

## EAI PRIMER



WHITE PAPER

Gourav Suri  
Nikhil V. Kapre

Mphasis Integration Architecture Team

May 2008

### Executive Summary

A lot is being said and written about 'EAI'. 'SOA' is the new silver bullet that has become the talk of the town.

This paper attempts to explain the various terms that are being used as 'jargon' by some and as architectural pattern/ design pattern/ implementation methodology/ tool(s) by others.

## Table of Contents

|  |    |
|--|----|
| 1. INTRODUCTION .....  | 2  |
| 2. WHAT IS ENTERPRISE APPLICATION INTEGRATION (EAI)? .....   | 2  |
| 3. WHY EAI? .....  | 2  |
| 4. WHAT ARE THE VARIOUS TYPES OF EAI SOLUTIONS? .....  | 3  |
| 5. WHAT ARE THE BASIC COMPONENTS REQUIRED TO ACHIEVE EAI? .....  | 4  |
| 6. WHAT TECHNOLOGIES ARE USED IN IMPLEMENTING EACH OF THE COMPONENTS IN THE EAI SOLUTION?...   | 4  |
| 7. WHAT ARE THE VARIOUS INTEGRATION PATTERNS THAT EAI SOLUTIONS IMPLEMENT? .....   | 5  |
| 8. WHAT ARE THE TWO BASIC ARCHITECTURES USED TO IMPLEMENT EAI? .....   | 5  |
| 9. WHAT ARE THE BASIC TYPES OF COMMUNICATION THAT AN EAI SOLUTION SHOULD SUPPORT? .....  | 5  |
| 10. WHAT ARE THE ADVANTAGES AND DISADVANTAGES OF AN EAI SOLUTION? .....  | 5  |
| 11. WHAT ARE THE PARAMETERS ON WHICH YOU SHOULD EVALUATE AN EAI OFFERING? .....  | 7  |
| 12. WHAT IS THE ROLE OF WEB SERVICES IN EAI ? .....  | 9  |
| 13. WHAT ARE THE SALIENT DIFFERENCES BETWEEN TRADITIONAL EAI SOLUTIONS AND WEB SERVICES? .....   | 10 |
| 14. WHAT IS SERVICE ORIENTED ARCHITECTURE (SOA)? .....   | 10 |
| 15. WHAT IS AN ENTERPRISE SERVICE BUS (ESB)? .....   | 11 |
| 16. WHAT IS THE DIFFERENCE BETWEEN AN ESB AND THE BUS BASED IMPLEMENTATIONS THAT HAVE<br>BEEN IN THE MARKET FOR QUITE A LONG TIME? ..... | 13 |
| 17. WHAT IS BUSINESS PROCESS MANAGEMENT (BPM)? .....   | 14 |
| 18. BPM VS SOA .....   | 14 |
| 19. WHAT IS BUSINESS PROCESS EXECUTION LANGUAGE (BPEL)? .....  | 15 |
| 20. REFERENCES .....   | 15 |

## 1. Introduction

Enterprise Application Integration (EAI) is a vast topic. Ideally, a complete book like 'Enterprise Integration Solutions' by Devin Spackman and Mark Speaker or 'Enterprise Integration Patterns' by Gregor Hohpe and Bobby Woolf would do justice to a vast subject such as EAI.

This article attempts to cover the broad areas that a technical or managerial professional needs to understand to kick-start the process of learning EAI .



## 2. What is Enterprise Application Integration (EAI)?

As the need to meet increasing customer and business partner expectations for real-time information continued to rise, companies were forced to link their disparate systems to improve productivity, efficiency, and, ultimately, customer satisfaction. The need for IT systems to communicate within an organization led to the evolution of enterprise application integration (EAI).

EAI is the process of creating an integrated infrastructure for linking disparate systems, applications, and data sources across the corporate enterprise. The very origin of EAI solutions can be linked to the need for providing a full duplex, bi-directional solution to share seamlessly and exchange data between ERP, CRM, SCM, databases, data warehouses, and other important internal systems within the company.

EAI is not an out-of-the-box solution, but rather an ongoing process of creating a flexible, standardized enterprise infrastructure that allows new IT based applications and business processes to be easily and efficiently deployed. The new infrastructure allows applications throughout an enterprise to seamlessly communicate with one another in a real-time manner.

## 3. Why EAI?

EAI can be used for different purposes:

- Data (information) integration: Ensuring that information in multiple systems is kept consistent. This is also known as EII (Enterprise Information Integration)
- Process integration: Linking business processes across applications
- Vendor independence: Extracting business policies or rules from applications and implementing them in the EAI system, so that even if one of the business

applications is replaced with a different vendor's application, the business rules do not have to be re-implemented.

- Common facade: An EAI system could front-end a cluster of applications, providing a single consistent access interface to these applications and shielding users from having to learn to interact with different applications.

Source : [http://en.wikipedia.org/wiki/Enterprise\\_application\\_integration](http://en.wikipedia.org/wiki/Enterprise_application_integration)

## 4. What are the various types of EAI solutions?

EAI solutions can take on many forms and exist at many levels. The appropriate level of EAI can be dependent on many factors including company size, company industry, integration/project complexity, and budget.

There are five types of middleware solutions to EAI:

### 1. User Interface Integration : Refacing

Refacing is the process of replacing the terminal screens of legacy systems and the graphical interfaces of PCs with one standardized interface, typically browser-based. Generally, the functionality of terminal screens applications can be mapped on a one-to-one basis with a browser-based graphical user interface. The new presentation layer is integrated with the existing business logic of the legacy systems or packaged applications such as ERP, CRM and SCM.

Enterprise business portals may also be considered a sophisticated refacing solution. A business portal consolidates the presentation of multiple applications into one customizable browser-based interface. The portal framework acts as a middleware solution in this type of EAI.

### 2. Data Integration

Data integration occurs at the database and data source level within an organization. The integration is achieved by migrating data from one data source to another. Data integration is the most prevalent form of EAI in existence today. One of the biggest problems with data integration, however, is that the business logic usually exists only within the primary system, limiting real-time transactional capabilities.

There are a lot of data replication and middleware tools to facilitate data transfer between data sources in both real-time and batch modes. Some data integration methods include:

- Batch Transfer
- Data Union

- Data Replication
- Extract, Transform, and Load (ETL) Solution

For non real-time solutions, the ETL solution, which is based on an ETL engine extracting, transforming, cleansing and loading data from various applications to data warehouses and/or data marts, has now become the preferred way for companies to achieve data integration.

