USN | | | | | | | | | | |

Internal Assessment Test 1 – Sept. 2018

Sub:	SERVICE ORIENTED ARCHITECTURE					Sub Code:	16MCA553	Branch:	MCA	
Date:	10 / 09 / 2018	Duration:	90 min's	Max Marks:	50	Sem / Sec:	V		OBE	

Answer any FIVE FULL Questions

		MARKS	CO	RBT
1 (a)	Define SOA and list down the design principles of SOA	[05]	CO1	L1
(b)	Discuss the common tangible benefits of SOA	[05]	CO1	L2
2 (a)	List all the common characteristics of contemporary SOA and Explain any eight	[10]	CO1	L1
3 (a)	Discuss about the standards organizations and major vendor that contribute to SOA	[06]	CO1	L2
(b)	What are the different types of service models? Explain any two	[04]	CO1	L1
4 (a)	What is an architecture? Explain the different types of architecture	[05]	CO1	L2
(b)	Explain distributed internet architecture and compare with SOA	[05]	CO1	L2
5 (a)	Define WSDL. Explain service descriptions using WSDL	[10]	CO1	L2

USN | | | | | | | | | | |

Internal Assessment Test 1 – Sept. 2018

Sub:	SERVICE ORIENTED ARCHITECTURE					Sub Code:	16MCA553	Branch:	MCA	
Date:	10 / 09 / 2018	Duration:	90 min's	Max Marks:	50	Sem / Sec:	V		OBE	

Answer any FIVE FULL Questions

		MARKS	CO	RBT
1 (a)	Define SOA and list down the design principles of SOA	[05]	CO1	L1
(b)	Discuss the common tangible benefits of SOA	[05]	CO1	L2
2 (a)	List all the common characteristics of contemporary SOA and Explain any Eight	[10]	CO1	L1
3 (a)	Discuss about the standards organizations and major vendor that contribute to SOA	[06]	CO1	L2
(b)	What are the different types of service models? Explain any two	[04]	CO1	L1
4 (a)	What is architecture? Explain the different types of architecture	[05]	CO1	L2
(b)	Explain distributed internet architecture and compare with SOA	[05]	CO1	L2
5 (a)	Define WSDL. Explain service descriptions using WSDL	[10]	CO1	L2

		MARKS	CO	RBT
6 (a)	Explain SOAP message frameworks and SOAP nodes	[10]	CO1	L2
7 (a)	Define MEP? Discuss the various types of MEPS with neat diagram	[10]	CO2	L2
8 (a)	Explain elaborately the concept of coordination related to WS with neat diagram	[10]	CO2	L2

		MARKS	CO	RBT
6 (a)	Explain SOAP message frameworks and SOAP nodes	[10]	CO1	L2
7 (a)	Define MEP? Discuss the various types of MEPS with neat diagram	[10]	CO2	L2
8 (a)	Explain elaborately the concept of coordination related to WS with neat diagram	[10]	CO2	L2

Service Oriented Architecture – 16MCA553

Sep 2018 – Internal Test - 1 – Answer Key

1.a) Define SOA and list down the design principles of SOA

Definition

SOA is a form of technology architecture that adheres to the principles of service-orientation.
When realized through the Web services technology platform, SOA establishes the potential to support and promote these principles throughout the business process and automation domains of an enterprise.

Principles of SOA

- *Loose coupling* Services maintain a relationship that minimizes dependencies and only requires that they retain an awareness of each other.
- *Service contract* Services adhere to a communications agreement, as defined collectively by one or more service descriptions and related documents.

- *Autonomy* Services have control over the logic they encapsulate.
- *Abstraction* Beyond what is described in the service contract, services hide logic from the outside world.
- *Reusability* Logic is divided into services with the intention of promoting reuse.
- *Composability* Collections of services can be coordinated and assembled to form composite services.
- *Statelessness* Services minimize retaining information specific to an activity.
- *Discoverability* Services are designed to be outwardly descriptive so that they can be found and assessed via available discovery mechanisms.

1. b) Discuss the common tangible benefits of SOA

Improved integration (and intrinsic interoperability)

 SOA creates solutions that consist of essentially interoperable services.

 A cross-application integration project into less of a custom development effort, and more of a modeling exercise.

 The bottom line: The cost and effort of cross-application integration is significantly lowered when applications being integrated are SOA-compliant.

Inherent reuse

 Service-orientation promotes the design of services that are inherently reusable.

 Designing services to support reuse from the get-go opens the door to increased opportunities for leveraging (force) existing automation logic.

Streamlined architectures and solutions

 The concept of composition is another fundamental part of SOA. .

 These are extensions to the basic Web services framework established by first-generation standards represented by WSDL(Web Service Description Language), SOAP(Small Object Access Protocol), and UDDI (Universal Description, Discovery, and Integration)

Leveraging the legacy investment

 Web services technology set has generated a large adapter market, enabling many legacy environments to participate in service-oriented integration architectures.

 This allows IT departments to work toward a state of federation, where previously isolated environments now can interoperate without requiring the development of expensive and sometimes fragile point-to-point integration channels.

 Still riddled with risks relating mostly

- to how legacy back-ends must cope with increased usage volumes,
- the ability to use what you already have with service-oriented solutions that you are building now and in the future is extremely attractive.

Establishing standardized XML data representation

 SOA is built upon and driven by XML. An adoption of SOA leads to the opportunity to fully leverage the XML data representation platform.

 A standardized data representation format (once fully established) can reduce the complexity of application environments.

Focused investment on communications infrastructure

 Web services establish a common communications framework,

 SOA can centralize inter-application and intra-application communication as part of standard IT infrastructure.

 This allows organizations to evolve enterprise-wide infrastructure by investing in a single technology set responsible for communication.

"Best-of-breed" alternatives

 A key feature of service-oriented enterprise environments is the support of "best-of-breed" technology.

 Because SOA establishes a vendor-neutral communications framework, it frees IT departments from being chained to a single proprietary development and/or middleware platform.

 For any given piece of automation that can expose an adequate service interface, you now have a choice as to how you want to build the service that implements it.

Organizational agility

 Agility is a quality inherent in just about any aspect of the enterprise.

 All parts contain a measure of agility related to how they are constructed, positioned, and leveraged.

 Regardless of what parts of service-oriented environments are leveraged, the increased agility with which IT can respond to business process or technology-related changes is significant.

 The bottom line: The cost and effort to respond and adapt to business or technology-related change is reduced.

