Choreography and Orchestration using Business Process Execution Language for SOA with Web Services

Aarti Karande[1], Milind Karande[2] and B.B.Meshram[3]

**[1] Computer Department Mumbai University,
Mumbai, India**

**[2] Systime.net CMS
Mumbai, India**

**[3] Computer Department Mumbai University,
Mumbai, India**

Abstract

Web services using XML provide information to applications through an application oriented interface, so that it can be parsed and processed easily rather than being formatted for display. Web services combine the advantages of the component-oriented methods and web techniques. An orchestration process presents different services which can be composed efficiently through a low in order to execute a business process. Application and business services can be freely designed to be processagnostic and reusable. Choreography describes multiparty collaboration and focuses on message exchange; each Web service involved in a choreography knows exactly when to execute its operations and with whom to interact. Choreography depends on the orchestration using the functional, resource dependencies among multiple services. Workflow services enable to interleave human interaction with connectivity to system and services within an end-to-end process flow. An orchestration process presents different services which can be composed efficiently through a flow in order to execute a business process. Business Process Execution Language (BPEL) defines a notation for specifying business process behavior based on Web services. BPEL for Web services is an XML-based language designed to enable task-sharing for a distributed computing or grid computing environment across multiple organizations using a combination of Web services. SOA allows the integration of existing systems, applications and users into a flexible architecture that can easily accommodate changing needs. BPEL Business Processes offer the possibility to aggregate web services and define the business logic between each of these service interactions that is BPEL orchestrates such web service interactions. Each service interaction can be regarded as a communication with a business partner. SOA allows the integration of existing systems, applications and users into a flexible architecture that can easily accommodate changing needs. Integrated design, reuse of existing IT investments and above all, industry standards are the elements needed to create a robust SOA.

Keywords: **Web Service, Orchestration, Choreography, Service Oriented Architecture.**

1. Introduction

A. **Web services**

Using application oriented interface Web services provide information to *applications* rather than to humans. This result in the structured information using XML, so that it can be parsed and processed easily rather than being formatted for display. A client and a service that support common communication protocols can interact regardless of the platforms on which they run using Web services. Web Services publish details of their functions and interfaces, but they keep their implementation details private. This makes Web services particularly applicable to a distributed heterogeneous environment [1]

1. The key specifications used by Web services:

• eXtensible Markup Language -a markup language for formatting, exchanging structured data. [12]
• SOAP (Simple Object Access Protocol)—an XML based protocol specifying envelope information, contents and processing information for a message.
• WSDL (Web Services Description Language)—an XMLbased language used to describe the attributes, interfaces and other properties of a Web service. A WSDL document can be read by a potential client to learn about the service.

2. Agents and Services: A Web service can be implemented by a concrete agent which is the concrete piece of software or hardware that sends and receives messages, while the service is the resource characterized by the abstract set of functionality that is provided.

3. Requesters and Providers: Web service will use a requester agent to exchange messages with the provider entity's provider agent.

4. Service Description: A Web service description written in WSDL, defines the message formats, data types, transport protocols, and transport serialization formats that should be used between the requester agent and the provider agent. It also specifies one or more network locations at which a provider agent can be invoked, and may provide some information about the message exchange pattern.

5. Semantics: The semantics of a Web service is the shared expectation about the behavior of the service. this is the contract between the requester entity and the provider entity regarding the purpose and consequences of the interaction.

B. **Working structure of a Web Service**

1) The requester and provider entities become known to each other
2) The requester and provider entities somehow agree on the service description and semantics that will govern the interaction between the requester and provider agents
3) The service description and semantics are realized by the requester and provider agents
4) The requester and provider agents exchange messages, thus performing some task on behalf of the requester and provider entities.

C. **Web services properties**

• Discoverable: for serving to other users so new services has to be discovered and accessed by consumers [2]
• Communicable: This is often asynchronous messaging as opposed to synchronous messaging.
• Conversational: A conversation involves sending and receiving documents in a context. This may be complex interactions between Web services and entails multiple steps of communication that are related to each other.
• Secure and Manageable: Security, manageability, availability, and fault tolerance are critical for a commercial web-service.