### 3. Data Federation

For real-time solutions, data across multiple systems is queried and used without the physical movement of source data. This has the advantage of delivering up-to-date data that is consistent and accurate without unnecessary replication and movement of data. It reduces development requirements, maintenance time and storage needs and is a cost-effective solution.

### 4. Business Process Integration

While data integration has proved a popular form of EAI, it can present problems from a security, data integrity, and business process perspective. The vast majority of data within an organization is accessed and maintained through business logic. The business logic applies and enforces the required business rules, processes and security for the underlying data.

Business process integration occurs at the business process level, which spans multiple applications. It is often characterized by the use of advanced middleware such as message brokers, which standardize and control the flow of information through a bus or hub-and-spoke framework.

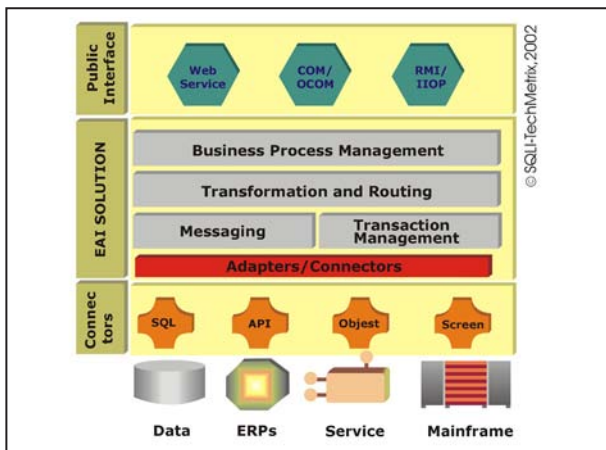
### 5. Method or Function Integration

Function or method integration involves the direct and rigid application-to-application (A2A) integration of cross-platform applications over a network. It can range from custom code (COBOL, C++, Java), to application programming interfaces (APIs), to remote procedure calls (RPCs), to distributed middleware such as TP monitors, distributed objects, common object request broker architecture (CORBA), Java remote method invocation (RMI), message oriented middleware (MOM), and Web Services.

Function or method oriented integration is primarily synchronous in nature, that is, it's based on request/reply interactions between the client (requesting program) and the server (responding program).

## 5. What are the basic components required to achieve EAI?

Currently, there is a lot of variation of thought on what constitutes the best infrastructure, component model, and standard structure for EAI. There seems to be consensus that four things are essential for a modern enterprise application architecture. Refer the diagram below:



Source : <http://www.ebizq.net/topics/eai/features/1555.html?pp=1>

1. There needs to be a centralized broker that handles security, access, and communication. This can be accomplished through integration servers (like the School Interoperability Framework (SIF) Zone Integration Servers) or through similar software like the Enterprise service bus (ESB) model which acts as a services manager.
2. The use of an independent data model based on a standard data structure. It appears that XML and the use of XML style sheets has become the de facto and in some cases de jure standard for applications that can live with the performance penalty that is incurred on use of XML.
3. A connector, or agent, model where each vendor, application, or interface can build a single component that can speak natively to that application and communicate with the centralized broker
4. A system model that defines the APIs, data flow and rules of engagement to the system such that components can be built to interface with it in a standardized way.

Although other approaches like connecting at the

database or user-interface level have been explored, they have not been found to scale or be able to adjust. Individual applications can publish messages to the centralized broker and subscribe to receive certain messages from that broker. Each application only requires one connection to the broker. This central control approach can be extremely scalable and highly evolvable.

EAI is related to middleware technologies such as message-oriented middleware (MOM), and data representation technologies such as XML. Other EAI technologies involve using web services as part of service-oriented architecture as a means of integration. EAI tends to be data centric. In the near future, it will come to include content integration and business processes

## 6. What technologies are used in implementing each of the components in the EAI Solution?

Multiple technologies are used in implementing each of the components of the EAI system:

**Bus/hub:** This is usually implemented by enhancing standard middleware products (application server, message bus, JMS) or implemented as a stand-alone program (i.e., does not use any middleware), acting as its own middleware.

**Adapters:** The bus/hub connects to applications through a set of adapters (also referred to as connectors). These are programs that know how to interact with an underlying business application. The adapter performs two-way communication, performing requests from the hub against the application, and notifying the hub when an event of interest occurs in the application (a new record inserted, a transaction completed, etc.). Adapters can be specific to an application (e.g., built against the application vendor's client libraries) or specific to a class of applications (e.g., can interact with any application through a standard communication protocol, such as SOAP or SMTP). The adapter could reside in the same process space as the bus/hub or execute in a remote location and interact with the hub/bus through industry standard protocols such as message queues, web services, or even use a proprietary protocol. In the Java world, standards such as JCA allow adapters to be created in a vendor-neutral manner.

**Data format Transformers:** To avoid every adapter having to convert data to/from every other applications' formats, EAI systems usually stipulate an application-independent (or common) data format. The EAI system usually provides a data transformation service as well to help convert between application-specific and

common formats. This is done in two steps: the adapter converts information from the application's format to the bus's common format. Then, semantic transformations are applied on this (converting zip codes to city names, splitting/merging objects from one application into objects in the other applications, and so on).

**Integration modules:** An EAI system could be participating in multiple concurrent integration operations at any given time, each type of integration being processed by a different integration module. Integration modules subscribe to events of specific types and process notifications that they receive when these events occur. These modules could be implemented in different ways: on Java-based EAI systems, these could be web applications or EJBs; or even POJOs that conform to the EAI system's specifications.

**Support for transactions:** When used for process integration, the EAI system also provides transactional consistency across applications by executing all integration operations across all applications in a single overarching distributed transaction (using two-phase commit protocols or compensating transactions).

Source : [http://en.wikipedia.org/wiki/Enterprise\\_application\\_integration](http://en.wikipedia.org/wiki/Enterprise_application_integration)

## 7. What are the various integration patterns that EAI solutions implement?

There are two patterns that EAI systems implement:

**Mediation:** Here, the EAI system acts as the go-between or broker between multiple applications. Whenever an interesting event occurs in an application (e.g., new information created, new transaction completed, etc.) an integration module in the EAI system is notified. The module then propagates the changes to other relevant applications.