2. List all the common characteristics of contemporary SOA and Explain any eight

1. Contemporary SOA is at the core of the service-oriented computing platform.
2. Contemporary SOA increases quality of service.
3. Contemporary SOA is fundamentally autonomous.
4. Contemporary SOA is based on open standards.
5. Contemporary SOA supports vendor diversity.
6. Contemporary SOA fosters intrinsic interoperability.
7. Contemporary SOA promotes discovery.
8. Contemporary SOA promotes federation.
9. Contemporary SOA promotes architectural composability.
10. Contemporary SOA fosters inherent reusability.
11. Contemporary SOA emphasizes extensibility.
12. Contemporary SOA supports a service-oriented business modeling paradigm.
13. Contemporary SOA implements layers of abstraction.
14. Contemporary SOA promotes loose coupling throughout the enterprise.
15. Contemporary SOA promotes organizational agility.
16. Contemporary SOA is a building block.
17. Contemporary SOA is an evolution.
18. Contemporary SOA is still maturing.
19. Contemporary SOA is an achievable ideal.

1. **Contemporary SOA is at the core of the service-oriented computing platform.**
   - SOA is used to qualify products, designs, and technologies an application computing platform consisting of Web services technology and service-orientation principles
   - *Contemporary SOA represents an architecture that promotes service-orientation through the use of Web services*.
2. **Contemporary SOA increases quality of service.**
   - The ability for tasks to be carried out in a secure manner, protecting the contents of a message, as well as access to individual services.
   - Allowing tasks to be carried out reliably so that message delivery or notification of failed delivery can be guaranteed.

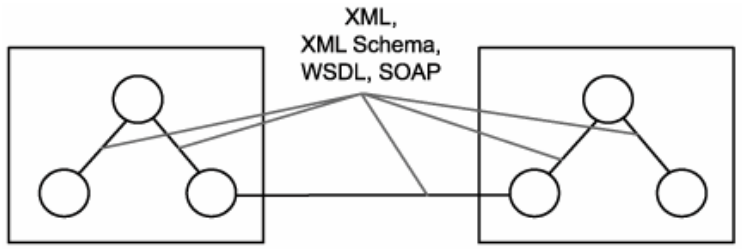
- Performance requirements to ensure that the overhead imposed by SOAP message and XML content processing does not inhibit the execution of a task.
- Transactional capabilities to protect the integrity of specific business tasks with a guarantee that should the task fail, exception logic is executed.

3.Contemporary SOA is fundamentally autonomous.

- The service-orientation principle of autonomy requires that
  - individual services be as independent and
  - self-contained as possible with respect to the control they maintain over their underlying logic.

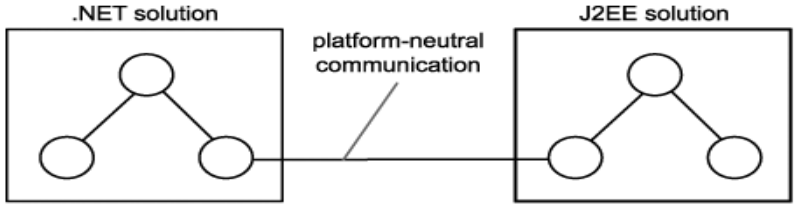
4.Contemporary SOA is based on open standards.

- Significant characteristic of Web services is the fact that
  - data exchange is governed by open standards.
  - After a message is sent from one Web service to another it travels via a set of protocols that is globally standardized and accepted.



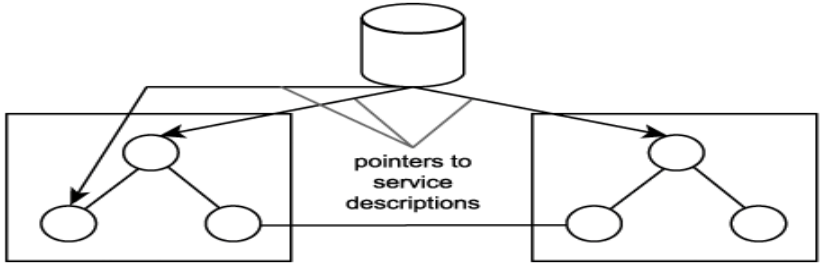
5.Contemporary SOA supports vendor diversity.

- Organizations continue itsbuilding solutions with existing development tools and server products.
- It is continue to leveraging(maximizing) the skill sets of in-house resources.
- Choice to explore the offerings of new vendors is always possible.
- This option is made possible by the
  - open technology provided by the Web services framework
  - the standardization and principles introduced by SOA.



6.Contemporary SOA promotes discovery.

- SOA supports and encourages the advertisement and discovery of services throughout the enterprise and beyond.
- A serious SOA will likely rely on some form of service registry or directory to manage service descriptions

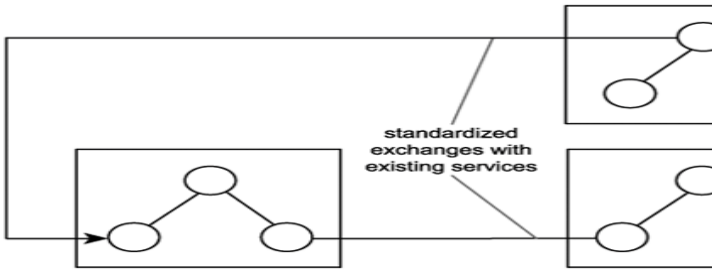


7.Contemporary SOA foster(advance) intrinsic(essential) interoperability.

- To leveraging(maximizing) and supporting the
  - required usage of open standards,
  - a vendor diverse environment, and
  - the availability of a discovery mechanism

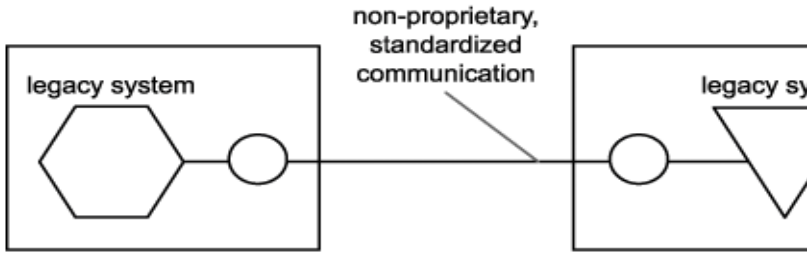
  is called intrinsic interoperability
- Whether an application actually has immediate integration requirements or not design principles can be

applied to outfit services with characteristics that naturally promote interoperability.



8.Contemporary SOA promotes federation.

- Establishing SOA within an enterprise does not necessarily require that you replace what you already have.
- SOA has the ability to introduce unity across previously non-federated environments.
- Web services enable federation
- SOA promotes by establishing and standardizing the ability to encapsulate legacy and non-legacy application logic and by exposing it via a common, open, and standardized communications framework



3.a) Discuss about the standards organizations and major vendor that contribute to SOA

Standards organizations that contribute to SOA

- Standards are produced, though, is not always that clear.
- Internet standards organizations have existed for some time now, but their respective agendas are not always distinct and sometimes even overlap.
- Microsoft, IBM, Sun Microsystems, and many others have played ly significant roles in formalizing Web services specifications, and accelerating the implementation of these specifications as industry standards.
- Let's first learn more about the three most prominent standards organizations.
- Collectively, they are responsible for seeing through the evolution of XML and Web services architectures.