D. **A Web Services Description Language**

WSDL document is an XML document that describes Web services that are accessible over a network. WSDL describes the following [13]
• Logic the Web service performs with the location of the Web service
• Method to use to access the Web service, including the protocol that the Web service
• Consumer must use to invoke the Web service and receive a response along with input parameters that the Web service consumer must supply to the Web service and the output parameters that the Web service returns.

E. **Uses of Web Services**

• **Remote Procedure Calls:** RPC Web services present a distributed function call interface, which is familiar to many developers. Typically, the basic unit is the WSDL operation.[3]
• **Service-oriented architecture:** Under SOA Web services are used to implement an architecture in which the basic unit of communication is a message, rather than an operation. This is often referred to as "message-oriented" services.
• **Representational state transfer:** REST Web Services uses WSDL to describe SOAP messaging over HTTP, which defines the operations. REST describes operations, can be implemented as an abstraction purely on top of SOAP or can be created without using SOAP at all.
In Reusable application-components uses Web Services offers most frequently used services like currency conversion, weather reports, language translation etc.

II Orchestration and Choreography

A. Service Orchestration

For web based solution, the business logic must be distributed into different services. These services must be organized and composed, which is commonly referred to as an orchestration. An orchestration process presents different services which can be composed efficiently through a flow in order to execute a business process.

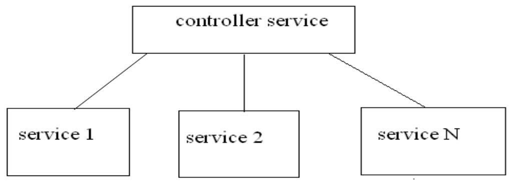


Figure1. Service Orchestration

In figure 1, the different services involved are coordinated through a controller service. The resulting service can be integrated hierarchically into another composition by means of its WSDL interface [4], [5].

Web services technologies define a standard mechanism to publish and consume data and application logic using IP such as the HTTP, or the SMTP. The orchestration service layer provides a powerful means by which contemporary service-oriented solutions can realize some key benefits [6]. E.g. by abstracting business process logic means:
• Application and business services can be freely designed to be process-agnostic and reusable.
• The process service assumes a greater degree of statefulness, thus further freeing other services from having to manage state.
• The business process logic is centralized in one location, as opposed to being distributed across and embedded within multiple services.

B. Choreography

At design-time choreography ensures interoperability between a set of peer services from a global perspective, meaning that all participating services are treated equally, in a peer-to-peer fashion. A choreography model describes multiparty collaboration and focuses on message exchange; each Web service involved in a choreography knows exactly when to execute its operations and with whom to interact.

C. Web Service Stack layer representation

On top of the Web Service stack, there are two other elements to be found: aggregation and choreography. These are two different ways to create composite services, the middle layer of the extended stack [8] make the distinction between constrained services and unconstrained ways of web service aggregation.

Constrained aggregation is coordinating services by means of a process-flow, and can realize by means of two technologies: WS-orchestration and WS-choreography. Figure 2 shows the stack view of the web services showing the layer representation. Using the layer representation relationship between the aggregation (Orchestration) and choreography can be analyzed. An executable process specifies the execution order between a number of constituent activities, the partners involved, the messages exchanged between these partners, and the fault and exception handling mechanisms. Coordinating Web Services by means of abstract business processes is Web Service Choreography, and coordinating them by means of executable business processes is Web Service Orchestration.

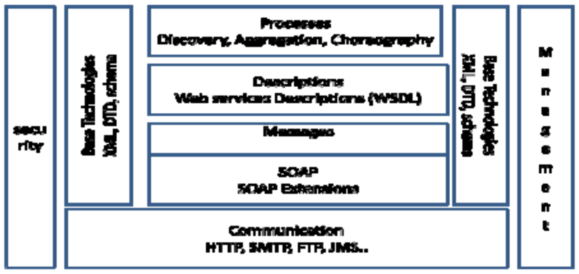


Figure 2: Web Services Stack [7]

WS-Choreography specifies the possible sequences between the messages that can be exchanged between different agencies. A message consisting of a request for advice has to be answered with a response message containing the advice.

