

UNIT-I

INTRODUCTION TO XML

9

XML document structure – Well formed and valid documents – Namespaces – DTD – XML Schema – X-Files.

PART A

Q.No	Questions	CO	Bloom's Level										
1	<p>What are the major portions of XML document?</p> <p>The major portions of an XML document include the following:</p> <ul style="list-style-type: none"> • The XML declaration • The Document Type Declaration • The element data • The attribute data • The character data or XML content 	C405.1	BTL1										
2	<p>What are the components of XML declaration?</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left; padding: 5px;"><i>Component</i></th> <th style="text-align: left; padding: 5px;"><i>Description</i></th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;"><code><?xml</code></td> <td style="padding: 5px;">Starts the beginning of the processing instruction (in case, for the XML declaration).</td> </tr> <tr> <td style="padding: 5px;"><code>Version="xxx"</code></td> <td style="padding: 5px;">Describes the specific version of XML being used in document (in this case, version 1.0 of the W3C specification). Future iterations could be 2.0, 1.1, and so on.</td> </tr> <tr> <td style="padding: 5px;"><code>standalone="xxx"</code></td> <td style="padding: 5px;">This standalone option defines whether documents are allowed to contain external markup declarations. This option can be set to "yes" or "no".</td> </tr> <tr> <td style="padding: 5px;"><code>encoding="xxx"</code></td> <td style="padding: 5px;">Indicates the character encoding that the document uses. The default is "US-ASCII" but can be set to any value that XML processors recognize and can support. The most common alternate setting is "UTF-8".</td> </tr> </tbody> </table>	<i>Component</i>	<i>Description</i>	<code><?xml</code>	Starts the beginning of the processing instruction (in case, for the XML declaration).	<code>Version="xxx"</code>	Describes the specific version of XML being used in document (in this case, version 1.0 of the W3C specification). Future iterations could be 2.0, 1.1, and so on.	<code>standalone="xxx"</code>	This standalone option defines whether documents are allowed to contain external markup declarations. This option can be set to "yes" or "no".	<code>encoding="xxx"</code>	Indicates the character encoding that the document uses. The default is "US-ASCII" but can be set to any value that XML processors recognize and can support. The most common alternate setting is "UTF-8".	C405.1	BTL1
<i>Component</i>	<i>Description</i>												
<code><?xml</code>	Starts the beginning of the processing instruction (in case, for the XML declaration).												
<code>Version="xxx"</code>	Describes the specific version of XML being used in document (in this case, version 1.0 of the W3C specification). Future iterations could be 2.0, 1.1, and so on.												
<code>standalone="xxx"</code>	This standalone option defines whether documents are allowed to contain external markup declarations. This option can be set to "yes" or "no".												
<code>encoding="xxx"</code>	Indicates the character encoding that the document uses. The default is "US-ASCII" but can be set to any value that XML processors recognize and can support. The most common alternate setting is "UTF-8".												
3	<p>What are the XML delimiter characters?</p>	C405.1	BTL1										

	<table border="1"> <thead> <tr> <th><i>Character</i></th> <th><i>Meaning</i></th> </tr> </thead> <tbody> <tr> <td><</td> <td>The start of an XML markup tag</td> </tr> <tr> <td>></td> <td>The end of an XML markup tag</td> </tr> <tr> <td>&</td> <td>The start of an XML entity</td> </tr> <tr> <td>;</td> <td>The end of an XML entity</td> </tr> </tbody> </table>	<i>Character</i>	<i>Meaning</i>	<	The start of an XML markup tag	>	The end of an XML markup tag	&	The start of an XML entity	;	The end of an XML entity										
<i>Character</i>	<i>Meaning</i>																				
<	The start of an XML markup tag																				
>	The end of an XML markup tag																				
&	The start of an XML entity																				
;	The end of an XML entity																				
4	<p>Define Document Type Declaration. The Document Type Declaration (DOCTYPE) gives a name to the XML content and provides a means to guarantee the document's validity, either by including or specifying a link to a Document Type Definition (DTD).</p>	C405.1	BTL1																		
5	<p>What are the components of Document Type Declaration?</p> <table border="1"> <thead> <tr> <th><i>Component</i></th> <th><i>Description</i></th> </tr> </thead> <tbody> <tr> <td><</td> <td>The start of the XML tag (in this case, the beginning of the Document Type Declaration).</td> </tr> <tr> <td>!DOCTYPE</td> <td>The beginning of the Document Type Declaration.</td> </tr> <tr> <td>NAME</td> <td>Specifies the name of the document type being declared. This must comply with XML naming rules.</td> </tr> <tr> <td>SYSTEM</td> <td>Specifies that the following system identifier is to be used and processed.</td> </tr> <tr> <td>"file"</td> <td>Specifies the name of the file to be processed.</td> </tr> <tr> <td>[</td> <td>Starts an internal DTD subset.</td> </tr> <tr> <td>]</td> <td>Ends the internal DTD subset.</td> </tr> <tr> <td>></td> <td>The end of the XML tag (in this case, the end of the Document Type Declaration).</td> </tr> </tbody> </table>	<i>Component</i>	<i>Description</i>	<	The start of the XML tag (in this case, the beginning of the Document Type Declaration).	!DOCTYPE	The beginning of the Document Type Declaration.	NAME	Specifies the name of the document type being declared. This must comply with XML naming rules.	SYSTEM	Specifies that the following system identifier is to be used and processed.	"file"	Specifies the name of the file to be processed.	[Starts an internal DTD subset.]	Ends the internal DTD subset.	>	The end of the XML tag (in this case, the end of the Document Type Declaration).	C405.1	BTL1
<i>Component</i>	<i>Description</i>																				
<	The start of the XML tag (in this case, the beginning of the Document Type Declaration).																				
!DOCTYPE	The beginning of the Document Type Declaration.																				
NAME	Specifies the name of the document type being declared. This must comply with XML naming rules.																				
SYSTEM	Specifies that the following system identifier is to be used and processed.																				
"file"	Specifies the name of the file to be processed.																				
[Starts an internal DTD subset.																				
]	Ends the internal DTD subset.																				
>	The end of the XML tag (in this case, the end of the Document Type Declaration).																				