The World Wide Web Consortium – (W3C)

- Founded by Tim Berners-Lee in 1994
- W3C has been hugely responsible for furthering the World Wide Web as a global, semantic medium for information sharing.
- First released HTML, one of the most popular technical languages
- Increased in eBusiness, the W3C responded by producing key foundation standards based on XML, such as XML Schema and XSLT.
- Four separate working groups made significant contributions to W3C Web Services Activity projects
- They developed of important base standards for Web services.
- First-most are the SOAP and WSDL standards,
- Recently, the W3C has produced the Web Services Choreography Description Language (WS-CDL) - A specification that governs standardized inter-service exchange patterns.
- Web Services Architecture document - it remains a reference point.
- The W3C is known for its formal and accurate approach to standards development.
- Specifications be subjected to numerous review and revision stages, with each new version being published to their public Web site.
- Standards can take two to three years to be completed.

Organization for the Advancement of Structured Information Standards (OASIS)

- Established in 1993 as the SGML, 5 years later changed to OASIS (SGML to XML-related standards)
- 1000 members from 600 organizations,
- International standards producing organization.
- OASIS a
- WS-BPEL specification
- ebXML (a specification that aims to establish a standardized means of B2B data interchange)
- UDDI specification (Core std for First generation web service)
- XML and Web services security extensions.
- Security Assertion Markup Language (SAML)
- Extensible Access Control Markup Language (XACML) provide important features in the areas of single sign-on and authorization.
- Web Services Security (WSS) technical committee - Security-related project. Further developing and realizing the important WS-Security framework.
- W3C focuses on establishing core, industry-agnostic standards
- OASIS group's primary interests lie in leveraging these standards to produce additional specifications that support various vertical industries.

The Web Services Interoperability Organization (WS-I)

- Object of WS-I is open interoperability
- Established in 2002, 200 organizations, including all major SOA vendors.
- Releasing the Basic Profile, a recommendation-based document that contains available standards collectively used in interoperability architec.
- WSDL, SOAP, UDDI, XML, and XML Schema, the Basic Profile has become an important document within the IT community.
- Developed the Basic Security Profile. (most important collection of Web services and XML security technologies.
- It continue releasing Profiles for each major aspect of Web services-related interoperability, reliable messaging, Web service management, and orchestration.
- Profiles also supplement
- sample implementations and
- best practices on how the standards are to be used together to achieve a quality level of interoperability.
- Provides a series of testing tools that can be used to ensure compliance with Profiles.
- Validity checkers that use Basic Profile conformance as part of the validation criteria.
- Membership includes significant SOA vendors, no one company has more power than another, regardless of its size or market share.
- W3C recently rejected an invitation to become an associate member of the WS-I
- Working group members from the WS-I continue to contribute to W3C and OASIS initiatives by directly participating in their respective working groups.
- The role of these WS-I representatives is to provide continual feedback relating to interoperability issues.

4.2.3. Major vendors that contribute to SOA

- Standards organizations have their own culture and philosophies around how standards should be developed, they are all heavily influenced by the commercial market.
- Vendors supply a significant portion of the contributors that actually end up developing the standards.
- Some of the companies that have participated in the standards development processes include Microsoft, IBM, BEA Systems, Sun Microsystems, Oracle, Tibco, Hewlett-Packard, Canon, Commerce One, Fujitsu, Software AG, Nortel, Verisign, and WebMethods.

The vendor influence

- IBM has laid out a technology path for increasing support of SOA within its WebSphere platform.
- Microsoft increasing SOA features within the .NET

technology framework, and building Web services technology for Windows

- Web services non-proprietary, a vendor who can help shape a standard might be motivated to do so with proprietary technology considerations in mind.
- Challenge - getting *all vendors to agree on how one* standard should be designed.

Vendor alliances

- Battle between most established vendor leads distrust.
- Collaborate on specifications (interoperability) between vendor platforms turn into obstacles.
- Forming an alliance allows vendors to join forces in order to attain common goals.
- Lifespan of an alliance is based on d the development cycle of a specification.
- Most noticeable team of repeat-collaborators (IBM, Microsoft,
- and BEA)
- Persisted their working relationship to push forward a series of WS-* extensions.
- One of the more talked about examples of alliances playing a significant role in standards development is the creation of the

Choosing a standards organization

- Choice of standards organization can have implications.
- Standards development arena is directly related to market demand.
- Vendors have market-driven goals fueled by pressures to deliver product releases that meet customer demands and match
- Given that the W3C relies on a longer standards development process, it is tempting for vendors to submit their standards to OASIS instead.
- Organizations develop similar specifications may seem redundant; one always seems to rise to the top.
- And despite the fact that opposing motives may seem counter-productive to fostering a collection of platform-neutral technology standards, the quality of what's been delivered so far has been adequate for furthering the cause of SOA.

3.b) What are the different types of service models? Explain any two

The services are classified

- The manner in which services are being utilized in the real world
- Nature of the application logic they provide
- Their business-related roles within the overall solution.

These classifications are known as *service models*.

Business service model

- Most fundamental building block.
- It encapsulates a distinct set of business logic within a well-defined functional boundary
- Business services are used within SOAs as follows:
  - as fundamental building blocks for the representation of business logic
  - to represent a corporate entity or information set
  - to represent business process logic
  - as service composition members
- when building an SOA around layers of abstraction, the business service model can correspond to the business service layer
- the business service would act as a controller, composing utility application services.

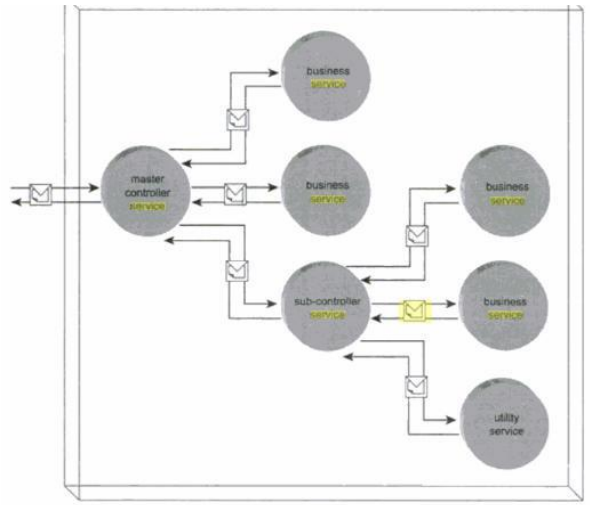


Figure 5.12
A service composition consisting of a master controller, a sub-controller, four business services, and one utility service.

Utility service model

- Any generic Web service or service agent designed for potential reuse can be classified as a *utility service*.
- The key to achieving this classification is that the reusable functionality be completely generic and non-application specific in nature.
- Utility services are used within SOAs as follows:
  - as services that enable the characteristic of reuse within SOA
  - as solution-agnostic intermediary services
  - as services that promote the intrinsic interoperability characteristic of SOA
  - as the services with the highest degree of autonomy
- When working with the service abstraction layers described in, a utility service is most commonly associated with the application service layer.
- A utility service can be referred to as a *utility application service*.