WS-Orchestration requires that the whole process that is needed for offering the service is defined at one place. This is an executable process, and offers an overview of all the activities that are to be performed in the process. These are internal activities within the agency executing the process, as well as activities at other agencies that are provided as a Web Service.

D. Difference between Orchestration and Choreography

• Orchestration describes a process flow between services from the perspective of one participant (centralized control), whereas choreography tracks a sequence of messages involving multiple parties (decentralized control, no central server), where no one party truly owns the conversation [9]

• Orchestration a bottom-up approach to design. Whereas choreography a top-down approach to design

• When multiple processes interact with each other, orchestrations describe the point of view of a single process only. When multiple processes interact with each other, choreographies describe the interplay between processes from a global perspective.

• Orchestration is well understood, whereas choreography an open research field

E. Orchestration depends on Choreography

After this overview over Web service choreography and orchestration and the main concerns how orchestration and choreography address. This can be distinguished using three main dimensions: structural, functional, and resource dependencies. [10]

a. **Structural Dependencies:** Structural dependencies are those driving the overall structure or organization of a process definition, and concern with involved activities, conditions, ramifications within the process flow, and so on. Starting from an overall activity diagram, the authors first extract the role-specific view of the process and then refine it in order to reach a granularity level where the single activities of the remaining diagram reflects the single service invocations required for achieving the specific functionality.

b. **Functional Dependencies:** Dependencies arise, whenever the functionalities they provide are used within a process specification and the composition language "delegates" the relative competencies to the underlying coordination protocols.

c. **Resource Dependencies:** Resources executing a specific work item are provided with the exact amount of data that is required for the correct execution of that task. Involved resources do not have a task-surviving behavior with constraints affecting the overall process definition. Composite service designers must know about the coordination requirements of the services they use and take them into account when defining composite services.

F. Working of Choreography with Orchestration

The design processes and execution infrastructure for service choreography models are inherently more complex than orchestration; decentralized control brings a new set of challenges which are the result of message passing between distributed asynchronous, concurrent processes. However, although more complex, there are a number of arguments for adopting choreography. [11] From a software design perspective, orchestration is suitable when the goal is to build individual services or to service enable existing applications. However during the early phases of service design the emphasis lies not on building individual services but rather on how groups of services work together, by identifying collections of potential services and understanding and analyzing their interactions; at this early stage in the design process engineers require a global view of how Web services interact with one another; choreography provides just these tools and is a description of multi-party collaboration.

Choreography is an unambiguous way of describing the relationships between services in a global peer-to-peer collaboration, without requiring orchestration at all. Each party takes an equal, predefined and pre-agreed role in the choreography, this removes the scenario where businesses are interacting over a shared task but one organization has control over another by orchestrating their services. Centralized control through an orchestration engine is a valid solution for scenarios found in e-Commerce, where relatively small quantities of intermediate are moved between services in a workflow. However, centralized servers make less sense when dealing with data centric workflows, common to scientific applications. Passing large quantities of intermediate data through a centralized orchestration engine results in unnecessary data transfer and wasted bandwidth, overloading the engine and thereby decreasing the performance of a workflow.

G. Workflow Services

Workflow services includes following terms for the task
• Task – Work needs to be done by user, role or Group
• Notification – An email, voice, fax, SMS that is send when user is assigned as task or informed that the status of the task has changed
• Work list – an enumeration of task or work item assigned to or of interest to a user
• Human Task Editor – A tool that enables to specify task setting such as task outcome, payload structure, notification setting and so on.
• task file – The metadata task configuration file stores task setting specified with the Human Task Editor
• Routing slip – contains information about flow pattern for the Workflow, assignee, escalation policy, expiration duration, sequence in which participants interact in task.

Features of Workflow services

=>Work Queues
->Standard work queues – high priority task
->Custom work queues – user can define new work queues based on specific search criteria
->Proxy work queues – can grant access to other users to select work queues.