6	<p>Define elements in XML document.</p> <p>XML elements can be defined as building blocks of an XML. Elements can behave as containers to hold text, elements, attributes, media objects or all of these. Each XML document contains one or more elements, the scope of which are either delimited by start and end tags, or for empty elements, by an empty-element tag.</p>	C405.1	BTL1
7	<p>Define attributes in XML</p> <p>When you create XML Schemas, you define the individual elements and attributes and assign valid types to them. Elements describe data, whereas attributes are like properties of an element, in that they provide further definition about the element the way that properties describe characteristics of objects and classes.</p>	C405.1	BTL1
8	<p>Define components in XML</p> <p>Comments are quite simple to include in a document. The character sequence <!-- begins a comment and --> ends the comment. Between these two delimiters, any text at all can be written, including valid XML markup. The only restriction is that the comment delimiters cannot be used; neither can the literal string --. Comments can be placed anywhere in a document and are not considered to be part of the textual content of an XML document.</p>	C405.1	BTL1
9	<p>Define DTD</p> <p>Document Type Definitions (DTDs) provide a means for defining what XML markup can occur in an XML document. Basically, the DTD provides a mechanism to guarantee that a given XML document complies with a well-defined set of rules for document structure and content. These rules provide a framework for guaranteeing the “validity” of a document. DTDs and the more recent XML Schema are the means for defining the validity constraints on XML documents.</p>	C405.1	BTL1
10	<p>Define XML content</p> <p>The content between XML elements is where most of the value lies in an XML document. In fact, that is almost exclusively where all the variable content lies. XML elements are usually well defined and strict in their application</p>	C405.1	BTL1

11	<p>When the document will be considered as valid documents?</p> <p>A well-formed XML document is considered valid only if it contains a proper Document Type Declaration and if the document obeys the constraints of that declaration. In most cases, the constraints of the declaration will be expressed as a DTD or an XML Schema. Well-formed XML documents are designed for use without any constraints, whereas valid XML documents explicitly require these constraint mechanisms. In addition to constraining the possible elements and the ordering of those elements in a document, valid XML documents can take advantage of certain advanced features of XML that are not available to merely well-formed documents due to their lack of a DTD or XML Schema.</p>		BTL1
12	<p>Define well-formed documents.</p> <p>An XML document is well formed if it follows all the preceding syntax rules of XML. On the other hand, if it includes inappropriate markup or characters that cannot be processed by XML parsers, the document cannot be considered well formed. It goes without saying that an XML document can't be partially well formed.</p>	C405.1	BTL1
13	<p>Define Namespaces in XML.</p> <p>The xmlns attribute in the second <table> element gives the f: prefix a qualified namespace. When a namespace is defined for an element, all child elements with the same prefix are associated with the same namespace. Namespaces can also be declared in the XML root element: <root xmlns:h="http://www.w3.org/TR/html4/"</p>	C405.1	BTL1
14	<p>Define DTD Attributes</p> <p>XML attributes are name/value pairs that are used as metadata to describe XML elements. XML attributes are very similar to HTML attributes. In HTML, src is an attribute of the img tag, as shown in the following example: In this example, width and height are also attributes of the img tag.</p>	C405.1	BTL1
15	<p>Define DTD Entities</p> <p>Entities in DTDs are storage units. They can also be considered placeholders. Entities are special markups that contain content for insertion into the XML document. Usually this will be some type of information that is bulky or repetitive. Entities make this type of information more easily handled because the DTD author can use them to indicate where the information should be inserted in the XML document.</p>	C405.1	BTL1

16	<p>What are the DTD Drawbacks and Alternatives?</p> <p>There are several drawbacks that limit the ability of DTDs to meet these growing and changing validation needs.</p> <p>First and foremost, DTDs are composed of non-XML syntax. Given that one of the central tenets of XML is that it be totally extensible, it may not seem to make a lot of sense that this is the case for DTDs.</p> <p>Additionally, there can only be a single DTD per document. It is true that there can be internal and external subsets of DTDs, but there can only be a single DTD referenced. In the modern programming world, we are used to being able to draw the programming constructs we use from different modules or classes.</p>	C405.1	BTL1
17	<p>How to create XML Schemas?</p> <p>Authoring an XML schema consists of declaring elements and attributes as well as the “properties” of those elements and attributes. We will begin our look at authoring XML schemas by working our way from the least-complex example to the most-complex example. Because attributes may not contain other attributes or elements, we will start there.</p>	C405.1	BTL1
18	<p>How to create XML Schemas?</p> <p>Authoring an XML schema consists of declaring elements and attributes as well as the “properties” of those elements and attributes. We will begin our look at authoring XML schemas by working our way from the least-complex example to the most-complex example. Because attributes may not contain other attributes or elements, we will start there.</p>	C405.1	BTL1
19	<p>Define XPath.</p> <p>The XML Path Language (XPath) is a standard for creating expressions that can be used to find specific pieces of information within an XML document. XPath expressions are used by both XSLT (for which XPath provides the core functionality) and XPointer to locate a set of nodes. To understand how XPath works, it helps to imagine an XML document as a tree of nodes consisting of both elements and attributes.</p>	C405.1	BTL1
20	<p>Define XPointer.</p> <p>The XML Pointer Language (XPointer), currently in the candidate recommendation stage of the W3C approval process, builds on the XPath specification. An XPointer uses location steps the same as XPath but with two major differences: Because an XPointer describes a location within an external document, an XPointer can target a point within that XML document or a range within the target XML document. You can find the</p>	C405.1	BTL1

	complete specification at http://www.w3.org/TR/xptr .		
21	<p>Define XLink.</p> <p>The anchor element, <a>, within HTML indicates a link to another resource on an HTML page. This could be a location within the same document or a document located elsewhere. In HTML terms, the anchor element creates a hyperlink to another location. The hyperlink can either appear as straight text, a clickable image, or a combination of both. Although HTML anchor elements contain a lot of functionality, they are still limiting— they require the use of the anchor element (<a>) itself, and they basically sit there waiting for someone to click them before navigating to the specified location.</p>	C405.1	BTL1
22	<p>Describe targeting Namespaces.</p> <p>You can view an XML schema as a collection of type definitions and element declarations targeted for a specific namespace. Namespaces allow us to distinguish element declarations and type definitions of one schema from another. We can assign an intended namespace for an XML schema by using the targetNamespace attribute on the <schema> element. By assigning a target namespace for the schema, we indicate that an XML document whose elements are declared as belonging to the schema’s namespace should be validated against the XML schema.</p>	C405.1	BTL1
23	<p>How to model groups?</p> <p>A model group, at least in terms of a schema definition, is a logically grouped set of elements. A model group within the XML Schema Definition Language consists of a “compositor” and a list of “particles” or element declarations). A model group can be constructed using one of the following XML Schema Definition elements:</p> <ul style="list-style-type: none"> • <all> • <choice> • <sequence> 	C405.1	BTL1
24	<p>What is anonymous type declaration?</p> <p>If you look closely, you’ll see the declaration of a <Name> element that does not have a type attribute specified. Instead, the <element> element, itself, contains a <simpleType> element without a name attribute specified. This is known as an “anonymous” type definition.</p>	C405.1	BTL1

25	<p>How to declare simple types?</p> <p>Sometimes, it's not necessary to declare a complex element type within an XML schema. In these cases, you can use the <simpleType> element of the XML Schema Definition Language.</p>	C405.1	BTL1
26	<p>Define the structure of a Document Type Definition?</p> <p>The structure of a DTD consists of a Document Type Declaration, elements, attributes, entities, and several other minor keywords. We will take a look at each of these topics, in that order. As we progress from topic to topic, we will follow a mini case study about the use of XML to store employee records by the Human Resources department of a fictitious company.</p>	C405.1	BTL1
27	<p>Define Ranges in XPointer.</p> <p>An XPointer range defines just that—a range consisting of a start point and an endpoint. A range will contain the XML between the start point and endpoint but does not necessarily have to consist of neat subtrees of an XML document. A range can extend over multiple branches of an XML document. The only criterion is that the start point and endpoint must be valid.</p>	C405.1	BTL1
28	<p>Define Simple Links.</p> <p>A simple link combines the functionality provided by the different pieces available through an extended link together into a shorthand notation. A simple link consists of an xlink:type attribute with a value of simple and, optionally, an xlink:href attribute with a specified value. Simple links play multiple roles in linking documents. For instance, the simple link, itself, acts as a resource XLink type for the local document. It is the combination of this functionality that shortens the XLink definition for a simple link.</p>	C405.1	BTL1
29	<p>Define Extended Links.</p> <p>Within the XML Linking Language, extended links give you the ability to specify relationships between an unlimited number of resources, both local and remote. In addition, these links can involve multiple paths between the linked resources. Local resources are part of the actual extended link, whereas remote resources identify external resources to the link. An out-of-line link is created when there are no local resources at all for a link.</p>	C405.1	BTL1
30	<p>Define the rules for well formed documents in XML NOV/DEC 2016</p> <p>An XML document is well formed if it follows all the preceding syntax rules of XML. On the other hand, if it includes inappropriate markup or characters that cannot be processed by XML parsers, the document cannot be considered well formed. It goes without saying that an XML document can't be partially well formed.</p>	C405.1	BTL1