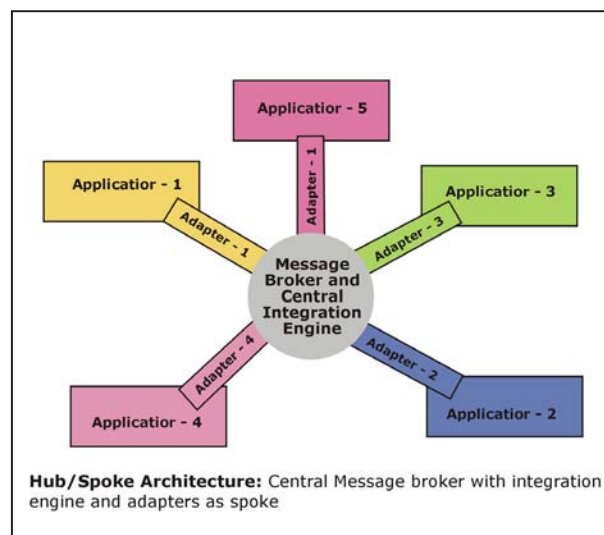
**Federation:** In this case, the EAI system acts as the overarching facade across multiple applications. All accesses from the 'outside world' to any of the applications are front-ended by the EAI system. The EAI system is configured to expose only the relevant information and interfaces of the underlying applications to the outside world, and performs all interactions with the underlying applications on behalf of the requester.

Both patterns are often used concurrently. The same EAI system could be keeping multiple applications in sync (mediation), while servicing requests from external users against these applications (federation).

Source : [http://en.wikipedia.org/wiki/Enterprise\\_application\\_integration](http://en.wikipedia.org/wiki/Enterprise_application_integration)

## 8. What are the two basic architectures used to implement EAI?

### Hub/Spoke



Hub/Spoke architecture uses a centralized broker (Hub) and adapters (Spoke) which connect applications to Hub.

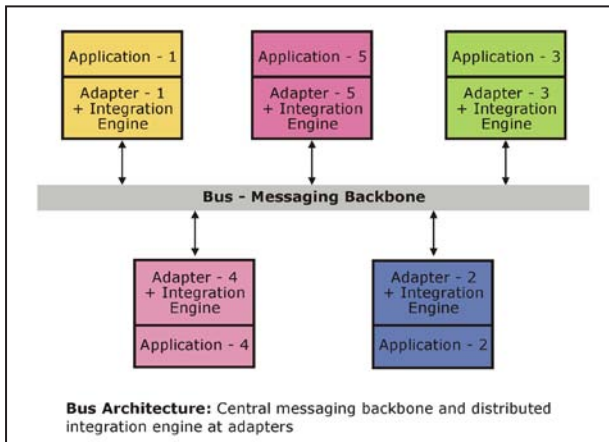
Spoke connect to application and convert application data format to a format which Hub understands and vice versa.

Hub, on the other hand, brokers all messages and takes care of content transformation/translation of the incoming message into a format the destination system understands and routing the message.

Adapters take data from source application and publish messages to the message broker, which, in turn, does transformation/translation/routing and passes messages to subscribing adapter which sends it to destination application(s).

Having a single Hub makes system with this architecture easy to manage but scalability takes a hit. At some point of time as number of messages increase, scalability gets dependent on hardware. Having a bigger box to scale application has never been an ideal solution. In order to overcome this limitation, most vendors have incorporated the concept of federated hub and spoke architecture in which multiple hubs can be present, each hub would have local metadata and rules as well as global metadata. Changes to global rules and metadata are automatically propagated to other hubs. Federated hub spoke architecture alleviates scalability issue while central management of multiple hubs makes this architecture easy to manage and brings down support cost.

**BUS :**



Source : <http://hosteddocs.ittoolbox.com/Enterprise%20Integration%20-%20SOA%20vs%20EAI%20vs%20ESB.pdf>

Service requesters and providers interact by exchanging messages. The bus, acting as a logical intermediary in the interactions, provides loosely coupled interconnectivity between the requester of a function and the provider of that function. Its role as a logical intermediary allows the bus to intercept messages and process them as they flow between a service requester and service provider. This processing is called mediation.

Applications would publish messages to bus using adapters.

These messages would flow to subscribing applications using message bus. Subscribing applications will have adapters which would take message from bus and transform the message into a format required for the application.

Service-oriented solutions include multiple cross-cutting aspects such as security, management, logging, and auditing. The bus can implement or enforce cross-cutting aspects on behalf of service requesters and providers, removing such aspects from the concern of therequesters and providers.

Some aspects of an interaction can be abstracted away from the participants. Consider auditing, for example: If a solution requires auditing, the bus can add it to interactions with no impact on the participants. Similarly, the bus can add to or enhance service-level agreements using its virtualization capabilities to retry failed interactions, or in more sophisticated situations, the bus can match requester requirements with provider capabilities. There are limits, however, since some aspects of an interaction cannot be abstracted. For example, the bus cannot provide a reliable end-to-end interaction if the bus cannot interact reliably with one of the participants.

Key difference between hub/spoke and bus topology is that for the bus architecture, the integration engine that performs message transformation and routing is distributed in the application adapters and bus architecture requires an application adapter to run on the same platform as the original applications.

Since adapters have integration engine and run on same platform on which source and target applications run, this scales much better and is complex to maintain compared to hub/spoke topology.

### 9. What are the basic types of communication that an EAI solution should support?

**Asynchronous:** Provided by products such as IBM’s MQ Series, used in situations where the communicating applications don’t need to be active at the same time. This allows applications to be “loosely coupled in time.” An example of this would be a terminal application feeding a batch process. JMS has support for both Asynchronous and blocking reads.

**Synchronous:** Provided by OMG’s CORBA and Microsoft’s COM, used where the business logic requires that the applications communicate in real-time. The requesting transaction waits until it receives the result set from the other application.

### 10. What are the advantages and disadvantages of an EAI solution?

**Advantages :**

- Real-time information access among systems
- Streamlines business processes and helps raise organizational efficiency.
- Maintains information integrity across multiple systems
- One of the main advantage of EAI is we are reusing the existing applications/systems. The investment the companies have made is not wasted.

**Disadvantages :**

- High development costs, especially for small and mid-sized businesses (SMBs).
- EAI implementations are very time consuming, and need a lot of resources.
- Require a fair amount of upfront design, which many managers are not able to envision or not willing to invest in. Most EAI projects usually start off as point-to-point efforts, very soon becoming unmanageable as the number of applications increase.