=>Rules integration
->User rules - Can define custom delegation auto approval
->Group Rules – can define auto assignment of rules for Roles or Groups Task Assignment and routing
->This includes creating task from business to business process and assigning task to user or role
->Support for task expiration and automatic renewal
->Support for task delegation, approval and escalation
->Creating custom task escalation rule functions
->Override and restrict default system action
-> Jsp based form for viewing and updating task details

H. Business Process Execution Language

Business Process Execution Language (BPEL) defines a notation for specifying business process behavior based on Web services. BPEL is used to model the behavior of both executable and abstract processes. (Executable processes model actual behavior in business transactions. Abstract processes interact without revealing their internal behavior.)[9] The scope of what BPEL includes is:
• The sequencing of process activities, especially Web service interactions.
• Correlation of messages and process instances.
• Recovery behavior in case of failures and exceptional conditions.
• Bilateral Web service-based relationships between process roles.

H.1. Structure of a BPEL Process

A business process described using BPEL starts with a <process> tag. This tag defines the XML namespaces used during the description and establishes a set of attributes.

```
<process name = NOName” targetNamespace=”anyURL”
queryLanguage =”anyURL”?
expressionLanguage=”anyURL”?
xmlns=http://docs.casispen.org/wsbpel/proces/executable>
<import:namespace=”anyURL” Loation=anyURL’
importType=”anyURL” />
<partnerlinks>…</partnerlinks>
<variable>…</variable>
<correlationSets>…</correlationSets>
Activity
</process>
```

• <process> element: It is assigned a name value using the name attribute and is used to establish the process definition-related namespaces. [8]

• <partnerLink> element establishes the port type of the service (partner) that will be participating during the execution of the business process.
• <variables> Entire messages and data sets formatted as XSD schema types can be placed into a variable and retrieved later during the course of the process.
• <partnerLinks> and <correlationSets> declare a set of items with some characteristics as: they are identified by a name, a type and a value; their scope is the whole process definition, in the same way global variables are used in general purpose programming languages; and BPEL allows executing parallel flows, so the access to these elements must be exclusive.

H.2. Relationship to Business Partners

BPEL Business Processes offer the possibility to aggregate web services and define the business logic between each of these service interactions that is BPEL orchestrates such web service interactions. Each service interaction can be regarded as a communication with a business partner.
As Shown in the figure 3 partnerlinks are connected to the process. The interaction is described with the help of partner links. Partner links are instances of typed connectors which specify the WSDL port types the process offers to and requires from a partner at the other end of the partner link

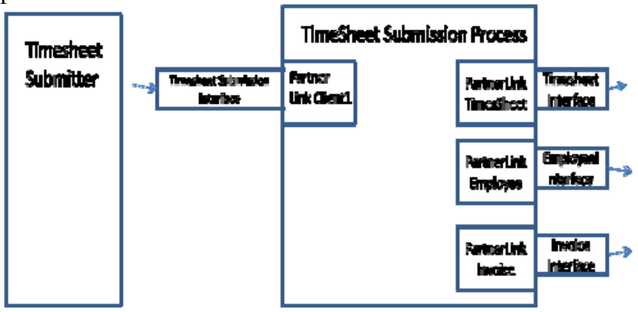


Fig3 . Example showing BPEL linking.

H.3. Behavior of a BPEL Process

BPEL processes are activities of two types. First are structured activities which can contain other activities and define the business logic between them. In contrast, other one is basic activities only performs their intended purpose and has no means to define any other logic by containing other activities.

a. Providing and Consuming Web Services

Activities with the purpose of consuming messages from (receive activity) and providing messages to web service partners (reply activity).These activities allow exchanging messages with external partners (services).

b. Structuring the Process Logic

BPEL provides means to structure the business logic according to needs. The sequence activity is used to define a collection of activities which are executed sequentially in lexical order. Another activity used for structuring the business logic is the if-else activity. BPEL offers three activities that allow the repeated execution of a piece of business logic like while activity, repeatUntil activity, forEach activity.

c. Parallel Processing

BPEL offers besides the flow activity, the parallel variant of forEach activity and event handlers allow multiple instances of its child activity to run in parallel. Adding a link introduces a control dependency which means that the activity which is the target of the link will only be executed if the activity that is the source of the link has completed. Join conditions are associated with activities, usually if the activities have any incoming links. A joinCondition specifies for an activity something like a "start condition", e.g. all incoming links must have the status of true in order for the activity to execute, or at least an incoming link must have the status true.

