32. Why does a service need to be described?

(Lec03)For “use of service” to happen, the consumer needs to understand and the provider needs to tell the functions, usages and quality of the service.

33. In general, what aspects of a service need to be described for it to be understood properly by consumers? Which Web service standards are used to describe these aspects?

(Lec03)

*  What the service can do – functionality: (by WSDL …)
*  How good the service can do it – qualities: (partly by other WS standards …)
*  How to use the service – usage: (partly by WSDL …)

34. Outline the structure of WSDL documents and explain the purpose of its elements

(Lec03) Abstract (interface) definitions

*  <types> data type definitions
*  <message> operation parameters
*  <operation> abstract description of service actions
*  <portType> set of operation definitions

 Concrete (implementation) definitions

*  <binding> operation bindings
*  <port> association of an endpoint with a binding
*  <service> location/address for each binding

35. Name and explain the WSDL message exchange patterns (MEP). How can we tell from a WSDL document which MEP is being used for an interaction?

(Lec03) One-way Messaging, Request/Response Messaging, Notification Messaging, Solicit/Response Messaging.

A one-way message defines only an input message.   
 A request/response message declare a single <input> element followed by a single <output> element in an <operation> element.  
 Notification message is when a <portType> element contains an <output> tag but no <input> message definitions.  
 Solicit/Response Message is when a <portType> element first declares an <output> tag and then a <input> message definition – exactly the reverse of a request/response operation

36. Explain how a Web service interface is linked to an implementation of the interface.

(Lec03) WSDL use binding to link the abstract interface operations to concrete service operations.

37. Develop the WSDL description for the following service: a flight booking service of a given airline.

(Lec03) <wsdl:definitions <!-- xsd namespaces defined here -- >  
 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"  
 xmlns:tns="http://www.example.org/PostalCodeServiceSOAPTopdown/"  
 xmlns:wsdl=http://schemas.xmlsoap.org/wsdl/ xmlns:xsd="http://www.w3.org/2001/XMLSchema"  
 targetNamespace="http://www.example.org/FlightBookingServiceSOAPTopdown/">

**<wsdl:types>** <!-- xsd type definition for message parts here -- >

<xsd:schema targetNamespace="http://example.com/stockquote/schemas"  
 **<xsd:complexType name="CustomerInfoType">  
 <xsd:sequence>  
 <xsd:element name="CusNamer" type="xsd:string"/>  
 <xsd:element name="CusAddress" type="xsd:string"/>  
 </xsd:sequence>  
 </xsd:complexType>  
 <xsd:complexType name="POType">  
 <xsd:sequence>  
 <xsd:element name="PONumber" type="integer"/>  
 <xsd:element name="PODate" type="string"/>  
 </xsd:sequence>  
 </xsd:complexType>  
 <xsd:complexType name="InvoiceType">  
 <xsd:all>  
 <xsd:element name="InvPrice" type="float"/>  
 <xsd:element name="InvDate" type="string"/>  
 </xsd:all>  
 </xsd:complexType>**

</xsd:schema>

**</wsdl:types>**

**<wsdl:message** name="**POMessage**"> <!--input message -->

**<wsdl:part** name="PurchaseOrder" **type="tns:POType"**/>

**<wsdl:part** name="CustomerInfo" **type="tns:CustomerInfoType”**/>

**</wsdl:message>**

**<wsdl:message** name="**InvMessage**"> <! -- output message -->

**<wsdl:part** name="Invoice" type="tns:InvoiceType"/>

**</wsdl:message>**

**<wsdl:portType name="PurchaseOrderPortType">  
 <wsdl:operation name="SendPurchase">  
 <wsdl:input message="tns:POMessage"/>  
 <wsdl:output message="tns:InvMessage"/>  
 </wsdl:operation>  
</wsdl:portType>**

**<wsdl:binding name="PurchaseOrderSOAPBinding"  
 type="tns:PurchaseOrderPortType">  
 <!-- leverage off soapbind:binding synchronous style -->  
 <soapbind:binding style="rpc"  
 transport="http://schemas.xmlsoap.org/soap/http/"/>**

**<wsdl:operation name="SendPurchase">  
 <!-- again bind to SOAP -->  
 <soapbind:operation  
 soapAction="http://supply.com/ PurchaseService/wsdl/ SendPurchase" style="rpc"/>  
 <!-- furthur specify that the messages in the wsdl:operation use SOAP -->  
 <wsdl:input>  
 <soapbind:body use="literal" namespace="http://supply.com/PurchaseService/wsdl"/>  
 </wsdl:input>  
 <wsdl:output>  
 <soapbind:body use=“literal"  
 namespace="http://supply.com/ PurchaseService/wsdl"/>  
 </wsdl:output>  
 </wsdl:operation>  
</wsdl:binding>  
<wsdl:service name=“PurchaseOrderService">  
 <wsdl:port name=“PurchaseOrderPort" binding="tns:PurchaseOrderSOAPBinding">  
 <!-- give the binding a network endpoint address or URI of service -->  
 <soapbind:address location="http://supply.com:8080/PurchaseOrderService"/>  
 </wsdl:port>  
</wsdl:service>**

38. What is the difference between “Business Process” and a “workflow”

(Lec07) A business process is a set of logically related tasks performed to achieve a well-defined business outcome. And workflow is more like a sequence of activities for complete a task.