31	What is a XML namespace? NOV/DEC 2016 XML namespaces are used for providing uniquely named elements and attributes in an XML document. They are defined in a W3C recommendation. An XML instance may contain element or attribute names from more than one XML vocabulary.	C405.1	BTL1
32	What is XML? NOV/DEC 2017 In computing, Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable through use of tags that can be created and defined by users.	C405.1	BTL1
33	What is XML document prolog? NOV/DEC 2017 The prolog refers to the information that appears before the start tag of the document or root element. It includes information that applies to the document as a whole, such as character encoding, document structure, and style sheets.	C405.1	BTL1
34	Define service Component.(MAY/JUNE 2018) Service-component architecture (SCA) Service-component architecture (SCA) is a group of specifications intended for the development of applications based on service-oriented architecture (SOA), which defines how computing entities interact to perform work for each other.	C405.1	BTL1

PART-B

Q.No	Questions	CO	Bloom's level
1	1. Explain XML document structure in detail (Ron Schmelzer Page 39-50)	C405.1	BTL5
2	2. Explain Namespaces in Detail. (Ron Schmelzer Page 58-61) (NOV/DEC 2017)	C405.1	BTL5
3	3. Explain Document Type Definition in detail (Ron Schmelzer Page 67-103)	C405.1	BTL5
4	4. How to create XML Schemas? Explain in detail (Ron Schmelzer Page 116-159) (NOV/DEC 2017)	C405.1	BTL5
5	5. Explain X-Files in detail (Ron Schmelzer Page 169-219)	C405.1	BTL5
6	Create a document type definition that defines the structure for email message, further create a XML document that reference to the created document type definitions	C405.1	BTL6
7	Explain with examples internal and external DTD (8) [NOV/DEC 2016]	C405.1	BTL5

8	Explain in detail about XML schema and XML files. (16) [NOV/DEC 2016]	C405.1	BTL5
9	Explain the Characteristics of SOA in Detail) [MAY/JUNE 2017]	C405.1	BTL5
10	Discuss the Principles of service orientation in detail[MAY/JUNE 2017]	C405.1	BTL6

UNIT-2

BUILDING XML- BASED APPLICATIONS

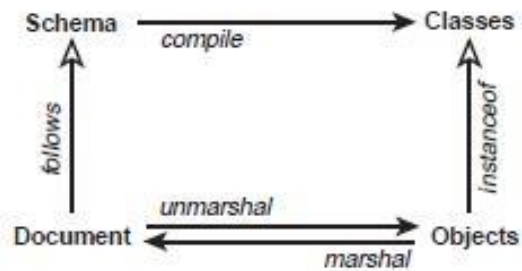
9

Parsing XML – using DOM, SAX – XML Transformation and XSL – XSL Formatting – Modeling Databases in XML.

PART-A

Q.No	Questions	CO	Bloom's Level
1	What is DOM? The Document Object Model (DOM) is the model that describes how all elements in an HTML page, like input fields, images, paragraphs etc., are related to the topmost structure: the document itself. By calling the element by its proper DOM name, we can influence it.	C405.2	BTL1
2	What are the disadvantages of Using DOM? One of the big issues is that DOM can be memory intensive. As mentioned earlier, when an XML document is loaded, the entire document is read in at once. A large document will require a large amount of memory to represent it. Other parsing methods, such as SAX, don't read in the entire document, so they are better in terms of memory efficiency for some applications	C405.2	BTL1
3	1. Define DOM Levels The DOM working group works on phases (or <i>levels</i>) of the specification. At the time of this writing, three levels are in the works. The DOM Level 1 and Level 2 specifications are W3C recommendations. This means that the specifications are final and can be implemented without fear of things changing. Level 1 allows traversal of an XML document as well as the manipulation of the content in that document. Level 2 extends Level 1 with additional features such as namespace support, events, ranges, and so on. Level 3 is currently a working draft.	C405.2	BTL1
4	Describe Java Bindings If you think about it, a class and a schema perform similar functions. Classes describe Java objects, whereas schemas describe XML documents. An object is an instance of a class, and a document follows a schema. The diagram in Figure 7.3 illustrates the relationships between schemas, classes, documents, and objects.	C405.2	BTL1

FIGURE 7.3
Binding relationships.

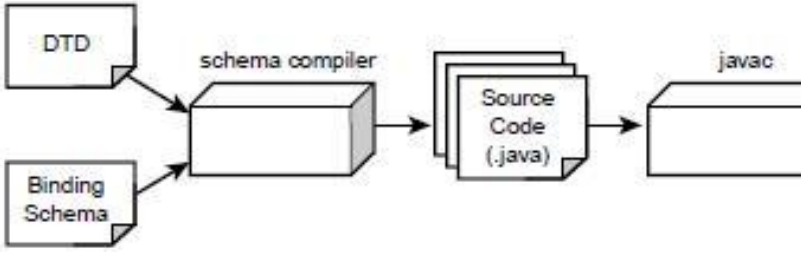


	<p>FIGURE 7.3 <i>Binding relationships.</i></p>		
5	<p>Describe SAX implementation. SAX implementations include all the underlying classes needed to parse documents. The SAX API by itself does not include these underlying classes, so you will need to obtain an implementation. You can find a list of implementations at http://www.megginson.com/SAX/applications.html. When looking for an implementation, you might want to consider several factors, such as version support, validating/nonvalidating, DTD/XML Schema support, and so on.</p>	C405.2	BTL1
6	<p>Define DOM Core The DOM core is available in DOM Level 1 and beyond. It permits you to create and manipulate XML documents in memory. As mentioned earlier, DOM is a tree structure that represents elements, attributes, and content.</p>	C405.2	BTL1
7	<p>Define DOM interface. NOV/DEC 17 A number of extended interfaces are not mandatory but may be available in some implementations. These interfaces are beyond DOM Level 1 and are discussed later in this chapter. You can determine whether these interfaces are supported by calling the <code>hasFeature()</code> method of the <code>DOMImplementation</code> interface. You can use the arguments "XML" and "2.0" for the feature and version parameters of the <code>hasFeature()</code> method. For a detailed explanation, refer to the DOM specification on the W3C Web site.</p>	C405.2	BTL1
8	<p>Define JAXB solutions. In the JAXB solution, we will model the rental property database as an XML document. First we need to review the database schema. After reviewing the schema, we will develop our desired XML document based on an XML schema. After we have the XML schema developed, we can create the JAXB binding schema. The JAXB binding schema contains instructions on how to bind the XML</p>	C405.2	BTL1