d. Data Manipulation

BPEL offers the assign activity which contains one or more copy operations. Each copy operation has a from–spec and a to–spec, indicating the source and target elements where data gets copied from and to, respectively. One of the goals of BPEL was not to reinvent yet another XML-based data manipulation language, but to reuse other standard like XPath and XSLT.

e. Exception Handling

BPEL offers the concept of fault handlers which can be attached to a scope a process even directly as a shorthand notation on an invoke activity. A fault handler gets installed as soon as the scope, it is associated to, gets started. As an example, the process level fault handler gets installed when the process starts. If the process completes normally, the installed fault handler then gets discarded, but if a fault situation occurs, that fault gets propagated to the fault handler.

f. Selective Event Processing

There is a set of multiple suitable messages where each one of them can trigger subsequent steps in the business process.

g. Multiple Event Processing

More than one message is needed to trigger subsequent steps. BPEL offers number of the event performing the processing which are interrelated to each other.

h. Concurrent Event Processing

Event handlers are associated with the whole process or a scope. They are enabled when their associated scope is initialized and disabled when their associated scope terminates. When enabled, any number of events may occur. They are processed in parallel to the scope's primary activity and in parallel to each other. Message events also represent Web services operations exposed by a process and are modeled as onEvent elements. Timer events are modeled as onAlarm elements, similar to pick activities. In event handlers, timer events can be processed multiple times.

i. Message Correlation

If a Web service request message does not lead to the creation of a new process instance, how does it "find" the running process instance it is designated for? This mechanism may also be used to identify process instances but BPEL does not mandate this approach. BPEL provides a portable correlation mechanism called *correlation sets*. The major observation behind this concept is the fact that most messages exchanged between business processes and the outside world already carry the key data required to uniquely identify a process instance.

j. Concurrent Message Exchanges

Each inbound message activity, that is, receive, onMessage, or onEvent, may have an associated reply activity in order to implement a WSDL request-response operation. If a process has multiple receive and reply activities that point to the same partner link and WSDL operation then the association between individual receive and reply activities is ambiguous. In this case, you use the messageExchange attribute to establish the relationship.

k. More Parallel Processing

In addition to sequential loops, there is a variation of the forEach activity. Instead of performing each loop iteration in a sequence, all loop iterations are started at the same time and processed in parallel. Besides the flow activity and event handlers, this is the third form of parallel processing in BPEL.

H.4. Securing a BPEL process with an example

Consider the BPEL process as a web service standardized way. Next, create a new WS-Policy set that will require the client to provide a WS-Security header with UsernameToken containing the username and password for user authentication. We will attach this WS-Policy set to our BPEL process's web service export to protect the process. Only authenticated users will have access to the BPEL process. We will then test the example, with and without providing credentials, to see if security is working. Next, we will see that authentication information (user's identity) is not automatically propagated to the BPEL process. So, the process does not know which user called it. We will extract a user's identity from UsernameToken and propagate this identity to the BPEL process. Through another round of testing, we will see that the user who called the process will become the process instance owner. The final step in this example will be to configure the BPEL process in a way that only authenticated users, who are authorized, will be able to call it. To achieve this, we will have to define the security permission qualifier on our BPEL process, define a role that will have access to our BPEL process, and map users to this role to actually allow specific users to access our BPEL process. Later on, we will be able to add or remove users from the role to enable runtime access permission update for specific users.

H.5. Business Process Execution Language for Web Services (BPEL4WS)

The standard language for WSO is BPEL4WS, Business Process Execution Language for Web Services [2], or BPEL for short. BPEL is developed by Microsoft, IBM, and BEA, and unifies two older languages from Microsoft and IBM: XLANG and WSFL. The BPEL specification also contains a part to specify abstract business processes (business protocols). BPEL can be viewed as a layer on top of WSDL, as it uses the properties of WS that are specified by means of WSDL.

