EAI Edition software is the integration solution for the Sun Java System (formerly Sun ONE) architecture - the hub that brings together a new generation of Web services. The Sun ONE Integration Server EAI Edition now includes Simple Object Access Protocol (SOAP) support, a remote procedure call form of the eXtensible Markup Language (XML) that invokes a call on another application using SOAP and the Web. It allows different parts of the enterprise to exchange structured information over the Web independent of the types of systems or application platforms. SOAP functions as a wire protocol to connect Web services to multiple Web portals.

Oracle Application Integration Architecture (http://www.oracle.com)

Oracle Application Integration Architecture delivers an open standards based framework for orchestrating cross-application business processes that help you connect and optimize all your applications. Application Integration Architecture (AIA) is the most complete integration solution for orchestrating agile, user-centric business processes across your enterprise applications. With Application Integration Architecture organizations can:
- Gain business and IT efficiencies
- Increase their competitive edge
- Accelerate innovation

With AIA , organizations can leverage a complete set of tools, templates and methodologies to do more with their existing IT investments, it gives business and IT more flexibility and choices and lowers their overall IT total cost of ownership (TCO).
Oracle Application Integration Architecture is a comprehensive set of products that delivers sustainable business process based integrations across Oracle, third party and custom applications. With its state of the art, open, standards based foundation, Oracle Application Integration Architecture enables you to integrate your broad portfolio of applications for greater business efficiency and responsiveness.
Rapidly automate end-to-end business processes across your front- and back-office systems to increase user productivity. Reuse, update and replace processes more easily. Start moving at the speed of business and create the differentiating application capabilities you need, when business demands.
Oracle Application Integration Architecture is comprised of three key components:

- Industry Reference Models – optimize business performance through documented industry best practices processes
- Process Integration Packs – quickly integrate Oracle applications to create the business processes you need, without the risk
- Pre-built SOA leveraging Fusion Middleware SOA Suite – accelerate time to value through a pre-built SOA designed for maximum flexibility, maintainability and reuse.

Together, these components provide the foundation for adaptability within a business process and include the essential tools you need to create cross application business processes, allowing you to bring your applications together in a way that yields the greatest competitive advantage to your business all while keeping your costs low.

References

Books

1. [BC-2008] Service-Oriented Java Business Integration: Enterprise Service Bus Integration Solutions for Java Developers, by Binildas A Christudas, Packt Publishing © 2008 (436 pages) ISBN:9781847194404
2. [WLVS-2007]Enterprise Architecture and Integration: Methods, Implementation and Technologies , by Wing Lam and Venky Shankararaman (eds), IGI Publishing © 2007 (364 pages) ISBN:9781591408871
3. [ASJS-2007] Semantic Web Technologies and E-Business: Toward the Integrated Virtual Organization and Business Process Automation , by A. F. Salam and Jason R. Stevens, IGI Publishing © 2007 (450 pages) ISBN:9781599041926
4. [NP-2008] Agent and Web Service Technologies in Virtual Enterprises, by Nicolaos Protogeros (ed), IGI Publishing © 2008 (408 pages) ISBN:9781599046488
5. [NB-2006] Service-Oriented Architecture Compass: Business Value, Planning, and Enterprise Roadmap, by Norbert Bieberstein et al. IBM Press © 2006 (272 pages) ISBN:9780131870024
6. [VS-2008] Intelligent Information Technologies: Concepts, Methodologies, Tools, and Applications, by Vijayan Sugumaran (ed), IGI Publishing © 2008 (2469 pages) ISBN:9781599049410

On line resources

1. www.ibm.com
2. www.accenture.com
3. www.soa.com
4. www.pervasive.com
5. www.sun.com
6. www.oracle.com
7. www.sap.com