	schema to a Java class. We'll take the JAXB binding schema and generate the appropriate Java classes.		
9	<p>Define DOM Traversal.</p> <p>Traversal is a convenient way to walk through a DOM tree and select specific nodes. This is useful when you want to find certain elements and perform operations on them.</p>	C405.2	BTL1
10	<p>Define DOM Range</p> <p>A range consists of two boundary points corresponding to the start and the end of the range. A boundary point's position in a Document or DocumentFragment tree can be characterized by a node and an offset. The node is the container of the boundary point and its position. The container and its ancestors are the ancestor containers of the boundary point and its position. The offset within the node is the offset of the boundary point and its position. If the container is an Attr, Document, DocumentFragment, Element, or EntityReference node, the offset is between its child nodes.</p>	C405.2	BTL1
11	<p>Describe JDOM.</p> <p>JDOM was designed specifically for Java. In contrast, DOM is purely an interface specification independent of any language. For example, a Java parser can leverage standard Java types and collections, such as the String class and the Collections API. The goal of W3C DOM is to be language independent, which works but can add a lot of unnecessary complications</p>		BTL1
12	<p>2. Define Java Binding</p> <p>If you think about it, a class and a schema perform similar functions. Classes describe Java objects, whereas schemas describe XML documents. An object is an instance of a class, and a document follows a schema. The diagram in Figure 7.3 illustrates the relationships between schemas, classes, documents, and objects.</p>	C405.2	BTL1
13	<p>What is SAX?</p> <p>SAX by itself is just an API, and a number of implementations are available from many of the familiar sources. The most commonly used parsers are Xerces from the Apache XML project and Java API for XML Processing (JAXP) from Sun Microsystems. A good list of parsers can be found at http://www.xmlsoftware.com.</p>	C405.2	BTL1

14	<p>Define SAX packages</p> <p>The SAX 2.0 API is comprised of two standard packages and one extension package. The standard packages are org.xml.sax and org.xml.helpers. The org.xml.sax package contains the basic classes, interfaces, and exceptions needed for parsing documents.</p>	C405.2	BTL1
15	<p>What is Entity references?</p> <p>SAX parsers will resolve entity references automatically. However, there are cases when you might want to resolve an entity reference yourself. In the following example, we will define an entity for hardcover books. It will be referenced as &hc; and defined in our DTD. If we use an HTTP URL to define the entity, the SAX parser will go out to the network to resolve it. What we want to do here is resolve the entity using a local file. We can accomplish this using an EntityResolver</p>	C405.2	BTL1
16	<p>Define Lexical Events.</p> <p>LexicalHandler is part of the org.xml.sax.ext package, which is not necessarily supported by all SAX implementations. Xerces, of course, provides support for the extension package. Notice that we are explicitly implementing LexicalHandler. This is necessary because DefaultHandler does not implement LexicalHandler. We must fill in all methods of LexicalHandler whether we are interested in them or not.</p>	C405.2	BTL1
17	<p>What are the Activities in the life cycle?</p> <ul style="list-style-type: none"> • Requirements specification • Architectural design • Detailed design • Coding and unit testing • Integration and testing <p>Maintenance</p>	C405.2	BTL1
18	<p>Define Handling Errors.</p> <p>The Locator interface can give us the parse position within a ContentHandler method. The position information includes line number and column number. It is important to note that the Locator object should <i>not</i> be used in any other methods, including ErrorHandler methods. Fortunately, ErrorHandler methods</p>	C405.2	BTL1

	supply a SAXParseException object that can also give us position information.		
19	<p>Describe SAX implementation.</p> <p>SAX implementations include all the underlying classes needed to parse documents. The SAX API by itself does not include these underlying classes, so you will need to obtain an implementation. You can find a list of implementations at http://www.megginson.com/SAX/applications.html. When looking for an implementation, you might want to consider several factors, such as version support, validating/nonvalidating, DTD/XML Schema support, and so on</p>	C405.2	BTL1
20	<p>Describe Basic document structure in XSL formatting objects.</p> <p>An XML-FO document follows the syntax rules of XML; as a result, it is well formed. XSL-FO elements use the following namespace: http://www.w3.org/1999/XSL/Format</p> <p>The following code snippet shows the basic document setup for XSL-FO:</p> <pre><?xml version="1.0" encoding="utf-8"?> <fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format"> <!-- layout master set --> <!-- page masters: size and layout --> <!-- page sequences and content --> </fo:root></pre>	C405.2	BTL1
21	<p>How to generate a PDF document?</p> <p>Follow these steps to generate a PDF document from simple.fo:</p> <ol style="list-style-type: none"> 1. Open an MS-DOS window. 2. Move to the directory <install_dir>\ch9_xsl\xsl_fo\. 3. Set up the Java classpath by typing setpaths. 4. Execute Apache-FOP by typing fop simple.fo simple.pdf. <p>The Apache-FOP formatter now reads the input file simple.fo and generates the output file simple.pdf.</p> <ol style="list-style-type: none"> 5. View the simple.pdf file in Adobe Acrobat Reader. Your screen should resemble 	C405.2	BTL1

22	<p>Describe XML Database Mapping?</p> <p>The first type of XML database solution provides a mapping between the XML document and the database fields. The system dynamically converts SQL result sets to XML documents. Depending on the sophistication of the product, it may provide a graphical tool to map the database fields to the desired XML elements. Other tools support a configuration file that defines the mapping. These tools continue to store the information in relational database management system (RDBMS) format. They simply provide an XML conversion process that is normally implemented as a server-side Web application.</p>	C405.2	BTL1
23	<p>How to create JAXB binding schema?</p> <p>Now that the DTD is defined for our document, we need to define the JAXB binding schema. The JAXB binding schema is an XML document that contains instructions on how to bind a DTD to a Java class. Using the JAXB binding schema, we can define the names of the generated Java classes, map element names to specific properties in the Java class, and provide the mapping rules for attributes. The following code example informs the JAXB system that the element <rental_property_list> should be mapped to a Java class and that it is the root element for the XML document:</p> <pre><element name="rental_property_list" type="class" root="true"/></pre>	C405.2	BTL1
24	<p>How to generate the JAXB classes based on schema?</p> <p>Now we are ready to generate the Java source files based on our schemas. JAXB provides a schema compiler for generating the Java source files. The schema compiler takes as input the DTD and the JAXB binding schema. Figure 10.6 illustrates the process.</p> <p>FIGURE 10.6 <i>Generating Java classes with the JAXB compiler.</i></p>  <pre> graph LR DTD[DTD] --> SC[schema compiler] BS[Binding Schema] --> SC SC --> SCODE[Source Code (.java)] SCODE --> JAVAC[javac] </pre>	C405.2	BTL1
25	<p>How to represent the XML Document?</p> <p>In any markup language, the first element to appear is called the "root element", which defines what kind of document the file will be. In an HTML file, the <html> tag is the root element. An HTML file will always have the HTML element as the root element, while in an XML file, it can be anything.</p>	C405.2	BTL1