A process that is specified in BPEL consists of two types of activities: basic activities, such as receive, reply, wait, and structured activities as switch, while, and sequence. The structured activities determine the structure, of the sequencing of the process, and the basic activities determine what happens in the process, for example the invocation of a WS, receiving a message from a WS, etc. The WS that are invoked are defined in BPEL as partnerlinks. BPEL also specifies the messagetypes of the messages that are sent to the invoked WS. These messagetypes must correspond with the messagetypes specified in the WSDL of the invoked WS. Data can be use at different places in the process by means of variables. The contents of a received message can be copied to a certain variable, and be used later on in the process when another WS is invoked. BPEL also offers functionality for error- and exception handling.

H.6. Workflow Management

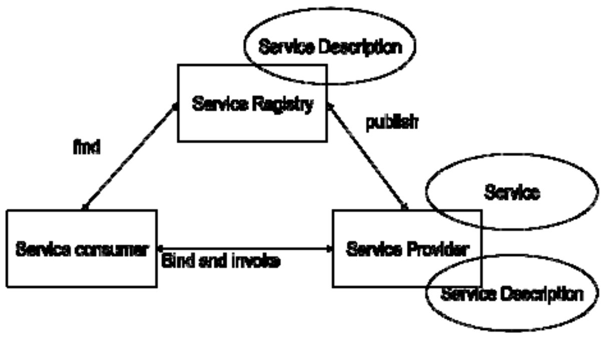
The Workflow Management Coalition (WFMC), a nonprofit, international organization of workflow vendors, users, analysts and university/research groups defines a workflow as: the automation of a business process, in whole or in part, during which documents, information, or tasks are passed from one participant to another, according to a set of procedural rules [21]. The goal of workflow is the automation of complete processes instead of isolated tasks [39]. Workflow traditionally concerned itself with passing work from one employee to another, assuring the latter that he could start his task. As workflow matured, we saw a shift in focus towards the automation of the whole process itself [1].

A process consists of activities that can be performed either manually or automated. A workflow can be executed by a Workflow Management System (WFMS). A WFMS is a system that defines, creates, and manages the execution of workflows through the use of software running on one or more workflow engines, which is able to interpret the process definition, interact with workflow participants and, where required, invokes the use of IT-tools and applications. WFMC was founded, with as main goal: to promote and develop the use of workflow through the establishment of standards for software terminology, interoperability and connectivity between workflow products [20].

III Service Oriented Architecture

SOA allows the integration of existing systems, applications and users into a flexible architecture that can easily accommodate changing needs. Integrated design, reuse of existing IT investments and above all, industry standards are the elements needed to create a robust SOA. SOA is a well-established method of designing, deploying and managing the individual activities that make up a business process. At the software level, use of SOA entails building components with published meta-data that defines both the information they require from each other and the services that they provide. This allows software tools to help rapidly orchestrate SOA services into new processes as the needs of the organization change. Loose coupling helps preserve the future by allowing parts to change at their own pace without the risks linked to costly migrations using monolithic approaches. SOA allows business users to focus on business problems at hand without worrying about technical constraints.

Data within a SOA generally falls into categories:
• **Service state** - This pertains the current state of the business process or service, i.e. which processes are active vs. closed vs. aborted, and so on.
• **Service Result** - This data is delivered by the business process/data service as a result of the execution.



•
Fig 4 . SOA basic architecture specifying component.

A. Principles of SOA

- Reuse, granularity, modularity, compos ability, componentization and interoperability and Standards-compliance[12].
- Services identification and categorization, provisioning and delivery, and monitoring and tracking.

B. Component Role in SOA

• **Service Provider**: The service provider creates a web service and publishes its interface and access information to the service registry. Service provider must decide which services to expose, how to make trade-offs between security of various services. With finding easy availability of services, it also decides how to price the services. Provider also check how/whether to exploit services for other value along with category the service should be listed in for a given broker service along with trading partner agreements required to use the service.

• **Service consumer**: The service consumer or web service client locates entries in the broker registry using various find operations and then binds to the service provider in order to invoke one of its web services.

• **Service Registry:** This is the centralize dictionary of the entire service list. It registers what services are available with service provider, and lists all the potential service recipients. Before providing the service to consumer from provider, registry validates available services in the service registry.