4.a) What is an architecture? Explain the different types of architecture

Application Architecture

- With the rise of multi-tier applications, the variations with which applications could be delivered began to increase
- A definition of a baseline definition application becomes important
- The definition is
  - abstract in nature,
  - but specifically explained the technology,
  - boundaries,
  - rules,
  - limitations, and
  - design characteristics that apply to all solutions - *application architecture.*
- An application architecture is a blueprint
- Different levels can be specified, depending on the organization Some keep it
  - high-level,
  - providing abstract physical and logical representations of the technical blueprint.
- Some more detail, such as
  - common data models,
  - communication flow diagrams,
  - application-wide security requirements, and
  - aspects of infrastructure.
- An organization that houses both .NET and J2EE solutions
- Having separate application architecture specifications for each.
- Key part - it should reflect immediate solution requirements, as well as long-term, strategic IT goals.

Enterprise Architecture

- In larger IT, Different application architectures co-exist and even integrate, the demands on the underlying hosting platforms can be complex.
- Master specification to be created, providing a high-level overview of all forms of heterogeneity that exist within an enterprise, as well as a definition of the supporting infrastructure.
- Enterprise architecture specification - what an urban plan is to a city.
- Relationship between an urban plan and the blueprint of a building are comparable to that of enterprise and application architecture specifications.

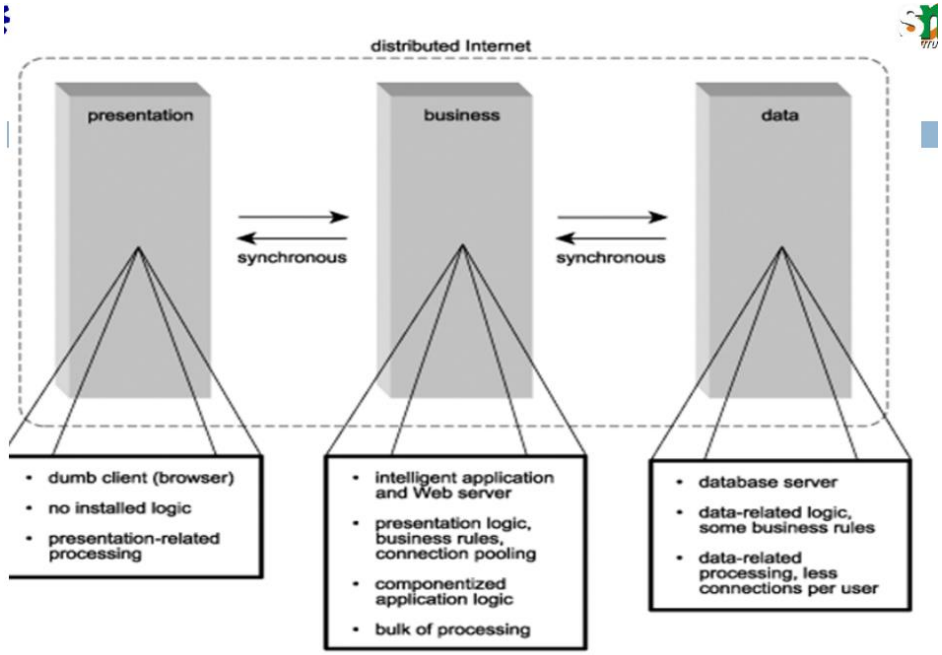
- Changes to enterprise architectures directly affect application architectures
- Architecture specifications often are maintained by the same group of individuals.
- EA contains a long-term vision of how the organization plans to evolve its technology and environments.
- For example, the goal of phasing out an outdated technology platform may be established in this specification.

Service Oriented Architecture

- Service-oriented architecture extents both enterprise and application architecture domains.
- The benefit offered by SOA can be realized when applied across multiple solution environments.
- Building reusable and interoperable services based on a vendor-neutral communications platform can fully be leveraged.
- This does not mean that the entire enterprise must become service-oriented. SOA belongs in those areas that have the most to gain from the features and characteristics it introduces.
- "SOA" does not imply a particular architectural scope.
- SOA refer to an application architecture or the approach used to standardize technical architecture across the enterprise.
- Because of the composable nature of SOA, it is absolutely possible for an organization to have more than one SOA.
- Web services platform offers one of a number of available forms of implementation for SOA.

4.b) Explain distributed internet architecture and compare with SOA

- Reduces problem of centralization on server.
- Dedicated servers which shares and manages database connections



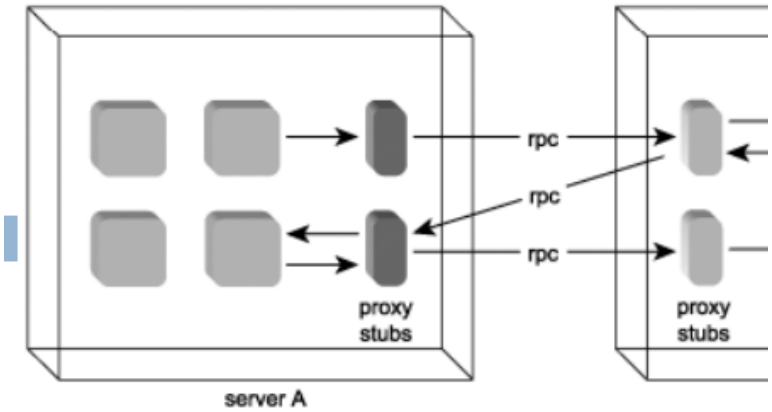
Brief History

- Share and manage pool of database connections– eliminate the concurrent usage on the DB server
- Component capable of processing multiple and concurrent request – difficult
- Late 90's – defacto computing platform for custom developed enterprise solutions.

Application Logic

- Application Logic – Server.
- None of the logic – Client
  - How application logic is partitioned
  - Where partitioned units reside
  - How units of logic should interact
- Traditional systems create components that reside on one or more application servers
- Components have varying degrees of functional granularity

- Components on the same server communicate via proprietary APIs.
- RPC protocols are used across servers via proxy stubs
- Actual references to other physical components can be embedded in programming code (tight coupling)
- Not easily altered.
- SOAs also rely on components
- Services encapsulate components
- Services expose specific sets of functionality
- Functionality can originate from legacy systems or other sources



- Functionality is wrapped in services and is exposed via open, standardized interface, irrespective of technology providing the solution
- Services exchange information via SOAP messages.
- Most applications rely on document-style
- Messages are structured to be self-sufficient and contain meta information, processing instructions, policy rules
- SOA fosters reuse on a deep level by promoting solution-agnostic services
- Promotes the proprietary protocols like DCOM and CORBA for remote data exchange.
- Relatively efficient and reliable
- It supports creation of stateful and stateless components.