	<p>Eg: <phonebook> <number> </number> <name> </name> </phonebook></p>		
26	<p>3. Mention any 3 XML Parsers ☒SAX (Simple API for XML) Parser ☒DOM (Document Object Model) Parser and ☒XSLT (XML Style Sheet) Parsers.</p>	C405.2	BTL1
27	<p>What is the purpose of the XML DTD The purpose of a DTD is to define the structure of an XML document. It defines the structure with a list of legal elements: <!DOCTYPE note [<!ELEMENT note (to,from,heading,body)> <!ELEMENT to (#PCDATA)> <!ELEMENT from (#PCDATA)> <!ELEMENT heading (#PCDATA)> <!ELEMENT body (#PCDATA)>]></p>	C405.2	BTL1
28	<p>What is XML Prolog XML file always starts with a Prolog. The minimal prolog contains a declaration that identifies the document as an XML document, like this: <?xml version="1.0"?> The declaration may also contain additional information, like this: <?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?> .</p>	C405.2	BTL1
29	<p>What is XSL Programming? XSL (XML Stylesheet) Programming is the Next Generation of the CSS (Cascading Style Sheet Programming). In CSS, users can use certain style tags which the browsers can understand and are predefined by W3 consortium. XSL takes ths to one step ahead and users can define any tags in the XML file. XML sheets can help in showing the same data in different formats. <?xml-stylesheet href="doc.xsl" type="text/xsl"?></p>	C405.2	BTL1

30	<p>What is XSLT?</p> <p>XSLT stands for XSL Transformations XSLT is the most important part of XSL XSLT transforms an XML document into another XML document XSLT uses XPath to navigate in XML documents XSLT is a W3C Recommendation</p>	C405.2	BTL1
31	<p>Define XML parser [NOV/DEC 2016]</p> <p>An XML Parser is a parser that is designed to readXML and create a way for programs to use XML. There are different types, and each has its advantages. Unless a program simply and blindly copies the whole XML file as a unit, every program must implement or call on anXML parser</p>	C405.2	BTL1
32	<p>Outline a simple style sheet using XSL [NOV/DEC 2016] □</p> <p>XSL is a language for expressing style sheets. An XSL style sheet is, like with CSS, a file that describes how to display an XML document of a given type. XSL shares the functionality and is compatible with CSS2 (although it uses a different syntax). It also adds</p> <p>Example Style Sheet:</p> <pre><xsl:template match="FX"> <fo:block font-weight="bold"> <xsl:apply-templates/> </fo:block> </xsl:template></pre>	C405.2	BTL2
33	<p>List the disadvantage of SAX [NOV/DEC 17]</p> <ul style="list-style-type: none"> • SAX is similar to a one-pass compiler. After it reads part of the document, it cannot navigate backward to reread the data it has processed, unless you start all over again. • Because SAX does not store the data that it has processed, you cannot modify this data and store it back in the original document. • Because SAX does not create an in-memory document structure, you cannot build an XML document by using a SAX parser. 	C405.2	BTL1
34	<p>Define the characteristics of Orchestration service layer. (MAY/JUNE 2018)</p> <p>The orchestration service layer introduces a parent level of abstraction that alleviates the need for other services to manage interaction details required to ensure that service operations are executed in a specific sequence. Within the orchestration service layer, process services compose other services that provide specific sets of functions, independent of the business rules and scenario-specific logic required to execute a process instance.</p>	C405.2	BTL1

32	<p>Define soap message. .(MAY/JUNE 2018)</p> <p>A SOAP message is an ordinary XML document containing the following elements –</p> <ul style="list-style-type: none"> • Envelope – Defines the start and the end of the message. It is a mandatory element. • Header – Contains any optional attributes of the message used in processing the message, either at an intermediary point or at the ultimate end-point. It is an optional element. • Body – Contains the XML data comprising the message being sent. It is a mandatory element. • Fault – An optional Fault element that provides information about errors that occur while processing the message. 	C405.2	BTL1
----	---	--------	------

PART-B

Q.No	Questions	CO	Bloom's level
1	What is DOM? Explain Parsing XML using DOM (Ron Schmelzer Page 267-296)	C405.2	BTL5
2	Explain Parsing using SAX (Ron Schmelzer Page 309-331)	C405.2	BTL5
3	Explain XML transformation with XSL (Ron Schmelzer Page 345-376)	C405.2	BTL5
4	Explain modelling databases in XML in detail (Ron Schmelzer Page 409-437)	C405.2	BTL5
5	Explain XSL formatting objects in detail (Ron Schmelzer Page 377-391)	C405.2	BTL6
6	Compare DOM and SAX based XML parsing. (6) [NOV/DEC 2016]	C405.2	BTL5
7	Explain SAX based parsing with example. (10) [NOV/DEC 2016]	C405.2	BTL5
8	Give a brief note on Modeling database in XML (8) [NOV/DEC 2016]	C405.2	BTL5

9	With example formulate how XSLT can transform an XML document into HTML.(8) [NOV/DEC 2016]	C405.2	BTL6
10	Discuss on the protocol of Atomic Transaction in detail. .(MAY/JUNE 2018)	C405.2	BTL6
11	Explain the elements of Web services platform in detail. .(MAY/JUNE 2018)	C405.2	BTL5

UNIT-3

SERVICE ORIENTED ARCHITECTURE

9

Characteristics of SOA, Comparing SOA with Client-Server and Distributed architectures – Benefits of SOA -- Principles of Service orientation – Service layers.

PART-A

Q.No	Questions	CO	Bloom's Level
1	What is architecture? Application architecture is to an application development team what a blueprint is to a team of construction workers. Different organizations document different levels of application architecture.	C405.3	BTL1
2	Define Service-oriented architecture An SOA can refer to application architecture or the approach used to standardize technical architecture across the enterprise	C405.3	BTL1
3	Define Client / Server architecture Mainframe back-ends served thin clients, are considered an implementation of the single-tier client-server architecture Mainframe systems natively supported both synchronous and asynchronous communication. The latter approach was used primarily to allow the server to continuously receive characters from the terminal in response to individual key-strokes. Only upon certain conditions would the server actually respond.	C405.3	BTL1
4	Define Distributed Internet architecture Distributing application logic among multiple components (some residing on the client, others on the server) reduced deployment headaches by centralizing a greater amount of the logic on servers. Server-side components, now located on dedicated application servers, would then share and manage pools of database connections, alleviating the burden of concurrent usage on the database server A single connection could easily facilitate multiple users.	C405.3	BTL1

