Documenting software architecture, Part 3: Develop the architecture overview

Three complementary views lay the foundation

Tilak Mitra
Certified Senior IT Architect
IBM

27 June 2008

In this series, learn why and how you should document software architecture. This article explains how to develop and document the high-level architecture overview for your system or application. The architecture overview, with its three main views, plays a critical role in providing the foundation for your enterprise, application, and systems architecture.

View more content in this series

Introduction

In Part 1 of this series, you learned about the importance of a disciplined approach to documenting software architecture and about mechanisms that can capture the architectural artifacts used in a typical development process. Part 2 focused on the system context, which is the first important architecture artifact, and how to document system context information with diagrams and information flows.

In this article, learn about the architecture overview, which provides a high-level schematic of the various dimensions of the architecture.

The architecture overview provides different views of the developing architecture of a system. Architecture, by definition, helps communicate the developing solution to key stakeholders of the company, though not all views are of interest to all the stakeholders. The CEO might be interested in the enterprise architecture view, while the chief architect may be interested in the IT views (system view and layered view). Meanwhile, the Service-Oriented Architecture (SOA) or application architect might be interested in only the services view. Each view focuses on a specific aspect, and there are guidelines on how to represent and document these. This article discusses the various views of the architecture and gives tips for documenting the views that collectively form the architecture overview of the enterprise or system. Architecture documentation is a key responsibility of the software architect.

Importance of the architecture overview

I have represented the architecture overview as a set of diagrams. Each diagram has a specific focus, which will be discussed in the next section. Documenting the architecture overview is important, because it can:

- Lay a foundation of the system's architecture and act as a guide for the more elaborate functional and operational architecture of the solution.
- Communicate a conceptual understanding of the evolving solution's architecture to the stakeholders.
- Provide a mechanism to evaluate different architecture solutions to a particular problem.
- Offer enterprise, system, or application views of the architecture in a single consolidated artifact.
- Help orient new team members on a project.

Documenting the architecture overview

The architecture overview is best represented when a set of complementary views are documented together. The views represent the various architecture components that comprise the solution. I recommend that you document the following three views:

- The enterprise architecture view identifies components that should be deployed to support the overall IT strategy and let you transform the business strategy and operating model into reality.
- The layered view presents another perspective of the IT system view.
- The IT system view illustrates the various services (process, technical, and functional) that make up the application architecture of the solution. Also groups them in broad functional categories.

The level of detail increases with each view. The IT system view paves the way for the component model and the operational model by identifying all the main functional nodes of the architecture.

Enterprise architecture view

First, a bit about why it's important to capture the enterprise architecture view. Any business or enterprise chooses one of the three operating models where it wants to excel. An *operating model* is made up of operating (business) processes, business structure, management structure, and culture—all of which are synchronized to create superior value. A business can be categorized into three basic business operating models: operational excellence, product leadership, and customer intimacy.

From an IT standpoint, the three business operating models can be mapped to four IT-level operating models:

- Diversification, with low standardization and low integration requirements.
- Coordination, which has low standardization but a high integration focus.
- Replication, with high standardization but a low integration focus.
- Unification, with high standardization and high integration imperatives.

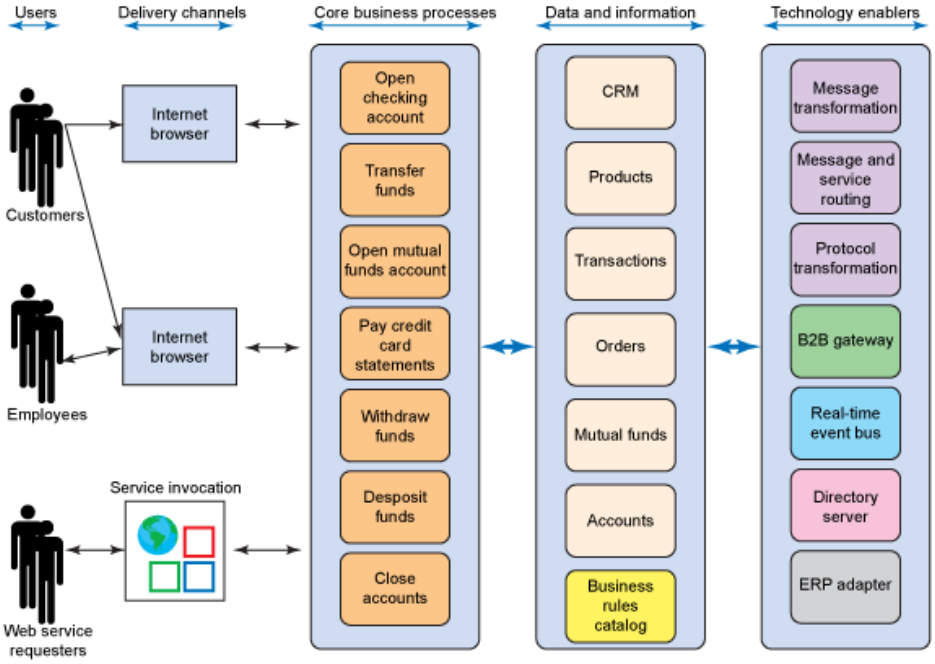
See Resources for more on IT operating models.