Application Processing

- SOA relies on message based communication, it involves serialization and deserialization of SOAP messages containing XML document payloads
  - Distributed
    - Server Side Script
    - HTML
    - XML *
    - Web Services *
  - SOA
    - Newer versions of VB
    - RDMS
    - HTML, CSS + XML with SOAP etc.,

  * Optional for distributed architecture
- provides header in security logic can be stored.

Administration

- Distributed architecture introduces web server and physical environment using HTTP.
- SOA requires additional runtime administration. ♦
- Problems with messaging framework can go ♦ undetected than with RPC based data exchange.

5) Define WSDL. Explain service descriptions using WSDL

Service Description provides the key to establishing a consistently loosely coupled form of communication between services implemented as Web services.
Description documents are required to accompany any service wanting to act as an ultimate receiver.

The primary service description document is the WSDL definition

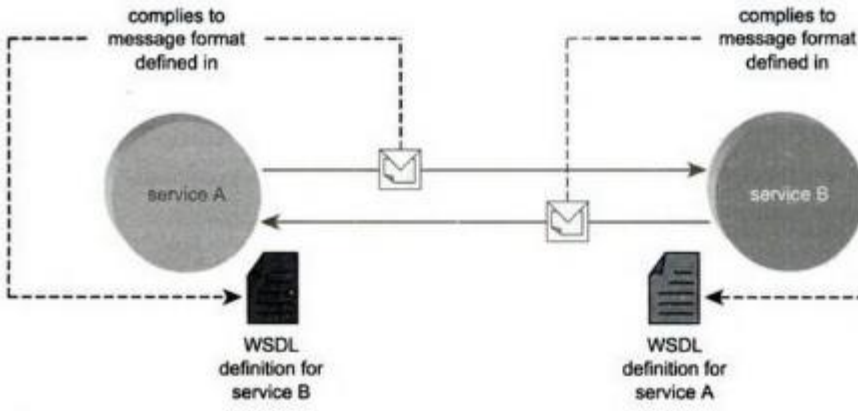


Figure 5.14
WSDL definitions enable loose coupling between services.

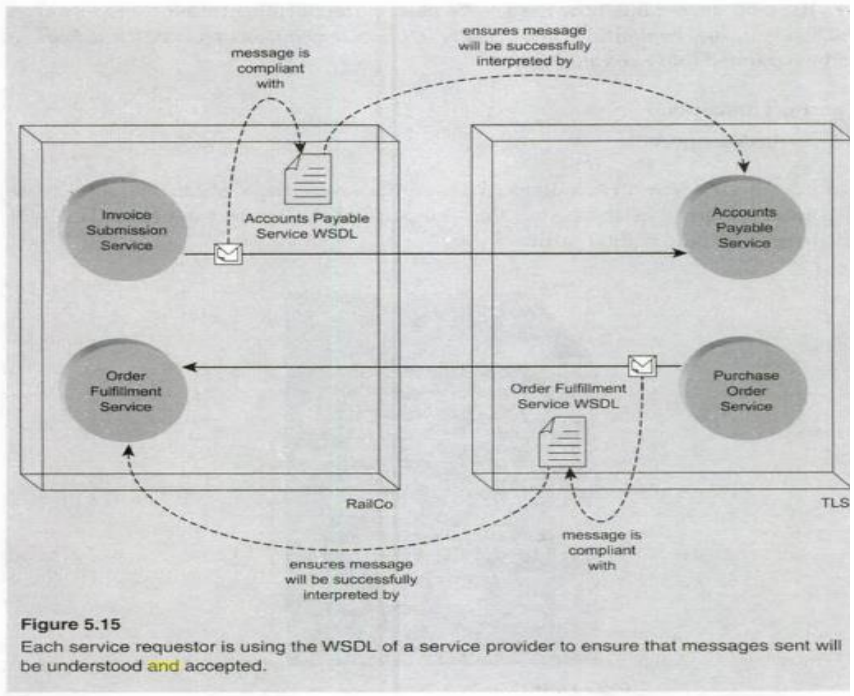


Figure 5.15
Each service requestor is using the WSDL of a service provider to ensure that messages sent will be understood and accepted.

5.3.1. Service endpoints and service descriptions

- A WSDL describes the point of contact for a service provider, also known as the *service endpoint* or just *endpoint*.
- It provides a formal definition of the endpoint interface and also establishes the physical location of the service.
- A WSDL service description can be separated into two categories:
  - abstract description
  - concrete description

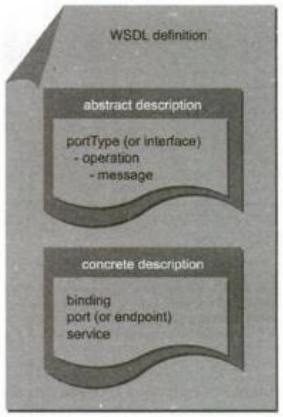


Figure 5.16
WSDL document consisting of abstract and concrete parts that collectively describe a service endpoint.

5.3.2. Abstract description

An *abstract description* establishes the interface characteristics of the Web service without any reference to the technology used to host or enable a Web service to transmit messages.

portType, operation, and message

- The parent *portType* section of an abstract description provides a high-level view of the service interface by sorting the messages a service can process into groups of functions known as *operations*.
- Each operation represents a specific action performed by the service.
- A service operation is comparable to a public method used by components in traditional distributed applications.
- Much like component methods, operations also have input and output parameters.

- Web services rely exclusively on messaging-based communication, parameters are represented as messages.
- An operation consists of a set of *input and output messages*.

5.3.3. Concrete description

- abstract interface definition to be connected to some real, implemented technology.
- Because the execution of service application logic always involves communication, the abstract Web service interface needs to be connected to a physical transport protocol.
- This connection is defined in the *concrete description* portion of the WSDL file, which consists of three related parts:

binding, port, and service

- A WSDL description's *binding* describes the requirements for a service to establish physical connections or for connections to be established with the service.
- SOAP is the most common form of binding, but others also are supported.
- A binding can apply to an entire interface or just a specific operation.
- *port*, which represents the physical address at which a service can be accessed with a specific protocol.
- *service* is used to refer to a group of related endpoints

5.3.4. Metadata and service contracts

- How a service can be interfaced with and what type of data exchange it supports.
- WSDL rely on XSD schemas to formalize the structure of incoming and outgoing messages.
- supplemental service description document is a *policy*.
- Policies can provide rules, preferences, and processing details above and beyond what is expressed through the WSDL and XSD schema documents.
- Three separate documents that each describe an aspect of a service:
    - WSDL definition
    - XSD schema
    - Policy

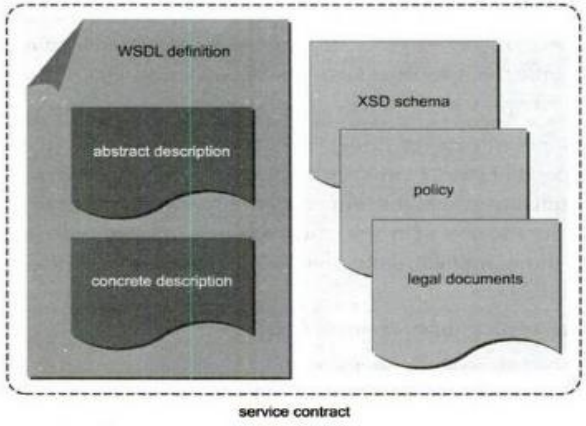


Figure 5.17
A service contract comprised of a collection of service descriptions and possibly additional documents.