5	<p>Define SOA Characteristics [NOV/DEC 17]</p> <ul style="list-style-type: none"> • Services are discoverable and dynamically bound. • Services are self-contained and modular. • Services stress interoperability. • Services are loosely coupled. • Services have a network-addressable interface. • Services have coarse-grained interfaces. • Services are location-transparent. • Services are composable. • Service-oriented architecture supports self-healing. 	C405.3	BTL1
6	<p>Define Coarse-Grained Services</p> <p>A service-based system controls the network access to the objects within the service through a set of coarse-grained interfaces. A service may still be implemented as a set of fine-grained objects, but the objects themselves are not accessible over a network connection. A service implemented as objects has one or more coarse-grained objects that act as distributed façades. These objects are accessible over the network and provide access to the internal object state from external consumers of the service. However, objects internal to the service communicate directly with each other within a single machine, not across a network connection</p>	C405.3	BTL1
7	<p>Define Service Component</p> <p>This is the true heart of the SOA. The Service Component is that logical unit of code which implements the functionality to support the Service. The Service Component exposes one or more Services. A Service Component is also usually associated with a data store of some kind. This can contain data about a fundamental data type, control data, process, data, etc depending on the nature of the particular service component.</p>	C405.3	BTL1
8	<p>Define Service-component-level Testing</p> <p>Service-component-level testing or Unit testing, is normally performed by the developers to test that the code not only successfully compiles, but the basic functionality of the components and functions within a service are working as specified. The primary goal of Component testing is to take the smallest piece of testable software in the application, isolate it from the remainder of the code, and determine whether it behaves exactly as you expect. Each Component is tested separately before integrating it into a service or services.</p>	C405.3	BTL1

9	<p>Define Process/Orchestration-level Testing</p> <p>Process/Orchestration testing ensures services are operating collectively as specified. This phase of testing would cover business logic, sequencing, exception handling and process decomposition (including service and process reuse).</p>	C405.3	BTL1
10	<p>Define Security Testing</p> <p>As SOA evolves and grows within your organization, the profile and necessity of Security testing will increase. Today, many organizations perform an inadequate amount of penetration testing at the very end of a project. SOA combined with Government and Regulatory compliance, will require Security testing activities to be incorporated into the entire project life cycle.</p>	C405.3	BTL1
11	<p>Explain Legacy System Adapter</p> <p>It is my personal opinion that any time a Service Component needs to interface to a legacy system, that an Adapter Pattern should be used. This Adapter performs a number of tasks. Its primary function is to convert to/from data formats spoken by the legacy system and the common data formats used by the Foundational Service Components. In order to perform the conversions, it may be necessary for an adapter to interface with many Service Components in order to perform data enrichment or break data apart into its fundamental data types.</p>	C405.3	BTL2
12	<p>Define Composite Service Component</p> <p>A Composite Service Component is a component which combines the functionality of one or more other Foundational or Composite Service Components. It may also encapsulate additional functional and data enrichment, business process, etc. In some cases, a Composite Service Component may be purpose built to support one and only one business process or application, where it makes sense to encapsulate a piece of reusable functionality on the server side instead of in the application.</p>	C405.3	BTL1
13	<p>Define WSDL?</p> <p>SOA services have self-describing interfaces in platform-independent XML documents. Web Services Description Language (WSDL) is the standard used to describe the services.</p>	C405.3	BTL1
14	<p>Define XSD?</p> <p>SOA services communicate with messages formally defined via XML Schema (also called XSD). Communication among consumers and providers or services typically happens in heterogeneous environments, with little or no knowledge about the provider. Messages between services can be viewed as key business documents processed in an enterprise.</p>	C405.3	BTL1

15	<p>Define UDDI?</p> <p>SOA services are maintained in the enterprise by a registry that acts as a directory listing. Applications can look up the services in the registry and invoke the service. Universal Description, Definition, and Integration (UDDI) is the standard used for service registry.</p>	C405.3	BTL1
16	<p>Explain about QOS?</p> <p>Each SOA service has a quality of service (QoS) associated with it. Some of the key QoS elements are security requirements, such as authentication and authorization, reliable messaging, and policies regarding who can invoke services.</p>	C405.3	BTL2
17	<p>Define Service Proxy?</p> <p>The service provider supplies a service proxy to the service consumer. The service consumer executes the request by calling an API function on the proxy. It then formats the request message and executes the request on behalf of the consumer. The service proxy is a convenience entity for the service consumer. It is not required; the service consumer developer could write the necessary software for accessing the service directly.</p>	C405.3	BTL1
18	<p>Define Service-Orientation and Interoperability</p> <p>Service-oriented computing, stating that services must be interoperable is just about as evident as stating that services must exist. Each of the eight principles supports or contributes to interoperability in some manner.</p>	C405.3	BTL1
19	<p>Define service Loose coupling</p> <p>The principle of Service Loose Coupling promotes the independent design and evolution of a service's logic and implementation while still guaranteeing baseline interoperability with consumers that have come to rely on the service's capabilities.</p>	C405.3	BTL1
20	<p>What is CCT</p> <p>CCT as an engineering tool giving one a rough measure of learnability and difficulty combined with a detailed description of user behavior. This can then be used by analysts employing their professional expertise</p>	C405.3	BTL1

21	<p>Define Business layer</p> <p>Business layer is responsible for supporting business process life cycle. Business process lifecycle consists of five stages: design, model, simulate, monitor, manage, and optimize business processes. Business layer users are either business analysts, or business managers. Business analysts create the initial drafts of the business processes, and business managers will manage and monitor those business processes.</p>	C405.3	BTL1
22	<p>Define Business services layer</p> <p>Business services layer holds orchestration and choreography engines under governance mechanisms to map business processes to composing Web services. Orchestration and choreography engines are the mapping enablers of business processes into executable services. Web services are stateless services that can not maintain business logic, operation flow, or user state; so, the need of an orchestration layer to include business logic is addressed. Orchestration and choreography engines maintain business process workflow logic, performance requirements, and system/user state. Business services layer has access to business rules repository</p>	C405.3	BTL1
23	<p>What is Session ID? A session ID is a unique identification</p> <p>string usually a long, random and alpha-numeric string, that is transmitted between the client and the server. Session IDs are usually stored in the cookies, URLs (in case url rewriting) and hidden fields of Web pages.</p>	C405.3	BTL1
24	<p>Define Composite Service Component</p> <p>A Composite Service Component is a component which combines the functionality of one or more other Foundational or Composite Service Components. It may also encapsulate additional functional and data enrichment, business process, etc. In some cases, a Composite Service Component may be purpose built to support one and only one business process or application, where it makes sense to encapsulate a piece of reusable functionality on the server side instead of in the application.</p>	C405.3	BTL1
25	<p>List out some characteristics of contemporary SOA. [NOV/DEC 2016]</p> <p>It is at the core of the service-oriented computing platform.</p> <p>It increases quality of service.</p> <p>It is fundamentally autonomous and based on open standards.</p> <p>It supports vendor diversity and fosters intrinsic interoperability.</p> <p>It promotes discovery and It promotes federation.</p> <p>It promotes architectural composability.</p> <p>It supports a service-oriented business modeling paradigm.</p>	C405.3	BTL1