C. Advantages of using SOA

->Reduction in development time and cost: SOA services are builded using existing services to form composite applications saving time and cost.
->Lower maintenance cost: Reusable services reduce the number and internal complexity of enterprise services.
->High-quality services: Increased service reuse creates high-quality services through multiple testing cycles from different service consumers.
->Lower integration costs: Standardized services know how to work together, enabling disparate applications to quickly and easily connect.
->Reduce risk: Fewer, reusable services provide greater control over corporate and IT governance policies, and reduce the overall compliance risk to an enterprise.

D. Steps to deploy SOA [5]

->Service enablement: each discrete application needs to be exposed as a service.
->Service orchestration: Distributed services need to be configured and orchestrated in a unified and clearly defined distributed process.

->Deployment: Emphasis should be shifted from test to the production environment, addressing security, reliability, and scalability concerns.
->Management: Services must be audited, maintained and reconfigured. The latter requirements require that corresponding changes in processes must be made without rewriting the services or underlying application.

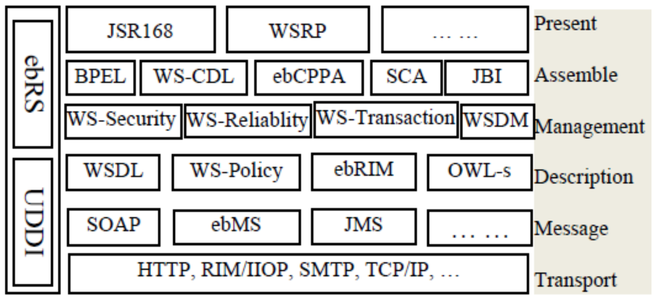


Figure 5. The SOA protocol stacks

As shown in the fig. 5 The SOA protocol stacks can be divided into seven levels by the standard functions in SOA. It shows as Fig. 1. From the bottom to top, they include transport lay, message lay, description lay, management lay, assembling lay, present and service finding register lay. The most of them have been used in the SOA applications except of ebXML and other ebusiness standards. ebRS UDDI[23]

D. SOA for the solution on the integration

• Business analysts focus on higher order responsibilities in the development lifecycle while increasing their own knowledge of the business domain. [2]
• Separating functionality into component-based services that can be tackled by multiple teams enables parallel development.
• Quality assurance and unit testing become more efficient; errors can be detected earlier in the development lifecycle
• Development teams can deviate from initial requirements without incurring additional risk
• Components within architecture can aid in becoming reusable assets in order to avoid reinventing the wheel
• Functional decomposition of services and their underlying components with respect to the business process helps preserve the flexibility, future maintainability and eases integration efforts
• Security rules are implemented at the service level and can solve many security considerations within the enterprise.

E. Value of SOA Business Process Choreography

• Increased automation and deeper process integration, as made possible by the introduction of a choreography component, lead to reduced manual effort and rework both in the wholesaler and the customer processes.
• Process and business rule agility leads to product and process changes being implemented faster and cheaper. Legitimate use of telecommunication resources can be ensured more easily, operational efficiency can be improved, and wastage reduced.
• A variety of interaction styles can be provided to customers while still ensuring consistent application of business rules.
• Reuse of shared application components is simplified.
•Standardization allows replacing unsupported components with commercial-off the- shelf software, which promises to simplify deployment and maintenance of the order management application.

F. Web service properties for SOA

->Logical view: The service is an abstracted, logical view of actual programs, databases, business processes, etc., defined in terms of what it does, typically carrying out a business-level operation.
->Message orientation: The service is formally defined in terms of the messages exchanged between provider agents and requester agents.
->Description orientation: The description supports the public nature of the SOA: only those details that are exposed to the public and important for the use of the service should be included in the description. The semantics of a service should be documented, either directly or indirectly, by its description.
->Granularity: Services tend to use a small number of operations with relatively large and complex messages.
->Network orientation: Services tend to be oriented toward use over a network, though this is not an absolute requirement.
->Platform neutral: Messages are sent in a platform-neutral, standardized format delivered through the interfaces. XML is the most obvious format that meets this constraint.[14]