Enterprise architecture provides a mechanism to identify and represent, in a single unified view, the business processes, data, technologies, and customer interfaces that take the operating model from vision to reality.

Enterprise architecture is the single diagram that communicates the enterprise core business processes, together with the application and infrastructure building blocks. The diagram is typically an overview; it does not drill down into details of the application, data, technology, or business process architecture. This single view becomes the starting point for further detailed IT architectures. Therefore, enterprise architecture cannot be used interchangeably with IT architecture, which has been a common mistake in the IT world. The enterprise architecture acts as the overarching guideline from which the detailed IT architectures may be defined.

Figure 1 shows a simple diagram of the high-level business processes, data, technology, and customer channels that together constitute the desired foundation for execution.

Figure 1. Enterprise architecture view from a banking example



You should document the key conceptual elements in the enterprise architecture diagram, as shown next for the example in Figure 1.

Users and delivery channels

The enterprise view in Figure 1 allows different audiences to access the banking applications over various delivery channels:

- Customers will access applications over the Internet using a Web browser.
- Employees, including call center personnel or administrators, will access applications over an intranet using a Web browser. They could also access the applications with a virtual private network (VPN).
- External partners can access a functional subset of the banking applications using Web services-based invocations.

Each channel provides the user with access to a subset of the banking applications. The capabilities and the presentation style may differ depending on the user and the channel. For example, employees might have access to more functions than customers, who might have different, more robust, and scalable presentation capabilities.

Core business processes

The enterprise view highlights the core business processes that collectively represent the operating processes that the client focuses on in order to excel in the chosen operating model. The business processes that promote operational excellence for the client are:

- Open checking account.
  The customer can open a checking account in less than 10 minutes. The process can be invoked at the branch office through a teller counter and through a self-service online banking portal.
- Transfer funds.
  This process provides the ability to transfer funds from one account type to another within the bank and to transfer funds between international bank accounts for a transaction fee (the least expensive in the industry and one that takes only three business days).
- Open mutual funds account.
  Customers and employees can open a mutual funds account with the bank and access the bank's most trusted and highest performing funds and analytics. This process also lets the clients link a mutual funds account with a checking account and provides up to 40 free transactions per month.
- Pay credit card settlement.
  Customers have online credit card settlement capabilities. Direct debit from the checking account and overdraft protection facilitate credit card payments.

The other processes should also be described to give a high-level overview of how the process helps the bank.

Data and information

The core conceptual information and data constructs required to realize the core set of business processes are highlighted in the enterprise view. The following data and information elements are fundamental business entities:

- CRM
  The customer relationship management (CRM) system is the entity that manages customer demographic information, number of products subscribed for, and account standings.
- Products
  The product entity represents the products that the bank offers to customers and employees. Examples of products are checking accounts, savings accounts, mutual funds, and credit cards.
- Order
  The order entity represents the orders placed by customers to the bank. Orders can be payments, mutual funds transactions, and funds transfers.
- Business rules catalog

The business rules catalog is a collection of business rules used to realize the various implementations of the business processes. Each business rule uses information elements and enforces certain conditional logic upon them. The rules can be represented in natural language and can be modified by business users. An example of a business rule is:
"If mutual_fund_transaction_number (less than or equal to) 40 then transaction_fee_flag = false."

The rest of the information and data entities should be similarly documented.