- Each of these three service description documents can be classified as service *metadata since they* provides information about the service.
- Service description documents called *service contract*a set of conditions that must be met and accepted by a potential service requestor to enable successful communication.
- Service contract can refer to additional documents or agreements not expressed by service descriptions.
- For example, a Service Level Agreement (SLA)

5.3.5. Semantic descriptions

- Technical data alone is not enough to describe the service
- Service's behavioral characteristics also important
- Most challenging part of providing a complete description of a Web service is in communicating its semantic qualities.
- Examples of service semantics include:
    - how a service behaves under certain conditions
    - how a service will respond to a specific condition
    - what specific tasks the service is most suited for

- service semantics are assessed by humans either verbally by discussing the qualities of a service with its owner, or by
- reading supplementary documentation published alongside service descriptions.
- Goal is to provide sufficient semantic information in a structured manner so that, in some cases, service requestors can go to evaluate and choose suitable service providers independently.
- It is importance when dealing with external service providers
- But even within organizational boundaries, semantic characteristics tend to take on greater relevance as the amount of internal Web services grows.

5.3.6. Service description advertisement and discovery

- Amount of services increases within and outside of organizations, mechanisms for advertising and discovering service descriptions may become necessary.
- Central directories and registries become an option to keep track of the many service descriptions that become available.
- These repositories allow humans (and even service requestors) to:
    - locate the latest versions of known service descriptions
    - discover new Web services that meet certain criteria

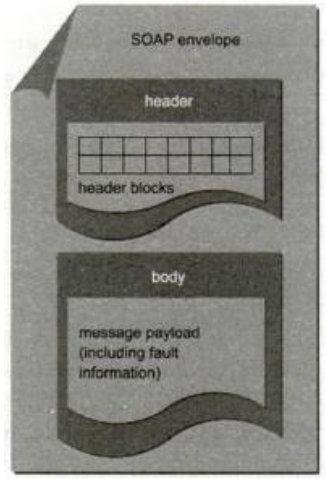
6) Explain SOAP message frameworks and SOAP nodes

5.4. Messaging (with SOAP)

- All communication between services is message-based,
- The messaging framework chosen must be standardized so that all services, regardless of origin, use the same format and transport protocol.
- Message-centric application design that an increasing amount of business and application logic is embedded into messages.
- The SOAP specification was chosen to meet all of these requirements
- Universally accepted as the standard transport protocol for messages processed by Web services

5.4.1. Messages

- Simple Object Access *Protocol*, the SOAP specification's main purpose is to define a standard message format.
- The structure of this format is quite simple, but its ability to be extended and customized
- **Envelope, header, and body**
- Every SOAP message is packaged into a container known as an *envelope*.
- Much like the metaphor this conjures up, the envelope is responsible for housing all parts of the message



gure 5.21
e basic structure of a SOAP message.

- Each message can contain a *header*, an area dedicated to hosting meta information.
- Service-oriented solutions, this header section important
- The actual message contents consists of XML formatted data.

- The contents of a message body are often referred to as the message *payload*.

Header blocks

- SOAP communications framework used by SOAs, the creating messages that are intelligence-heavy and self-sufficient
- Independence that increases the robustness and extensibility
- Message independence is implemented through the use of *header blocks*
- packets of supplementary meta information stored in the envelope's header area.
- It further reinforces the characteristics of contemporary SOA related to fostering reuse, interoperability, and composability.
- Examples of the types of features a message can be outfitted with using header blocks include:
    - processing instructions that may be executed by service intermediaries or the ultimate receiver
    - routing or workflow information associated with the message
    - security measures implemented in the message
    - reliability rules related to the delivery of the message
    - context and transaction management information
    - correlation information

Message styles

- The SOAP specification was originally designed to replace proprietary RPC protocols
- Distributed components to be serialized into XML documents, transported, and then deserialized into the native component format upon arrival.

Two types of Message styles

1. *RPC-style* message runs contrary to the emphasis SOA places on independent, intelligence-heavy messages.
2. SOA relies on *document-style* messages to enable larger payloads, coarser interface operations, and reduced message transmission volumes between services.

Attachments

- To facilitate requirements for the delivery of data not so easily formatted into an XML document, the use o*Sf OAP attachment* technologies exist.
- Each provides a different encoding mechanism used to bundle data in its native format with a SOAP message.
- SOAP attachments are commonly employed to transport binary files, such as images.

Faults

- SOAP messages offer the ability to add exception handling logic by providing an optiona*fla ult* section
- It resides within the body area.
- The typical use for this section is to store a simple message used to deliver error condition information when an exception occurs.

5.4.2. Nodes

The programs that services use to transmit and receive SOAP messages are referred to as *SOAP nodes.*

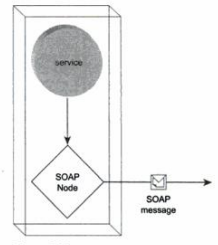


Figure 5.23
A SOAP node transmitting a SOAP message received by the service logic.

- Regardless of how they are implemented, SOAP nodes must conform to the processing standard set forth in the versions of the SOAP specification they support.

- Vendor-neutral communications framework upon which SOA is based on the SOAP node.
- It is what guarantees that a SOAP message sent by the SOAP node from service A can be received and processed by a SOAP node from any other service.

Node types

- *SOAP nodes* are given labels that identify their type, depending on what form of processing they are involved with in a given message processing scenario.
- Below is a list of type labels associated with SOAP nodes
- The SOAP specification has a different use for the term "role" and instead refers to these SOAP types or labels as *concepts*.
    - *SOAP sender*a SOAP node that transmits a message
    - *SOAP receiver*a SOAP node that receives a message
    - *SOAP intermediary*a SOAP node that receives and transmits a message, and optionally processes the message prior to transmission
    - *initial SOAP sender*the first SOAP node to transmit a message
    - *ultimate SOAP receiver*the last SOAP node to receive a message

SOAP intermediaries

Service intermediaries transition through service provider and service requestor roles, *SOAP intermediary* nodes move through SOAP receiver and SOAP sender types when processing a message

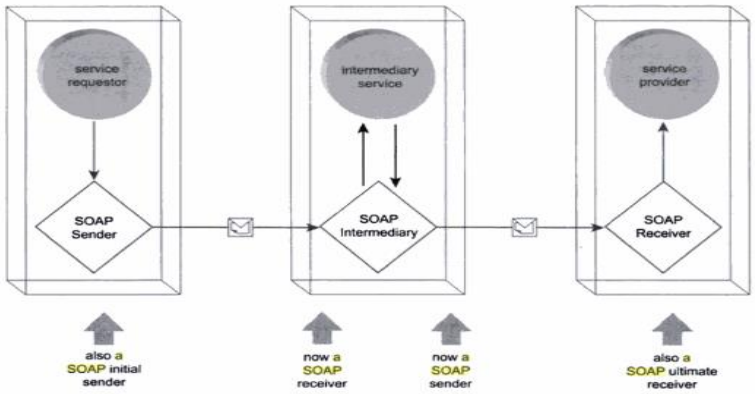


Figure 5.25
Different types of SOAP nodes involved with processing a message.