- For EAI, we need to have complete knowledge of each and every application, interfaces, data model, user model, role model.
- Integration is tricky for applications which have disparate user / role models, data models.

## 11. What are the parameters on which you should evaluate an EAI Offering?

Plug and Play business application system integration is the goal of investment in EAI technology. Buyers have a number of applications from various suppliers and a number of external trading partners with whom they want to exchange business transactions. EAI suppliers promote products that connect to a diverse set of applications (Adapters) transform data format, structure and content (Transformation Capabilities), move data among applications predictably and with assured content integrity (Transport), and create new transaction interconnections with assured process integrity (Workflow).

With a full complement of Adapters, the appropriate set of Transformation Capabilities, versatile communications Transport and flexible Workflow capabilities, heterogeneous applications, platforms and network

configurations cease to be impediments. EAI tools promise seamless and flexible interconnection with low overhead.

### Apparent capabilities vs. Vital capabilities

TEC (Technology Evaluation Centers - <http://www.technologyevaluation.com>) has examined a number of recent EAI acquisitions to refine its Selection Model and Knowledge-based Procurement Process. Of particular interest is the difference that they noticed when they compared selection criteria to success criteria. It is clear that a number of selections did not sufficiently address long-term cost of ownership and operation capabilities. Several of those capabilities are described in the following table.

The column titled Apparent Capability lists a number of EAI features and functions that customers included in their Request for Proposal (RFP). The Discovered Requirement provides examples of additional capabilities that at least one customer discovered should have been included as a requirement in its RFP. The third column, entitled Implications, captures important considerations for buyers based on actual discovery during implementation and use.

| Category     | Apparent Capability   | Discovered Requirement  | Implications   |
|--------------|---|---|--|
| Performance  | Product supports multiple instances operating on multiple servers and demonstrates linear performance increase characteristics. | Product supports linearly scalable performance and Quality of Service controls to manage available resources.   | Although the selected product did demonstrate performance improvements commensurate with increased resource availability, it was necessary to reconfigure transport flows to avoid "resource hogging" by a relatively small number of large and/or complex messages. |
| Availability | Product has features to assure 24 X 7 operational status.   | Product has features to assure 24 X 7 operational status and full environment backup without operational downtime.                                      | During failure recovery testing, it was found that all message services must be quiesced (shut down) in order to assure recovery from a backup fileset.  |
| Availability | Product supports transport server direction through named services and dynamic location (like Domain Name Services).            | Product supports server access through named services and supports access through multiple paths with assigned selection priority.                      | Although the product provided dynamic load balancing, it was not capable of employing all available network paths between servers.   |
| Management   | Product provides multi-tiered promotion of workflow configurations (e.g., Development to Quality Assurance to Productive).      | Multi-tiered workflow configuration promotion supports coexistence of multiple versions of the product operating simultaneously on different platforms. | A Productive version was upgraded to fix a bug and promotion transport services on another server refused to interact with the "incompatible" version.   |
| Management   | Product employs a repository of all metadata and an intuitive graphical user interface for configuration programming.           | Product provides a graphical user interface for configuration programming and to assist with message flow tracing during debug actions.                 | It was discovered that ease of configuration was not complemented with a similar facility for tracing and (single-stepping) messages through the system.   |

| Category       | Apparent Capability  | Discovered Requirement   | Implications  |
|----------------|--|--|---|
| Installation   | Supplier installs and verifies the operational status of the product.                          | Supplier also assures failure recovery and provides backup, recovery and upgrade procedures.   | Failover, backup and recovery systems were left to the customer (who did not have a full understanding) as was all process documentation. Lack of full understanding was discovered at an inopportune time. |
| Installation   | Supplier provides product training and implementation assistance.                              | Supplier also provides proficiency assessments with training as a follow-up service.   | Customer found that poor programming practices had become the norm and had to recode a number of message configurations to obtain desired levels of reliability and performance.                            |
| Adapters       | Product provides “native” connectivity to the applications and data storage facilities listed. | Product provides non-invasive connectivity for execution of the transactions described.  | Most Adapters were found to be “starter kits” with a limited number of transactions and an even more limited set transaction features.  |
| Adapters       | Supplier provides documentation and training for the development of custom adapters.           | Custom adapters can be incorporated into the repository and they can be readily validated and carried forward through system upgrades.                       | Custom adapters required “special handling” for use and their functionality was unknown to the metadata repository.   |
| Security       | Product provides secured access to configuration tools.  | Multiple levels of tool access are provided to assure configuration integrity, promotion control and administrative access without configuration capability. | Work could not be distributed without risking system integrity.   |
| Security       | Product is compatible with “Firewall Product” assuring network transparency.                   | List all firewalls that the product has been operated through with references to parties who configured them.  | An internal firewall was improperly configured to allow message flows resulting in a security risk.   |
| Security       | Product has the capability to employ user credentials and certificates.                        | User credentials can be associated with users who are authenticated only once or for each transaction.   | Multiple authentication models were found to be necessary only after implementation.  |
| Business Rules | Product allows business rules to be configured into the repository and reused.                 | Specific applications of business rules can be configured into “macros or super-processes” that span multiple business rules and process flows.              | Opportunities for improved reuse were discovered during system design and configuration.  |
| Workflow       | Product provides workflow capabilities beyond message routing.                                 | Product workflow features interoperate with the products listed.   | An application could have benefited from integration with Oracle Corporation’s Workflow.  |
| Transport      | Product supports publish/subscribe and request/reply communications models.                    | Product interoperates transparently with the Message Oriented Middleware (MOM) products listed.  | Newly acquired businesses already owned and operated MOM from a different supplier and wanted to maintain integrated messaging.   |

Source : [http://www.technologyevaluation.com/Research/ResearchHighlights/DataWarehousing/2000/02/research\\_notes/TN\\_DW\\_MFR\\_02\\_00\\_12.asp](http://www.technologyevaluation.com/Research/ResearchHighlights/DataWarehousing/2000/02/research_notes/TN_DW_MFR_02_00_12.asp)

## 12. What is the role of Web Services in EAI ?

Web Services are not EAI in and of themselves. Rather, Web Services are just another technology that enables EAI, and it can significantly change the traditional point-to-point approach.