Technology enablers

The enterprise view highlights the key set of technology enablers that constitute part of the IT infrastructure required to support implementation of core business processes. The conceptual technology artifacts are:

- Message transformation
  Provides the ability to transform different types of message formats used by clients and access the enterprise business processes in a single standard format used by the core business applications within the enterprise. It also enables transformation of the standard message format into the specific message format that the requesting client expects or supports.
- Message and service routing
  Provides basic and advanced message and service routing capabilities. It also provides the intelligence to find the correct service provider for a given service request and route the service request accordingly.
- Real-time event bus
  Provides basic and advanced features for simple and complex event processing. The bus features facilitate asynchronous processing of funds transfer within and outside of the enterprise in near real-time speed.
- Directory server
  Manages the user profiles required to validate user credentials and determine roles. User profiles are also used to perform authorization functions.
- Business-to-business (B2B) gateway
  Receives requests from third parties through service invocations. The gateway provides a focal point for handling requests that originate from external entities through Web services. The component can be used to mask the internal addresses of Web services and to transform the request to a different protocol.
  For incoming traffic, the gateway is responsible for receiving requests from external systems, invoking the necessary adapters to perform the business logic, and composing a response and sending it back to the requesting system. For outgoing requests, the gateway is responsible for sending credit card settlement requests and international funds transfer requests.

The remaining conceptual elements should also be documented at a similar level of detail.

As SOA emerges as an architecture paradigm, some variations to the enterprise architecture view are emerging. Especially enterprises that have ridden the SOA maturity curve have started acknowledging that their enterprise service portfolios are significant enough to become part of

their enterprise architectures. A set of enterprise services can be represented along with the list of core business processes. As enterprise architecture matures as a discipline, we may see different architecture artifacts making their way into the enterprise architecture view. Currently, the architecture constructs represented in Figure 1 are basic, but have withstood the test of time.

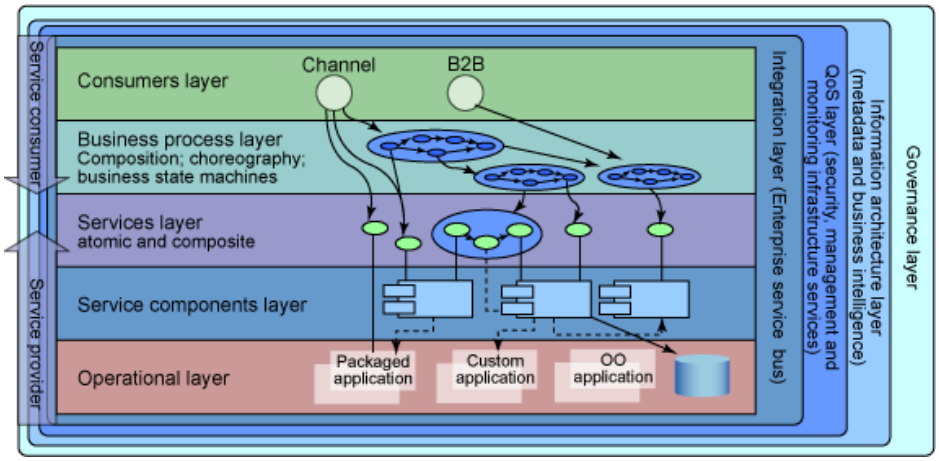
Layered architecture view

The layered view introduces another approach to grouping architecture components. It is the view most commonly used to represent the IT architecture. An architecture *layer* exhibits a certain set of characteristics and constraints. Architecture components, when identified, are evaluated against the characteristics of the architecture layers. The evaluation yields a grouping of the architecture components into layers. The architecture layers provide a logical separation of architecture components.

The layered architecture model introduces principles and guidelines that impose discipline and rigor on the identification of the correct architecture components, based on the guiding principles laid down by each of the architecture layers.

There are several different ways to represent a layered architecture. The nine-layered view, shown in Figure 2, is recommended.

Figure 2. Layered architecture view depicting an SOA reference architecture



The architecture view in Figure 2 introduces SOA as the architecture paradigm used in the solution. I recommend this view, because it addresses any architecture that is SOA based or non-SOA based. If it is non-SOA based, the services layer is not be required, and the service components layer may be replaced by a components layer.