39. What is HATEOAS and how is it used to implemented processes in RESTful architectures?

(Lec07) Hypertext As The Engine Of Application State (HATEOAS).   
Response from the server contain not just the information requested, but links to next step(s), and server can customize response link depend on resource state. Process enactment is driven by the clients from choices provided by the server.

40. How could progress of a process be tracked in a Resource-based service composition?

(Lec07) State-based – process tracked by the changing state of the artifact

41. Explain why event-driven processes can be readily adapted

(Lec07 )Because services do not need to know about each other, so it is easy to extend event-driven process by add new service.

42. What needs to be changed if the control flow of a RESTful process needs to be altered?

Not find

43. Explain what is meant by a BPEL abstract process.

(Lec08) An abstract process describes “behavior”. It may not be executable. It may omit certain information

44. List the 7 main sections in a BPEL process definition and briefly describe their function.

(Lec08 & 09)

partnerLinks: establish peer-to-peer partner relationships by specifying the roles of each party.

variables: To maintain the state of a business process, typed data structure that stores messages associated with a workflow instance in order to facilitate stateful interactions among Web services (partners)

correlationSets: a group of properties – it is used to correlate an (Lec 09)

incoming message with a process instance

activities: basic activities sending & receiving of messages so that a process (Web service) instance can communicate with other Web services. Structured activiteis describe how a business process is created. (Lec08 & 09)

(Lec09) faultHandlers: Handles faults thrown in their scope

(Lec09) compensationHandlers: Define how to undo a successfully completed activity

(Lec09) eventHandlers: Allows the scope to react to events (messages) or alarms (e.g. expiration of timers)

45. What is a PartnerLinkType in BPEL?

(Lec08) The <partnerLinkType> is used to describe the communicational relationship between two partners by defining the type of role (at least 1 and at most 2) that each partner plays in an interaction and the port types used in both directions.

46. How many roles does an asynchronous partner link have?

(Lec08) 2 roles: myRole & partnerRole

47. What is a BPEL binding scheme? Give examples of four different schemes.

(Lec08)

Static design-time binding: process model has explicit endpoints assigned

Static deployment time binding: process model does not have explicit endpoints assigned but end-points are defined when the process is deployed to the workflow management system

Dynamic binding using lookups: criteria defined on the partnerLink that an end-point must have registry

Dynamic binding with passed-in endpoints: copying an endpoint variable from an invocation or request

48. Describe how parallel activities are implemented in BPEL.

(Lec09) Use control link to synchronise a source activity to a target activity, Transitions conditions associated with each link evaluated when activity completes, then all target activities process in parallel as source activity complete. There may have a join point or may not.

49. How does BPEL create process instances? How is one process instance distinguished from another?

(Lec09) <receive> or <pick> (start activities) with flag createInstance="yes". First message to arrive at a "starting point" initiates a new process instance

Each process must have at least one explicit start activity. (This question is not sure)

50. What code is required to create an instance of a BPEL process?

(Lec09) <receive> or <pick> (start activities) with flag createInstance="yes".

51. What is the difference between BPEL ‘compensation handler’ and ‘fault handler’?

(Lec09) “fault handler” catch a fault and do activities specified in catch block.

“compensation handler” can only be used within FCT (fault-compensation-termination) -handlers, right after fault handler.

52. Which Web architectural style is more suitable for knowledge intensive processes. Explain the rationale for your answer.

(Lec07) EDA – Event Driven Architecture

Goal rather than task oriented

Process instance needs to be adapted

**Quality Attributes, Policies and Quality Standards**

1. What are policies and why are they important for WS applications?

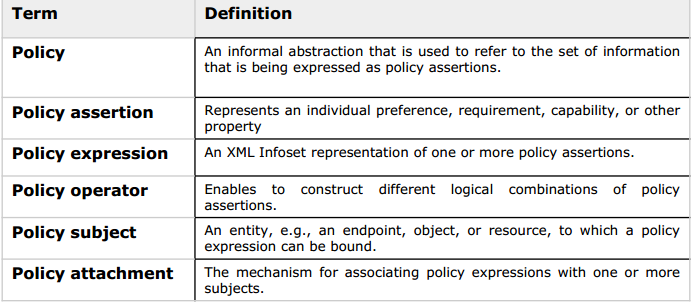
(Lec05) A way of extending WSDL to express quality aspects expressed in other standards

We may need to describe a range of other quality attributes of a service

2. List two common policies for Web services

(Lec05) Versioning policies, QoS policies, Security policies,

3. Describe and explain the structure of WS-Policy document. Name each of its parts.



4. How are policies usually enforced?

5. Describe the four levels of the message delivery assurances.

* **AtLeastOnce Delivery**: This feature guarantees that every message sent will be delivered or an error will be raised on at least one endpoint. Some messages may be delivered more than once.
* **AtMostOnce Delivery**: This feature guarantees that every message will be delivered at most once without duplication or an error will be raised on at least one endpoint. It is possible that some messages in a sequence may not be delivered.
* **ExactlyOnce Delivery**: This feature guarantees that every message sent will be delivered without duplication or an error will be raised on at least one endpoint. This delivery assurance is the logical "and" of the two prior delivery assurances.
* **InOrder Delivery**: This feature enforces the delivery of a sequence of messages at the destination, in the same order as the submission order by the sending application. This delivery assurance says nothing about duplications or omissions.