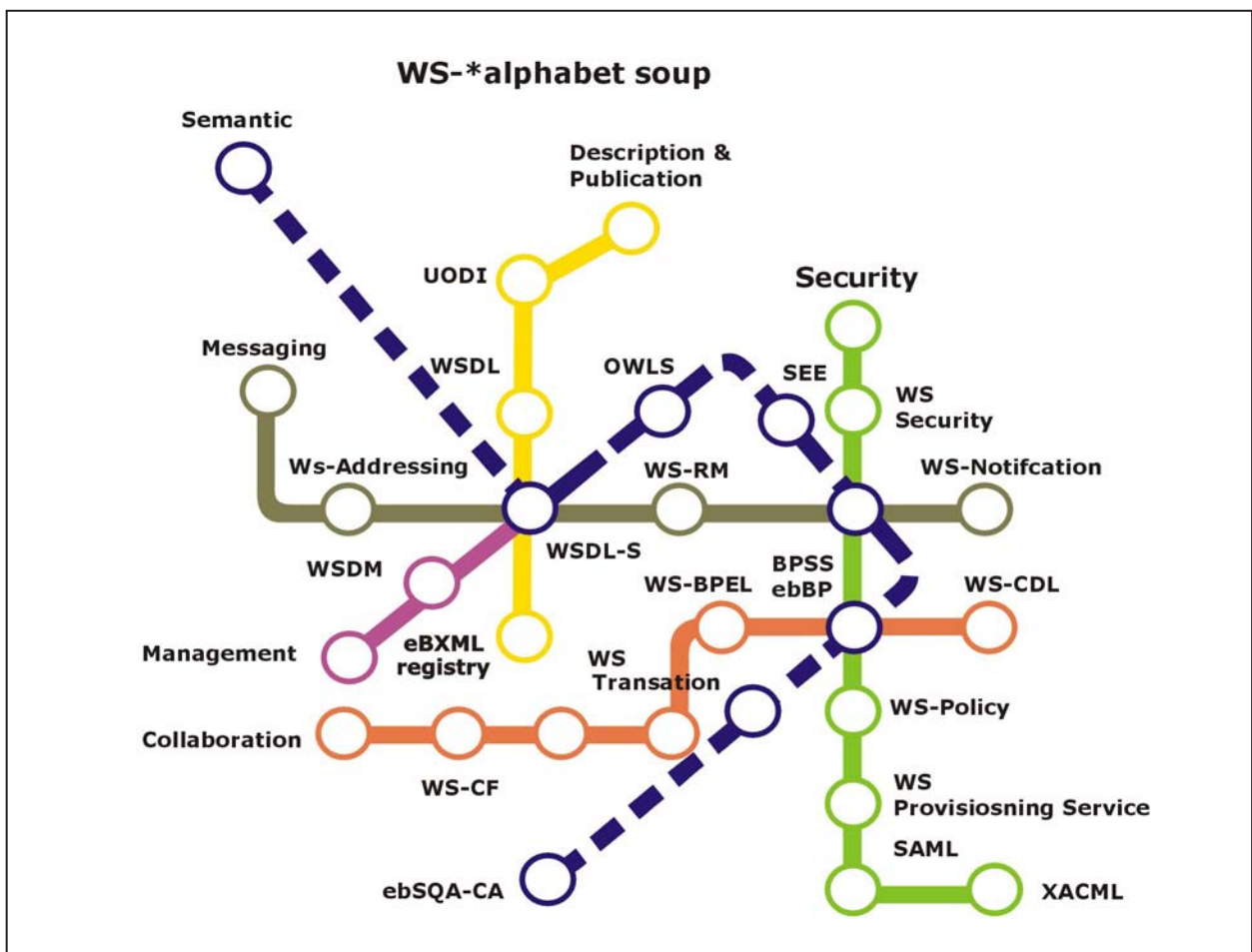
Using Web Services that loosely integrate applications, a company achieves just a subsection of EAI. EAI, on the other hand, takes a complete holistic approach of tightly integrating and connecting all applications and systems that support a company's business. EAI takes years of continued commitment and effort from different business and technical units within the company, high investment, and substantial resources.

Web Services, in their current form of loosely bound collections of services, are more of an ad hoc solution that can be developed quickly and easily, published, discovered, and bound dynamically. In this generation of Web Services, it is possible to achieve only function level

integration between applications. They are not transactional in nature and provide basic "request/response" functionality. The next generation of Web Services, however, will be functionally and technologically advanced, offering user interface encapsulation and security. They will be able to package an application and embed it into another application.

The current EAI solutions that predominately focus on integrating applications will have to be changed significantly, as packaged applications in the future will expose their functions as services using technologies such as XML, SOAP, and UDDI. Thus, the EAI solutions will have to provide a broad support for service integration rather than application integration.

The second-generation Web services standards and technologies (generally referred to as WS-\*) provide the foundation for the future Web services-based service-oriented enterprise. All major vendors are currently adopting portions of this second-generation platform, albeit at different levels of maturity.



Source : <http://www.soamag.com/it/0906-1.asp>

While not complete, the above figure provides us with a sense of the technological mapping that will be required to establish an SOA roadmap for the typical enterprise.

### 13. What are the salient differences between traditional EAI Solutions and Web Services?

A few essential differences between traditional EAI solutions and Web Services are, as follows:

(Note: Some of these differences take into account the future enhancements proposed in Web Services)

**Simple:** There is no doubt that Web Services are much simpler to design, develop, maintain, and use as compared to a typical EAI solution which may involve distributed technology such as DCOM and CORBA. Once the framework of developing and using Web Services is ready, it will be relatively easy to automate new business processes spanning across multiple applications.

**Open Standards:** Unlike proprietary EAI solutions, Web Services are based on open standards such as UDDI, SOAP, HTTP and this is probably the single most important factor that would lead to the wide adoption of Web Services. The fact that they are built on existing and ubiquitous protocols eliminates the need for companies to invest in supporting new network protocols.

**Flexible:** Since EAI solutions may require point-to-point integration, changes made at one end have to be propagated to the other end, making them very rigid and time consuming in nature. Web Services based integration is quite flexible, as it is built on loose coupling between the application publishing the services and the application using those services.

**Cheap:** EAI solutions, such as message brokers, are very expensive to implement. Web Services, in the future, may accomplish many of the same goals - cheaper and faster.

**Scope:** EAI solutions, such as message brokers, integrate applications treating them as single entities, whereas Web Services allow companies to break down big applications into small independent logical units and build wrappers around them. For example, a company can write wrappers for different business components of an ERP application such as order management - purchase order acceptance, status of order, order confirmation, accounts receivable, and accounts payable.