The example has five horizontal and four vertical layers. The horizontal layers follow the basic principle of a layered architecture model where architecture building blocks from layers above can access ones from layers below, but layers below may not access architecture building blocks from layers above. The vertical layers usually contain architecture building blocks that are cross-cutting, which means they may be applicable to and used by architecture building blocks in one or more of the horizontal layers. This can also be called a partially layered architecture because any layer above does not need to strictly interact with elements from its immediate lower layer.

For example, a specific access channel can directly access a service rather than having to go through a business process. The access constraints, however, are dictated by the applicable architectural style, guidelines, and principles for a given SOA solution. This view of the SOA reference architecture is independent of any specific technology, so it is a logical view. Instances of the logical architecture can be developed for a specific platform and technology.

The following definitions of the layers are to help architects identify architecture components and put them in the proper layers.

Layer	Definition
1 - Operational systems	Includes the operational systems that support business activities in the current IT environment of the enterprise. Includes all custom and packaged applications, legacy systems, transaction processing systems, and the various databases.
2 - Service component	Components in this layer conform to contracts defined by services in services layer. This layer is usually a one-to-one mapping between a service and a service component. A service component provides an implementation facade that aggregates functions from multiple, possibly disparate, operational systems while hiding the integration and access complexities from the service that's exposed to the consumer.
3 - Services	Includes all services that are defined in the enterprise service portfolio and the definition of each service with both syntactic and semantic information. Syntactic information is for the operations on each service, the input and output messages, and definition of the service faults. Semantic information is for the service policies, service management decisions, service access requirements, and so on. Services are defined so they are accessible by channels and consumers, independent of implementation and transport.
4 - Business process	Business processes depict how the business runs. A business process is an IT representation of the activities that are coordinated in an enterprise to perform a specific business function. This layer represents the processes as an orchestration of loosely coupled services. It is also responsible for entire life cycle management of the process orchestration. Processes in this layer connect business requirements and their manifestation as an IT-level solution using architecture building blocks from other horizontal and vertical layers. Users, channels, and B2B partner systems in the consumer layer use the business processes in this layer as a way to invoke application functions.
5 - Consumers	Depicts the various channels through which the IT functions are delivered. Channels can be different user types, such as external or internal consumers who access application functions through various mechanisms (B2B systems, portals, rich clients, and other forms).
6 - Integration	Provides capability for services consumers to locate service providers and initiate service invocations. Through mediation, routing, and data and protocol transformation, this layer helps foster a service ecosystem where services can communicate with each other while being part of a business process. It provides key nonfunctional requirements such as security, latency, and quality of service between adjacent layers in the reference architecture. Functions of this layer are increasingly being collectively defined as the enterprise service bus (ESB).
7 - Quality of service	Focuses on implementing and managing the nonfunctional requirements that services and components need to implement by providing infrastructure capabilities. This layer captures the data elements that provide the information around noncompliance to nonfunctional requirements at each of the horizontal layers. Standard nonfunctional requirements that it monitors are security, availability, scalability, and reliability.
8 - Information architecture	Ensures a proper representation of the data and information required in an architecture. This layer is responsible for the data architecture and information architecture representations and key considerations and guidelines for design and use of each horizontal layer. This layer addresses industry models such as ACCORD, IAA, JXDD and their usage to define the information architecture and business protocols used to exchange business data. It also stores the metadata required for data mining and business intelligence.
9 - Governance	Ensures the proper management of the entire life cycle of services. This layer is responsible for prioritizing which high-value services should be implemented for each layer in the architecture and provides a

	rationalization based on how the service satisfies a business or IT goal. Another key responsibility of this layer is enforcing design and run-time policies that the services should implement and conform to.

Executing SOA - A Practical Guide for the Service Oriented Architect (in Resources) has a detailed treatment of layered architecture.

Based on the definitions here, you can identify architecture components and place them in one of the nine layers. You would then need to document each architecture component to communicate its role and the responsibility it fulfills in the overall solution architecture.