SOAP nodes acting as intermediaries can be classified as forwarding or active.
When a SOAP node acts as a *forwarding intermediary*, it is responsible for relaying the contents of a message to a subsequent SOAP node. In doing so, the intermediary will often process and alter header block information relating to the forwarding logic it is executing.

Active intermediary nodes are distinguished by the type of processing they perform above and beyond forwarding-related functions. An active intermediary is not required to limit its processing logic to the rules and instructions provided in the header blocks of a message it receives. It can alter existing header blocks, insert new ones, and execute a variety of supporting actions.

7. Define MEP? Discuss the various types of MEPS with neat diagram

Message exchange patterns
Every task automated by a Web service can differ in both the nature of the application logic being executed and the role played by the service in the overall execution of the business task. Regardless of how complex a task is, almost all require the transmission of multiple messages. The challenge lies in coordinating these messages in a particular sequence so that the individual actions performed by the message are executed properly and in alignment with the overall business task .

The fundamental characteristic of the fire-and-forget pattern is that a response to a transmitted message is not expected.

Message exchange patterns (MEPs) represent a set of templates that provide a group of already mapped out sequences for the exchange of messages. The most common example is a request and response pattern. Here the MEP states that upon successful delivery of a message from one service to another, the receiving service responds with a message back to the initial requestor.
Many MEPs have been developed, each addressing a common message exchange requirement. It is useful to have a basic understanding of some of the more important MEPs, as you will no doubt be finding yourself applying MEPs to specific communication requirements when designing service-oriented solutions.

Primitive MEPs

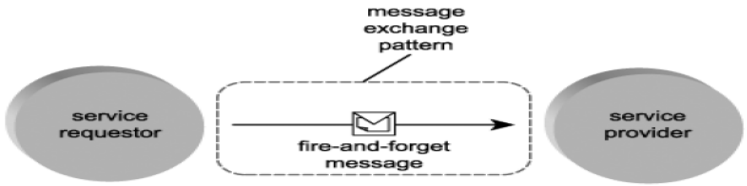
Before the arrival of contemporary SOA, messaging frameworks were already well used by various messaging-oriented middleware products. As a result, a common set of primitive MEPs has been in existence for some time.

Request-response

This is the most popular MEP in use among distributed application environments and the one pattern that defines synchronous communication (although this pattern also can be applied asynchronously).
The request-response MEP establishes a simple exchange in which a message is first transmitted from a source (service requestor) to a destination (service provider). Upon receiving the message, the destination (service provider) then responds with a message back to the source (service requestor).

Fire-and-forget

This simple asynchronous pattern is based on the unidirectional transmission of messages from a source to one or more destinations .
A number of variations of the fire-and-forget MEP exist, including:
The single-destination pattern, where a source sends a message to one destination only.
The multi-cast pattern, where a source sends messages to a predefined set of destinations.
The broadcast pattern, which is similar to the multi-cast pattern, except that the message is sent out to a broader range of recipient destinations.



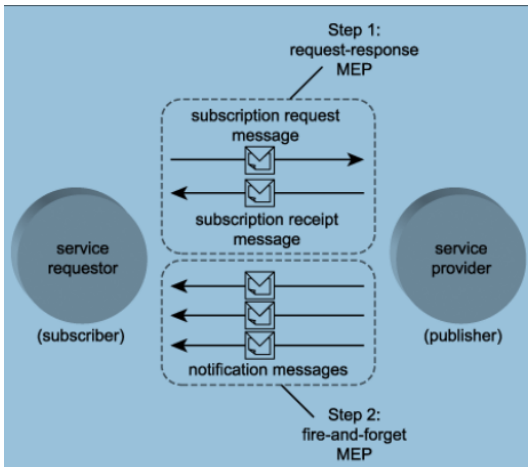
Complex MEPs

Even though a message exchange pattern can facilitate the execution of a simple task, it is really more of a building block intended for composition into larger patterns. Primitive MEPs can be assembled in various configurations to create different types of messaging models, sometimes called complex MEPs.

The **publish-and-subscribe pattern** introduces new roles for the services invo exchange. They now become publishers and subscribers, and each may be involved receipt of messages. This asynchronous MEP accommodates a requirement for messages available to a number of subscribers interested in receiving them.

Step 1 in the publish-and-subscribe MEP could be implemented by a request-res subscriber's request message, indicating that it wants to subscribe to a topic, is resp from the publisher, confirming that the subscription succeeded or failed.

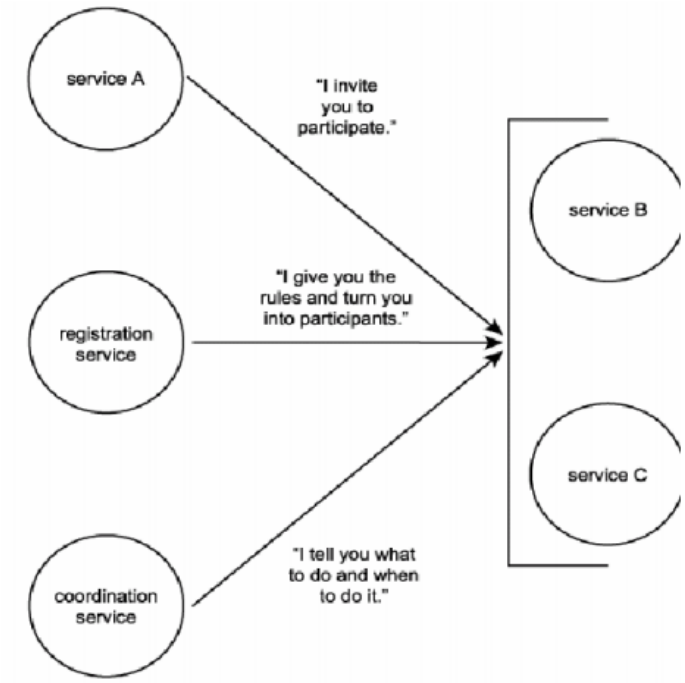
Step 2 then could be supported by one of the fire-and-forget patterns, allowing the series of unidirectional messages to subscribers



8. Explain elaborately the concept of coordination related to WS with neat diagram

- Every activity introduces a level of context into an application runtime environment. Something that is happening or executing has meaning during its lifetime, and the description of its meaning (and other characteristics that relate to its existence) can be classified as context information.
- The more complex an activity, the more context information it tends to bring with it. The complexity of an activity can relate to a number of factors, including:
  - the amount of services that participate in the activity
  - the duration of the activity
  - the frequency with which the nature of the activity changes – whether or not multiple instances of the activity can concurrently exist
- A framework is required to provide a means for context information in complex activities to be managed, preserved and/or updated, and distributed to activity participants. Coordination establishes such a framework.