	<p>It implements layers of abstraction.</p> <p>It is a building block.</p> <p>It is an evolution.</p> <p>It is still maturing.</p>		
26	<p>How loose coupling achieved in SOA? [NOV/DEC 2016]</p> <p>The principle of Service Loose Coupling promotes the independent design and evolution of a service's logic and implementation while still guaranteeing baseline interoperability with consumers that have come to rely on the service's capabilities.</p> <p>In order to ensure that the service contract is not tightly coupled to the service consumers and to the underlying service logic and implementation. This results in service contracts that could be freely evolved without affecting either the service consumers or the service implementation.</p>	C405.3	BTL1
27	<p>What is Text based communication & types? <u>APR/MAY2017</u></p> <p>Text-based communication is familiar to most people, in that they will have written and received letters. However, the style of letter writing and that of face-to-face communication are very different. The text-based communication in groupware systems is acting as a speech substitute, and, thus, there are some problems adapting between the two media.</p> <p>There are four types of textual communication in current groupware:</p> <p>discrete – directed message as in email. There is no explicit connection between different messages, except in so far as the text of the message refers to a previous one.</p> <p>linear – participants' messages are added in (usually temporal) order to the end of a single transcript.</p> <p>non-linear – when messages are linked to one another in a hypertext fashion.</p> <p>spatial – where messages are arranged on a two-dimensional surface.</p>	C405.3	BTL1
28	<p>Define XSD?</p> <p>SOA services communicate with messages formally defined via XML Schema (also called XSD). Communication among consumers and providers or services typically happens in heterogeneous environments, with little or no knowledge about the provider. Messages between services can be viewed as key business documents processed in an enterprise.</p>	C405.3	BTL1

29	<p>What are the fundamental parts of SOA framework?[NOV/DEC 2017]</p> <ul style="list-style-type: none"> • Enterprise service bus (ESB) • Service registry • Business processes • MDM hub • Data management 	C405.3	BTL1
30	<p>What is the responsibility of service(MAY/JUNE 2018)</p> <p>These components are responsible for realizing the functionality of services. The middle layer is the Service Layer, which is where exposed services (both individual and composite services) carrying out business functions reside.</p>	C405.3	BTL1

PART-B

Q.No	Questions	CO	Bloom's level
1	Explain how SOA can be compared to Client server architecture	C405.3	BTL5
2	Describe the principles of service orientation	C405.3	BTL5
3	Explain in detail about SOA components and how are they interrelate.	C405.3	BTL5
4	Decide how the 'golden rules' and heuristic help interface designers take account of cognitive psychology? Illustrate your answer with the design of Microsoft office word. Page no: 282 NOV/DEC2017	C405.3	BTL5
5	Explain about Hypertext and its characteristics.	C405.3	BTL5
6	What is distributed internet architecture? Compare it with SOA	C405.3	BTL4
7	Explain the anatomy of service oriented architecture	C405.3	BTL4
8	Describe how SOA can be compared to distributed internet architectures	C405.3	BTL5
9	Explain briefly about web services as component wrappers	C405.3	BTL6
10	Compare SOA with client server and distributed internet architectures. (MAY/JUNE 2018)	C405.3	BTL5
11	Propose the various principles of service orientation in detail. (MAY/JUNE 2018)	C405.3	BTL6

UNIT IV

Part – A

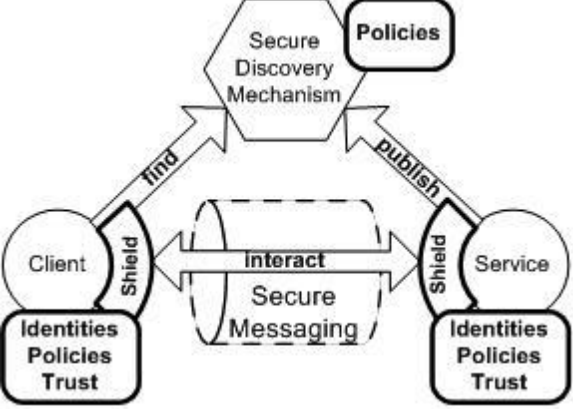
S. N o.	Question	Course Outco me	Blooms Taxano my Level
1	What is Web Services? Web Services can convert your applications into Web-applications. Web Services are published, found, and used through the C405.4 Web.		BTLI
2	What are Web Services? <ul style="list-style-type: none">• Web services are application components• Web services communicate using open protocols• Web services are self-contained and self-describing• Web services can be discovered using UDDI	C405.4	BTLI

	<ul style="list-style-type: none"> • Web services can be used by other applications • XML is the basis for Web services • 3. List the Web services platform elements: • SOAP (Simple Object Access Protocol) • UDDI (Universal Description, Discovery and Integration) • WSDL (Web Services Description Language) 		
3	<p>Define Web API</p> <p>Web API is a development in Web services (in a movement called Web 2.0) where emphasis has been moving away from SOAP based services towards Representational State Transfer (REST) based communications. REST services do not require XML, SOAP, or WSDL service-API definitions. Web APIs allow the combination of multiple Web services into new applications known as mashups.</p>	C405.4	BTLI
4	<p>Define Agents and Services</p> <p>A Web service is an abstract notion that must be implemented by a concrete agent. The agent 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. To illustrate this distinction, you might implement a particular Web service using one agent one day, and a different agent the next day with the same functionality. Although the agent may have changed, the Web service remains the same.</p>	C405.4	BTLI
5	<p>Define Soap message.</p> <p>A SOAP message is specified as an XML Information Set . While all SOAP message examples in this document are shown using XML 1.0 syntax, other representations MAY be used to transmit SOAP messages between nodes</p>	C405.4	BTLI
6	<p>Define Request-Response</p> <p>Request-Response is a pattern in which the service consumer uses configured client software to issue an invocation request to a service provided by the service provider. The request results in an optional response</p>	C405.4	BTLI
7	<p>Define Subscribe-Push</p> <p>A third pattern for interaction is called Subscribe-Push, shown in Figure 4-3. In this pattern, one or more clients register subscriptions with a service to receive messages based on some criteria. Regardless of the criteria, the externally visible pattern remains the same. Subscriptions may remain in effect over long periods before being canceled or revoked. A subscription may, in some cases, also register another service endpoint to receive notifications.</p>	C405.4	BTLI

8	<p>Data paging</p> <p>Some services automatically facilitate the paging of large data sets, enabling developers to focus on core application business logic instead of worrying about basic data management infrastructure.</p>	C405.4	BTLI
9	<p>Define Coordination Model</p> <p>The Coordination model provides integrated service to each key customer group. The integration results from sharing key data across the business units to present a common fact to the customer.</p>	C405.4	BTLI
10	<p>Define Atomic Service Transaction</p> <p>Atomic Service Transaction is the name of a design pattern authored by Thomas Erl and published as part of the SOA Design Patterns catalog. Within the catalog this pattern is further categorized as one of the Composition Implementation Patterns. The icon used to identify Atomic Service Transaction</p>	C405.4	BTLI
11	<p>List the Atomic Transaction Protocols</p> <p>A Completion protocol which is typically used to initiate the commit or abort states of the transaction. The Durable 2PC protocol for which services representing permanent data repositories should register. The Volatile 2PC protocol to be used by services managing non-persistent (temporary) data.</p>	C405.4	BTLI
12	<p>Define atomic transaction coordinator</p> <p>The atomic transaction coordinator plays a key role in managing the participants of the transaction process, and in deciding the transaction's ultimate outcome</p>	C405.4	BTLI
13	<p>Define Business activity</p> <p>Business activities consist of long-running, complex transactions involving numerous services. Hours, days, or even weeks can pass before a business activity is able to complete. During this period, the activity can perform numerous tasks that involve many participants.</p>	C405.4	BTLI
14	<p>Define Orchestration</p> <p>Refers to an executable business process that may interact with both internal and external Web services. Orchestration describes how Web services can interact at the message level, including the business logic and execution order of the interactions. These interactions may span applications and/or organizations, and result in a long-lived, transactional process. With orchestration, the process is always controlled from the perspective of one of the business parties.</p>	C405.4	BTLI
15	<p>Define Choreography</p>	C405.4	BTLI