IT system view

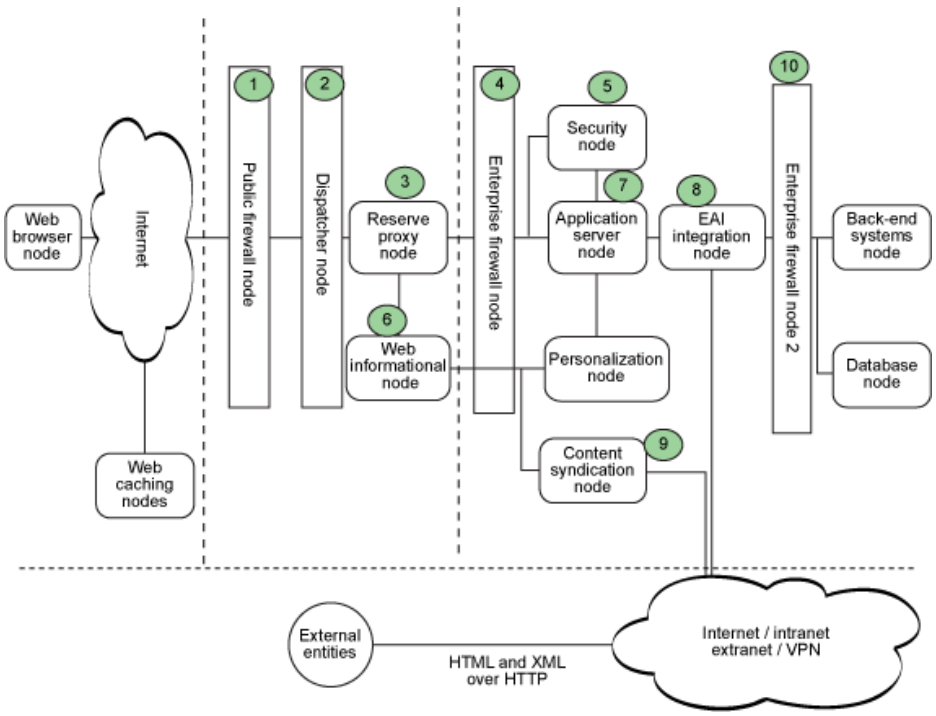
The IT system view provides an additional level of detail and shows the main nodes of the architecture. The nodes are components with a clearly defined functional role to play in the architecture. Well-structured application programming interfaces (APIs) and protocols are required between nodes. The nodes are logical constructs and do not correspond directly to servers.

The IT system view will be the starting point for the component model and operational model, and it may be extended or refined during the definition of these two models. The operational model will map the IT view to actual servers. A given server may support multiple nodes, or a given node may be deployed on multiple servers.

Component and operational models will be discussed in detail in a future article.

Figure 3 shows a typical IT system level view of the developing architecture of a system solution.

Figure 3. Sample IT system view



You should document the IT system view to provide enough information at an architecture overview level for subsequent development of other, more detailed architecture artifacts.

It is beyond the scope of this article to describe each node in Figure 3 in great detail. I recommend that you use a template to provide necessary and sufficient documentation of each of the nodes. Before discussing the template, though, here's a single line of description for each of the numbered nodes in Figure 3.

1. Public firewall node - resides in the demilitarized zone (DMZ) and allows only HTTP traffic to enter into the enterprise network
2. Dispatcher node - used for load balancing of multiple requests across a cluster of Web information nodes and application server nodes
3. Reverse proxy node - used for hardening Web servers around security considerations; acts as an interceptor of any requests from external clients; provides authentication and authorization capabilities through a single sign-on mechanism
4. Enterprise firewall node - opens only some selected and secure ports into the secured enterprise network, thereby protecting critical enterprise applications and data through a double door mechanism
5. Security node - provides authentication and authorization for some of the enterprise applications, databases, mainframe systems, and packaged applications
6. Web information node - serves only static content and enhances performance of Web-based applications
7. Application server node - handles business logic and transactional content,and executes business processes
8. EAI integration node - provides capabilities to integrate with back-end systems
9. Content syndication node - aggregates and publishes content
10. Enterprise firewall node 2 - Used in hosted situations where part of the IT system topology is hosted and maintained by third-party vendors

The single-sentence documentation above is moot if there's enough information available for more detailed documentation of each node in the IT system topology. The rest of this section provides a template for such an example.

Each node in the IT system view is expected to have the following information:

- Node name - specifies the name of the node
- Description - a detailed description of the characteristics and features of the node
- Services or components - describe the services or components that are running on the node, such as relational database, transaction manager, state management, and so on
- Nonfunctional characteristics - describes the nonfunctional characteristics that the node must exhibit and support
- Connections to other nodes - enlists the various others nodes in the topology that it connects to

Let's use application server node (number 7 in Figure 3) for an example of detailed documentation:

Node name

Application server

Description

The application server node is responsible for processing transactions and providing Web users access to back-end systems and databases. It supports Web transactions and provides many of the operational characteristics identified by the operational requirements of the application, including multithreaded servers to support multiple client connections, redundant services to handle additional loads, dynamic load balancing capabilities, automatic fail-over, and recoverability.