**Efficient:** As mentioned in the previous point, Web Services allow applications to be broken down into smaller logical components, which makes the integration of applications easier as it is done on a granular basis. This makes Web Services solutions for EAI much more efficient than traditional EAI solutions.

**Dynamic:** Web Services provide a dynamic approach to integration by offering dynamic interfaces, whereas traditional EAI solutions are pretty much static in nature.

### 14. What is Service Oriented Architecture (SOA)?

Competition forces the enterprise to adopt new business practices (like outsourcing, M&A, etc.), while regulation forces the enterprise to inject new controls into its current business practices. In both cases, the enterprise answer is openness. This includes the adoption of standardized interfaces simply to transfer the burden of integration out of traditional enterprise boundaries.



Figure : Key enterprise drivers leading toward SOA adoption.

Source : <http://www.soamag.com/11/0906-1.asp>

With openness in mind, SOA is seen as the right answer for the enterprise. Service oriented architecture is approach to have software resources in an enterprise available and discoverable on network as well defined services. Each service would achieve a predefined business objective and perform discrete units of work. The services are independent and do not depend on the context or state of the other services. They work within distributed systems architecture.

Although current SOA implementations strongly leverage Web services technology, SOA is at first a concept and an orientation. One of the greatest challenges to achieving a successful SOA implementation is ensuring that the high-level business objectives (continuity of business, security exposure, business requirements, liabilities, etc.) are addressed in the implementation of each individually delivered service. Attaining these strategic goals requires, on a fundamental level, that the concept of "service" be first defined outside of technology and specific implementation concerns.

SOA represents an evolution in distributed processing approaches that distinguishes itself by emphasizing a strict separation of responsibilities and concerns using standard contracts (external interfaces). While the underlying principles are not tied to any specific technology, SOA is commonly implemented with Web services because this technology framework inherently support some of the major principles (like the separation of concerns and the definition of standard contracts) of service-orientation.

It is often recommended to start an enterprise SOA initiative with a small project, but it must be understood that SOA itself is not a project but rather a journey. The enterprise must understand that the impact of SOA is huge and goes far beyond traditional IT boundaries and simple point-to-point integration. When planning this journey, especially from an enterprise perspective, it is helpful to have an idea of your final destination (this long-term vision will also make your CIO and CFO feel better).

Source : <http://www.soamag.com/It/0906-1.asp>

## 15. What is an Enterprise Service Bus (ESB)?

An ESB is an open standards, message-based, distributed, integration solution that provides routing, invocation, and mediation services to facilitate the interactions of disparate distributed information technology resources (applications, services, information, platforms) in a reliable manner.

The key terms in our ESB definition :

- **Open Standards:** Open standards refers to both the ESB solution components (runtime container, messaging infrastructure, integration services, design-time notations) and the mechanisms for integrated resources to participate (attach, request, respond) on the bus.
- **Message-Based:** The communication mechanism of an ESB is messaging, using standard message notation, protocols, and transports.

- **Distributed:** The ESB runtime environment can be distributed across a networked environment for the purposes of quality of service, quality of protection, and economics.
- **Routing, Invocation, and Mediation:** Routing, invocation, and mediation are the basic functions of the ESB. Routing includes addressability and content-based routing. Invocation refers to the ability to make requests and receive responses from integration services and integrated resources. Mediation refers to all translations and transformations between disparate resources including security, protocol, message notation/format and message payload (data/semantics).

- **Facilitate:** The ESB must coordinate the interactions of the various resources and provide transactional support.

- **Reliable:** The ESB must guarantee message delivery.

Some SOA deployments solely rely on the use of an ESB, while others just utilize an ESB to offload non-core processing tasks (for example by pushing audit and log information onto the bus for delayed processing).

There seems to be general agreement on what an ESB should do. The table below lists the basic and optional features and functions of an ESB:

| Enterprise Service Bus Features and Functions |  |       |          |
|---|--|-------|----------|
| Category                                      | Feature/Function   | Basic | Optional |
| Routing                                       | Addressability   | Y     |          |
|   | Content-based routing  | Y     |          |
|   | Transport—synchronous and asynchronous (JMS, WS-Rel <sup>m</sup> , HTTP) | Y     |          |
| Invocation                                    | WS-I (SOAP, UDDI, WSDL)  | ?     | ?        |
|   | JMS, JCA   | ?     | ?        |
| Mediation                                     | Data transformation and translation (XSD, DTD, XSLT, XPath)              | Y     |          |
|   | Protocol translation (SOAP, JMS, WS-Rel <sup>m</sup> , JCA, SMTP)        | ?     | ?        |
|   | Security model translation   | Y     |          |
| Process Orchestration                         | Business process support (BPEL)  |       | Y        |
| Complex Event Processing                      | Event interpretation, event correlation, event pattern matching          |       | Y        |

| Enterprise Service Bus Features and Functions (continued) |  |       |          |
|---|--|-------|----------|
| Category  | Feature/Function   | Basic | Optional |
| Adapters  | Adapters for major application infrastructure platforms (J2EE, .Net), application packages, DBMS, FTP, EDI |       | Y        |
| Integration Tooling                                       | Graphical design-time tooling  | Y     |          |
|   | Deployment tools (Ant)   | Y     |          |
|   | Testing tools  |       | Y        |
| Change Management   | Hot deployment   | Y     |          |
|   | Multiple versions (production)   | Y     |          |
|   | Service lifecycle management (development, testing, QA, production)  | Y     |          |
| Quality of Service  | Transaction control and compensation   | Y     |          |
|   | Failover at service and service container  | Y     |          |
|   | Distributed network topology with flexible deployment for performance and scalability                      | Y     |          |
| Quality of Protection                                     | Encrypt/decrypt message content  | Y     |          |
|   | Secure endpoints (authentication, access control)  | Y     |          |
|   | Secure persistence mechanisms  | Y     |          |
|   | Distributed architecture for DMZ deployment  | Y     |          |
| Management  | Standards-based monitoring alerts for infrastructure and integration scenario execution                    | Y     |          |
|   | Audit and logging  | Y     |          |
|   | Unified management console for monitoring and administration of infrastructure and integration scenario    | Y     |          |
|   | Usage metering   |       | Y        |
|   | Standards-based monitoring alerts for business process execution (BAM)                                     |       | Y        |

© 2005 Patrivia Sebold Group Lic.