IV Conclusion

Web Services automatic discovery mechanism helps the business to easy find the Service Providers. This also helps customer to find their services easily increasing revenue by exposing their own. Web Services Manager is a security administrator's environment designed to secure access to Web services and monitor activities performed on protected Web services.
Orchestration is an executable business process that interacts with both internal and external Web services. A choreography model describes multiparty collaboration and focuses on message exchange. Workflow services enable to interleave human interaction with connectivity to system and services within an end-to-end process flow. BPEL can be viewed as a layer on top of WSDL, as it uses

the properties of WS that are specified by means of WSDL. SOA is a well-established method of designing, deploying and managing the individual activities that make up a business process. SOA with business process provides standardization. Web service with SOA can be an abstracted view of the actual programs. Process and business rule agility leads to product and process changes being implemented faster and cheaper. Legitimate use of telecommunication resources can be ensured more easily, operational efficiency can be improved, and wastage reduced.

Web Services Manager is a security administrator's environment designed to secure access to Web services and monitor activities performed on protected Web services. Workflow services enable to interleave human interaction with connectivity to system and services within an end-to-end process flow. Most of the process definition languages have inherited their modeling approaches from the field of workflow management.

Structured dependencies among coordination protocols and composition schemas is done by stepwise refining the portion of a process definition relative to only one of the participating services. Orchestration as an executable business process interacts with both internal and external Web services. A choreography model describes multiparty collaboration and focuses on message exchange. XML provides flexibility (its self describing nature is much more forgiving than previous rigid data formats)

V References

[1] The Advantages of Web Service Orchestration in Perspective Jeffrey Gortmaker
[2] Enterprise Architecture and Web Services Dinarle Ortega, Elluz Uzcátegui, María M. Guevara 2009 Fourth International Conference on Internet and Web Applications and Services
[3]Web Services Architecture W3C Working Group Note 11 February 2004.
[4] Alonso, F. Casati, H. Kuno, and V. Machiraju, Web Services: Concepts, Architectures and Applications.Springer,2004.
[5] WSDL specification - W3C. (2001) www.w3.org/TR/wsdl
[6] Service-Oriented Architecture Concepts, Technology, and DesignThomas Erl
[7] Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C.Orchard, D. Web Service Architecture, W3C, 2003.
[8] An Overview of SOA computing by latha srinivasan and jem Treadwell
[9]Mark Colan, “Service-Oriented Architecture expands the vision of Web services”, www-128.ibm.com/developerworks /library/ws-soaintro.html, IBM, U.S., 21 Apr 2004.
[10] Insights into Web Service Orchestration and Choreography Florian Daniel, Politecnico di Milano, Italy Barbara Pernici, Politecnico di Milano, Italy
[11] The Benefits of Service Choreography for Data-Intensive Computing Adam Barker
[12] Web Services Architecture by oracle
[13] Web Services Developer’s Guide Version 7.1.1 15
[14] Web Services Architecture by oracle
[15]SOA and web services from site http://www.roseindia.net
[16]A Dynamic Data Integration Model Based on SOA Jun Wang 2009 ISECS International Colloquium on Computing, Communication, Control, and Management . Alonso, F. Casati, H. Kuno, and V. Machiraju, Web Services:
[17]WSDL specification - W3C(2001) www.w3.org/TR/wsdl Utilizing WS-BPEL business processes through ebXML BPSS. Bahareh heravi
[18]Web Services Architecture W3C Working Group Note 11 February 2004.
[19] Web Services Architecture by oracle GAO, “Assessing the liability of Computer-Processed Data”, Ext Ver 1, October 2002.
[20] WFMC.About the WFMC - Introduction to the Workflow Management Coalition. Retrieved 12/07, 2004, from http://www.wfmc.org/about.htm
[21] Lawrence, P. (ed.), Workflow Handbook 1997. John Wiley & Sons LTD, Chichester, 1997.
[22]. R.Radhakrishnan, B.Sriraman, Aligning Architectural Approaches towards an SOA-Based Enterprise Architecture, in Proc. Working IEEE/IFIP Conference on Software Architecture 2007 (WICSA), Mumbai, India, Jan. 2007, pp. 38-38.
[23] Research and Application of SOA Standards in the Integration on Web Services Baoan Li