The application server node provides an integration point for the back-end applications and databases. This node, which could be made up of many servers, will host the connectors and gateways into the back-end systems.

Services or components

The Web application server node, which is an instance of the application server node, will have the following components executing:

- Application server software
- Java™ Virtual Machine
- Software for monitoring Web site usage and gathering statistics
- Systems management software to detect, diagnose, and automatically correct failures, and automatic notification of support personnel.

Nonfunctional characteristics

The traffic through this node, measured in number of money transfers processed in an hour (hits per hour), is planned at two levels:

- Average or typical load encountered during most times in a day during most of the trading day
- Peak load at different times during a normal trading day

Response time requirements, measured in the peak hour from receipt of a request for data to the final response message (when the server responds with a page), must not exceed eight seconds for 90 percent of transactions. It is important to have the ability to scale this node as needed by adding more instances of this node (hardware and software).

This node must be available 24 hours a day and seven days a week. Failover is provided to ensure availability. Applications running on this node must have the same service availability characteristics as that of the node.

Connections to other nodes

The node is directly connected to four nodes:

- Enterprise firewall node that provides security at the network protocol level
- Personalization node that provides targeted application features and capabilities
- Security node that propagates security credentials to the applications
- EAI integration node that provides integration logic, facilitating back-end integration with database systems, packaged applications, and legacy systems

Representing the above information in tabular format might make it easier for the reader. Each of the nodes should have the level of documentation shown in this example.

Conclusion

In this article, you learned that the architecture overview document is a fundamental artifact that helps shape the software architecture of any system. Three views—enterprise, layered, and IT system—are commonly used in developing the architecture overview. There are effective ways to thoroughly document the three views that make up this important architecture artifact. I highly recommend that you do an architecture overview as a deliverable in any software architecture.

I hope this article influences architects to put time into developing this artifact and influences project managers to allow time for such development.

Stay tuned for more articles in this series, which will elaborate on the rest of the architecture artifacts, and how to use guidelines for documenting each of them.

Resources

- Check out the other parts of this series:
  - Part 1, "What software architecture is, and why it's important to document it"
  - Part 2, "Develop the system context"
- Get the RSS feed for this series.
- Learn more about SOA in *Executing SOA: A Practical Guide for the Service-Oriented Architect*, a recently published developerWorks series book that Tilak coauthored. Use coupon code IBM3748 for a 35% discount.
- Read a compendium of published software architecture definitions.
- "Design an SOA solution using a reference architecture" (developerWorks, Mar 2007) provides information about the SOA solution stack and usage of the SOMA methodology to instantiate it.
- Learn more about SOA in *Executing SOA: A Practical Guide for the Service-Oriented Architect*, a recently published developerWorks series book that Tilak coauthored. Use coupon code IBM3748 for a 35 percent discount.
- In the Architecture area on developerWorks, get the resources you need to advance your skills in the architecture arena.
- Browse the technology bookstore for books on these and other technical topics.
- Check out developerWorks blogs and get involved in the developerWorks community.

About the author

Tilak Mitra

Tilak Mitra is a Senior Certified Executive IT Architect in IBM. He specializes in SOAs, helping IBM in its business strategy and direction in SOA. He also works as an SOA subject matter expert, helping clients in their SOA-based business transformation, with a focus on complex and large-scale enterprise architectures. His current focus is on building reusable assets around Composite Business Services (CBS) that has the ability to run on multiple platforms like the SOA stacks for IBM, SAP and so on. Tilak recently coauthored a developerWorks series book: *Executing SOA: A Practical Guide for the Service-Oriented Architect*, available in Resources. He lives in sunny South Florida and, while not at work, is engrossed in the games of cricket and table tennis. Tilak did his Bachelors in Physics from Presidency College, Calcutta, India, and has an Integrated Bachelors and Masters in EE from Indian Institute of Science, Bangalore, India. Find out more about SOA at Tilak's blog. View Tilak Mitra's profile on LinkedIn or e-mail him with your suggestions at tmitra@us.ibm.com.