Table A: This table lists the features and functions of an enterprise service bus. The question marks indicate features about which vendors disagree as to whether they are basic or optional.

Source : [http://www.ebizg.net/hot\\_topics/esb/features/6132.html](http://www.ebizg.net/hot_topics/esb/features/6132.html)

When you “open the box” of an ESB, you would find integration tooling, a management console, service containers (ESB peer), and integration services.

There are however, variations in what else you might find in your ESB. The variations relate to messaging infrastructure implementation, protocol support,

adapters, and exposure of ESB services, as follows:

- Messaging Infrastructure: An ESB may be tied to its own messaging infrastructure (MOM), or it may provide adapters (JMS or WS-Rel\*) to run any of the more popular messaging infrastructures (WebSphere MQ, MSMQ, Tibco Rendezvous).

- **Protocol Support:** An ESB may be single protocol or multiprotocol. This refers to “how you get on the bus.” For example, most single-protocol ESBs require a WS-SOAP interface, but if you send a “plain” XML message using JMS, an adapter translates the incoming message to SOAP to participate on the bus. In a multiprotocol ESB, varieties of protocols (WS-SOAP, JMS, JCA, etc.) natively interact with the bus, without employing an adapter.
- **Adapters:** As with any integration solution, the collection of adapters (protocol bridges) supplied by the ESB vendor varies. The adapters will also depend on the protocol support, as mentioned above.
- **Exposure of ESB Services:** Some ESBs expose their underlying services, for integration or management, as services for consumption outside of the ESB solution set. This facilitates remote management and reuse of integration services.

Regardless of how the ESB performs out-of-the-box, there's plenty of tuning to be done. Take a look at JVM (Java Virtual Machine) memory tweaks, the number of connections allowed for HTTP connections and the code implementing a service; all are factors in the overall performance of any given scenario. In addition, the ability to perform in-memory calls between orchestrated services is imperative, especially when the orchestration involves SOAP over HTTP. Removing the overhead of a TCP session when services reside on the same machine is a huge performance benefit, and you should inquire about this capability to ensure it's available.

Another impediment to performance is the execution of complex style-sheet transformations or the use of encryption/decryption as per the WS-I Basic Security Profile. Being able to perform transformations quickly and efficiently will have a direct impact on the overall performance of any ESB. The choice of XML parsers, too, will have an impact, as well as imposing possible limitations on the size of XML documents that can ride the bus.

Source : [http://www.ebizq.net/hot\\_topics/esb/features/6132.html](http://www.ebizq.net/hot_topics/esb/features/6132.html)  
[http://www.ebizq.net/hot\\_topics/esb/features/6117.html?page=3](http://www.ebizq.net/hot_topics/esb/features/6117.html?page=3)

## 16. What is the difference between an ESB and the Bus based implementations that have been in the market for quite a long time?

Current EAI landscape is seeing many vendors who offer enterprise service bus and claim it to be a brand new concept. Actually there is not much difference between ESB and proprietary buses except for a few subtle ones.

Main difference between ESB and proprietary bus implementation is the cost factor which is significantly low for ESB. Reason for this cost difference is two fold. Firstly, proprietary bus offers lot of built-in functionalities as a suite of product which need to be developed for ESB implementations based on business requirement; Secondly, most proprietary buses use some proprietary formats to enhance the performance and that increases the cost. ESB on the other hand is usually standard based, so it is a tradeoff between performance and cost between proprietary bus and ESB. Main advantage of ESB is that it costs much less than hub/spoke or bus based product suites and it is standard based.

Following table gives a quick comparison of hub/spoke, bus based product suites and ESB. Also all these three architectures can be service oriented depending on implementation which is reflected in this comparison.

SOA brings cost-effective, reusable and low lead time solutions to an organization but EAI and SOA are both going to coexist. Web services alone as SOA cannot handle the complex, secure and SLA based applications of an enterprise currently and unless we see a technological breakthrough, it is going to remain that way.

Enterprise service bus would enable low cost integration and would be used by companies with limited IT resources and environments that involve a handful of systems and moderate transaction volumes.

Packaged EAI solutions would have SOA as basic tenet and would continue to be used for large scale integration by companies having huge number of diverse system and high transaction volumes. Next generation EAI solutions would use more and more of SOA to provide reliable, secure, low cost and flexible solutions.

So, in short:

1. SOA brings cost-effective, reusable and low lead time solutions to an organization but EAI and SOA are both going to coexist.
2. SOA is more than web services, in fact web services alone cannot handle the complex, secure and SLA based applications of an enterprise.
3. Enterprise service bus would enable low cost integration and would be used by companies with limited IT resources.
4. Packaged EAI solutions in future would have SOA as basic tenet and would continue to be the prime choice for large scale integration.

| Evaluation Parameter | Hub Architecture   | Bus Architecture  |   |
|----------------------|--|---|---|
|                      |  | Proprietary bus based product suite                                 | ESB   |
| Installation Effort  | Less installation effort compared to solutions with bus architecture                             | Moderate effort   | Moderate effort   |
| Administration       | Easy to maintain and Administrate because of central hub   | Administration may be complex depending upon the integrated systems | Administration may be complex depending upon the integrated systems   |
| Cost                 | High   | High  | Low cost because it does not use proprietary formats to enhance performance; Also it does not provide all the services usually provided by proprietary product suites |
| Scalability          | High if federated architecture is used otherwise limited by the hardware of box used to host Hub | Highly scalable   | Highly scalable   |
| Standards            | Mostly standard based but may use proprietary internal formats                                   | Mostly standard based but may use proprietary internal formats      | Service oriented  |
| SOA                  | Can be implemented as service oriented   | Can be implemented as service oriented                              | Service oriented  |

Source : <http://hosteddocs.ittoolbox.com/Enterprise%20Integration%20-%20SOA%20vs%20EAI%20vs%20ESB.pdf>

## 17. What is Business Process Management?

Business Process Management (BPM) is a management model that allows the organizations to manage their processes as any other assets, and improve and manage them over the period of time. In a medium to large organization scenario, a good business process management system allows business to accommodate day-to-day changes in business processes due to competitive, regulatory or market challenges in business processes without overly relying IT departments. This strikes a fine balance between dynamic business areas that want to avoid every risk and grab every opportunity on their way through agile changes in their way to business but are very often restricted by a very stable and hard to change IT infrastructure.