Coordination provides services that introduce controlled structure into activities



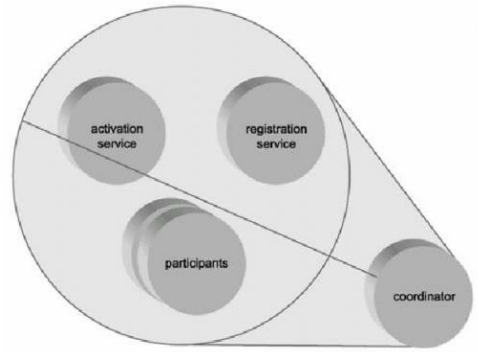
Coordinator composition

WS-Coordination establishes a framework that introduces a generic service based on the coordinator service model. This service controls a composition of three other services that each play a specific part in the management of context data

The coordinator composition consists of the following services:

- **Activation service** Responsible for the creation of a new context and for associating this context to a particular activity.
- **Registration service** Allows participating services to use context information received from the activation service to register for a supported context protocol.

The coordinator service composition



- **Protocol-specific services** These services represent the protocols supported by the coordinator's coordination type.
- **Coordinator** The controller service of this composition, also known as the coordination service.

Coordination types and coordination protocols

- Each coordinator is based on a coordination type, which specifies the nature and underlying logic of an activity for which context information is being managed.
- Coordination types are specified in separate specifications.
- The WS-Coordination framework is extensible and can be utilized by different coordination types, including custom variations.
- However, the two coordination types most commonly associated with WSCoordination are
  - WS-AtomicTransaction and
  - WS-BusinessActivity.
- Coordination type extensions provide a set of coordination protocols, which represent unique variations of coordination types and consist of a collection of specific behaviors and rules.
- A protocol is best viewed as a set of rules that are imposed on activities and which all registered participants must follow.

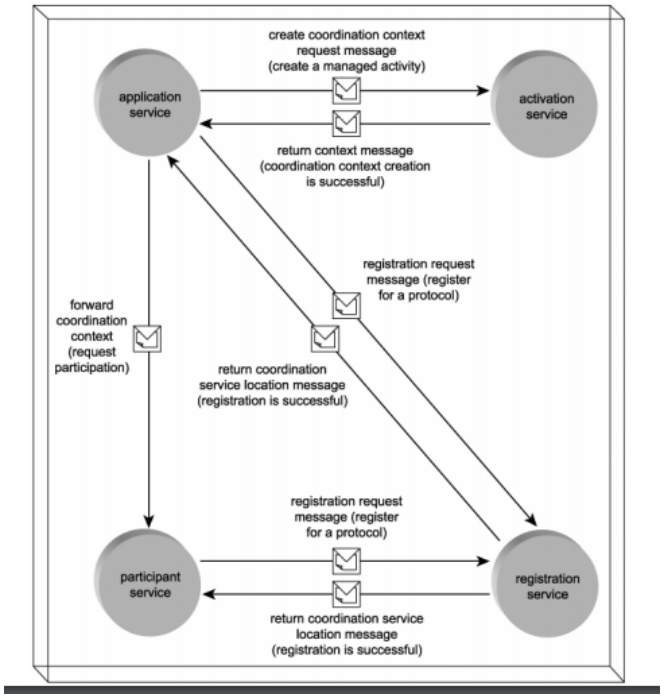
Coordination contexts and coordination participants

- A context created by the activation service is referred to as a coordination context. It contains a collection of information that represents the activity and various supplementary data.
- Examples of the type of data held within a coordination context include:
  - a unique identifier that represents the activity
  - an expiration value
  - coordination type information
- A service that wants to take part in an activity managed by WSCoordination must request the coordination context from the activation service. It then can use this context information to register for one or more coordination protocols.
- A service that has received a context and has completed registration is considered a participant in the coordinated activity.

The activation and registration process

- The coordination service composition is instantiated when an application service contacts the activation service as given in the next figure. ia a CreateCoordinationContext request message, it asks the activation service to generate a set of new context data.
- Once passed back with the ReturnContext message, the application service now can invite other services to participate in the coordination. This invitation consists of the context information the application service originally received from the activation service.

The WS-Coordination registration process

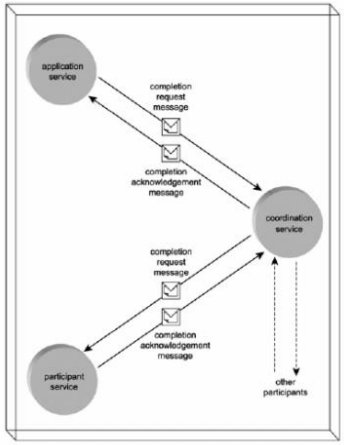


- Any Web service in possession of this context information may issue a registration request to the registration service.
- This allows the service to enlist in a coordination based on a specific protocol. (Protocols are provided by separate specifications and are discussed later on as part of the Atomic transaction and Business activities sections.)
- Upon a successful registration, a service is officially a participant.
- The registration service passes the service the location of the coordinator service, with which all participants are required to interact.
- At this time, the coordination service is also sent the address of the new participant.

The completion process

The application service can request that a coordination be completed by issuing a completion request message to the coordination service.
The coordinator, in turn, then issues its own completion request messages to all coordination participants.

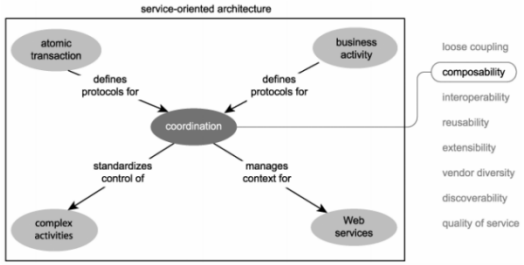
Each participant service responds with a completion acknowledgement message



Coordination and SOA

- A coordinator-based context management framework, as provided by WS-Coordination and its supporting coordination types, introduces a layer of composition control to SOAs.
- It standardizes the management and interchange of context information within a variety of key business protocols.
- Coordination also alleviates the need for services to retain state.
- Statelessness is a key service-orientation principle applied to services for use within SOAs.
- Coordination reinforces this quality by assuming responsibility for the

Coordination as it relates to other parts of SOA



- management of context information.