## 18. BPM vs SOA

- BPM is top-down. It starts with the end-to-end process in mind. SOA is bottom-up. It starts by factoring the world into atoms and molecules of reuse – business services – that can be consumed by any number of processes or applications.
- BPM is business-driven. A “model” created by the business drives the implementation. SOA is IT-driven. Technical architects define the scope of business services based on their own notions of “abstraction”

rather than business need.

- In BPM, success is measured by business metrics and KPIs at the end-to-end process level. In SOA, success is measured by architecture, logical consistency, ease of integration. Performance goals do not extend beyond the individual service level.
- BPM is project-oriented. The goal is solving a real business problem today. SOA is enterprise infrastructure-oriented. The goal is creating a foundation for solving many business problems a few years down the road.
- In BPM, what is reused is the process model, i.e. the “abstract” design of a process fragment. It is inherently transparent – you see its steps and their individual performance parameters. In SOA, what is reused is the service implementation. It is inherently opaque, defined simply by its interface. Its internal steps and performance parameters are invisible, except for an overall SLA for the service as a whole.
- In BPM, a business process is inherently hierarchical, composed of nested and chained orchestrations. In SOA, services are inherently independent. An end-to-end process is more likely to be implemented as a spaghetti-like SCA assembly than a nested hierarchy of BPEL orchestrations.
- In BPM suites based on service orchestration,

process activities are bound to service endpoints. In SOA (supposedly), orchestrated services are supposed to be abstract, with connection and mapping to endpoints mediated by an ESB. The ability to swap out endpoints performing the same abstract service is the effective definition of loose coupling.

Source : <http://www.brsilver.com/wordpress/2006/08/21/bpm-on-soa-what-would-it-look-like-part-1/>

## 19. What is Business Process Execution Language (BPEL)?

BPEL (Business Process Execution Language) for Web services is an XML-based language designed to enable task-sharing for a distributed computing or grid computing environment - even across multiple organizations - using a combination of Web services. Written by developers from BEA Systems, IBM, and Microsoft, BPEL combines and replaces IBM's WebServices Flow Language (WSFL) and Microsoft's XLANG specification. (BPEL is also sometimes identified as BPELWS or BPEL4WS.)

BPEL is an Orchestration language, not a choreography language. The primary difference between orchestration and choreography is scope. A choreography model provides a scope specifically focusing on the view of one participant (e.g. a peer to peer model). Instead, an orchestration model encompasses all parties and their associated interactions giving a global view of the system. The orchestration and the choreography distinctions are based on analogies: orchestration describes central control of behavior as a conductor in an orchestra, while choreography is about distributed control of behavior where individual participants perform processing based on outside events, as in a choreographed dance where dancers react to behaviors of their peers.

In addition to providing facilities to enable sending and receiving messages, the BPEL programming language also supports:

- A property-based message correlation mechanism
- XML and WSDL typed variables
- An extensible language plug-in model to allow writing expressions and queries in multiple languages: BPEL supports XPath 1.0 by default
- Structured-programming constructs including if-then-elseif-else, while, sequence (to enable executing commands in order) and flow (to enable executing commands in parallel)
- A scoping system to allow the encapsulation of logic with local variables, fault-handlers, compensation-handlers and event-handlers

- Serialized scopes to control concurrent access to variables

## 20. References

- [http://en.wikipedia.org/wiki/Enterprise\\_application\\_integration](http://en.wikipedia.org/wiki/Enterprise_application_integration)
- <http://www.webservicesarchitect.com/content/articles/samtani01.asp>
- <http://www.softwaremag.com/L.cfm?Doc=archive/2000feb/EAI.html>
- <http://www.theserverside.com/tt/articles/article.tss?l=ESBParadigm>
- [http://www.technologyevaluation.com/Research/ResearchHighlights/DataWarehousing/2000/02/research\\_notes/TN\\_DW\\_MFR\\_02\\_00\\_12.asp](http://www.technologyevaluation.com/Research/ResearchHighlights/DataWarehousing/2000/02/research_notes/TN_DW_MFR_02_00_12.asp)
- <http://hosteddocs.ittoolbox.com/Enterprise%20Integration%20-%20SOA%20vs%20EAI%20vs%20ESB.pdf>
- [http://en.wikipedia.org/wiki/Business\\_process\\_management](http://en.wikipedia.org/wiki/Business_process_management)
- <http://www.brsilver.com/wordpress/2006/08/21/bpm-on-soa-what-would-it-look-like-part-1/>
- [http://searchsoa.techtarget.com/sDefinition/0,,sid26\\_gci845110,00.html](http://searchsoa.techtarget.com/sDefinition/0,,sid26_gci845110,00.html)
- <http://www.ebizq.net/topics/eai/features/1555.html?&pp=1>
- <http://www.sas.com/technologies/dw/datafederation/index.html>
- [http://www.ebizq.net/hot\\_topics/esb/features/6132.html](http://www.ebizq.net/hot_topics/esb/features/6132.html)
- <http://www.soamag.com/l1/0906-1.asp>

## Contact us

### USA

460 Park Avenue South  
Suite #1101, New York  
NY 10016, USA  
Tel.: +1 212 686 6655  
Fax: +1 212 686 2422

### UK

88 Wood Street  
London EC2V 7RS,, UK  
Tel.: +44 20 85281000  
Fax: +44 20 85281001

### Australia

410 Concord Road  
Rhodes, NSW 2138, AUS  
Tel.: +61 290 221 146,  
Fax: +61 290 221 134

### INDIA

Bagmane Technology Park  
Byrasandra Village,  
C.V. Raman Nagar  
Bangalore 560 093, India  
Tel.: +91 80 4004 0404,  
Fax: +91 80 4004 9999

## About Mphasis

Mphasis is a leading Applications, Infrastructure Technology, and BPO services provider.

The company delivers real improvements in business performance for clients through a combination of technology know-how, domain and process expertise. With currently over 36,000 people, Mphasis services clients in Financial Services, Healthcare, Communications, Media & Entertainment, Transportation & Logistics, Energy & Utilities, Consumer & Retail, and Governments around the world. To know more, visit [www.mphasis.com](http://www.mphasis.com).

Mphasis and the Mphasis logo are registered trademarks of Mphasis Corporation. All other brand or product names are trademarks or registered marks of their respective owners. Mphasis is an equal opportunity employer and values the diversity of its people. Copyright © Mphasis Corporation. All rights reserved.