	<p>More collaborative in nature, where each party involved in the process describes the part they play in the interaction. Choreography tracks the sequence of messages that may involve multiple parties and multiple sources. It is associated with the public message exchanges that occur between multiple Web services.</p>		
16	<p>Define Service Layers</p> <p>When building various types of services, it becomes evident that they can be categorized depending on: - the type of logic they encapsulate - the extent of reuse potential this logic has - how this logic relates to existing domains within the enterprise As a result, there are three common service classifications that represent the primary service models used in SOA projects: - Entity Services - Task Services - Utility Services</p>	C405.4	BTLI
17	<p>Define Application Services layer.</p> <p>Application services layer holds applications exposed as services, newly added services, and legacy applications wrapped by standard Web services interface. Services at Application Services layer are set of stateless Web services that perform certain task(s). Business process is the summation of C405.4 BTLI tasks performed by one or more services of application services layer at the sequence maintained by orchestration and choreography engines. Services of Application Services layer are reusable among different business processes, can be integrated in new applications, and can be extended address new business processes</p>		
18	<p>Service Orchestration</p> <p>Service Orchestration is nothing more than a fancy title for a program that calls web services during its execution. There's nothing magical or omnipotent about a —service orchestration!. In fact we'll learn later that any service orchestration implemented using BPEL is also just a standard, WSDL-defined web service in its own right. In context of Java, a service orchestration might be a Java class that not only constructs and calls methods on other classes but which also invokes web services using special Java classes and interfaces designed specifically for that purpose.</p>	C405.4 BTLI	the
19	<p>Define the Reply activity.</p> <p>A Reply is responsible for extracting data from a process variable and placing it into a WSDL message that is then sent back in response to the one accepted by the preceding Receive. In fact C405.4 BTLI syntactically, a Reply activity requires that the process have a Receive (or equivalent) somewhere earlier in the flow of the process. Here's the XML syntax our BPEL Process uses to reply to the loan approval</p>		
20	<p>ACID transactions</p> <p>They preserve the atomicity, consistency, isolation, and durability of the operation(s) encompassed by the transaction. This</p>	C405.4	BTLI

	helps preserve the integrity of our programs' logic execution as well as the integrity of the data managed by those programs. Java transaction managers are specifically designed to provide this functionality.		
21	<p>List the types of Choreography. APR/MAY 2018</p> <ul style="list-style-type: none"> • Abstract Choreography • "portable" choreography • Concrete Choreography 	C405.4	BTLI
22	<p>Define Abstract Choreography. The types of information that is exchanged, for example an order sent between a buyer and a seller The sequence and conditions under which the information is sent.</p>	C405.4	BTLI
23	<p>Define Portable Choreography. Detailed definitions of the physical structure of the information that is exchanged including the WSDL port types and operations Details of the technology to be used, for example, how to secure the messages and send them reliably Rules that express, as far as possible, the conditions that are used to control the sequence of exchange of information, in terms of, for example XPath expressions that reference data in the messages.</p>	C405.4	BTLI
24	<p>Define Concrete Choreography. Choreography, where all the details are specified that are required to send a message. This extends the definition in a Portable Choreography to include information about the "endpoints". This can include information such as: The URLs that are the destinations of the messages that are sent, and Other "endpoint" specific rules such as digital certificates to be used for securing messages</p>	C405.4	BTLI
25	<p>Define Orchestration service Orchestration services encapsulate the entire business process. For example, a service containing the entire flow of the "Employer Registration" business process is an orchestration service. The complete registration process identifying the employer type, determining liability and registering the employer in the database and various other steps.</p>	C405.4	BTLI

			
26	<p>Define SOAP SOAP is a method of accessing remote objects by sending XML messages, which provides platform and language independence. It works over various low-level communications protocols, but the most common is HTTP.</p>	C405.4	BTLI
27	<p>Explain about the operations in entity-centric? GetSomething UpdateSomething AddSomething DeleteSomething</p>	C405.4	BTLI
28	<p>What are Java Servlets? Servlets are Java technology's answer to CGI programming. They are programs that run on a Web server and build Web pages.</p>	C405.4	BTLI
29	<p>Define the term JSP EL EL means the expression language , it is a simple language for accessing data, it makes it possible to easily access application data stored in JavaBeans components. The jsp expression language allows a page author to access a bean using simple syntax such as \$(name).</p>	C405.4	BTLI
30	<p>Sketch the anatomy of a SOAP message. [NOV/DEC 2016] A SOAP message is an ordinary XML document containing the following elements – Envelope – Defines the start and the end of the message. It is a mandatory element.</p>	C405.4	BTLI

	<p>Header – Contains any optional attributes of the message used in processing the message, either at an intermediary point or at the ultimate end-point. It is an optional element.</p> <p>Body – Contains the XML data comprising the message being sent. It is a mandatory element.</p> <p>Fault – An optional Fault element that provides information about errors that occur while processing the message.</p> <p>All these elements are declared in the default namespace for the SOAP envelope</p> <pre><?xml version="1.0"?> <SOAP-ENV:Envelope xmlns:SOAP- ENV="http://www.w3.org/2001/12/soap-envelope" SOAP- ENV:encodingStyle="http://www.w3.org/2001/12/soap-encoding"> <SOAP-ENV:Header> </SOAP-ENV:Header> <SOAP-ENV:Body> <SOAP-ENV:Fault> </SOAP-ENV:Fault> ... </SOAP-ENV:Body> </SOAP_ENV:Envelope></pre>		
31	<p>List out some primitive MEPs. [NOV/DEC 2016]</p> <p>A common set of primitive MEPs are listed below</p> <ol style="list-style-type: none"> i. Request-response ii. Fire-and-forget 	C405.4	BTLI

	iii. Complex MEPs		
32	<p>List Down The Basic platform blocks APR/May 2018</p> <ol style="list-style-type: none"> 1.services 2.Business processes 3.Human actors 4. Events 5.Service Descriptions, Contracts, and Policies 6.Service Compositions 7.Programs 8.Information Items, Data Items, and Data Stores 	C405.4	BTLI
33	<p>Define ASP.NET Web Forms APR/May 2018</p> <p>ASP.NET Web Forms is a part of the ASP.NET web application framework and is included with Visual Studio. ... Web Forms are pages that your users request using their browser. These pages can be written using a combination of HTML, client-script, server controls, and server code</p>	C405.4	BTLI
34	<p>What is UDDI NOV/DEC 2017</p> <p>UDDI is an XML-based standard for describing, publishing, and finding web services. UDDI stands for Universal Description, Discovery, and Integration. UDDI is a specification for a distributed registry of web services</p>	C405.4	BTLI
35	<p>List any four pitfalls of SOA NOV/DEC 2017</p> <p>Implementation pitfalls</p> <ul style="list-style-type: none"> • Not Invented Here Syndrome • Versioning • Security <p>Architectural/design pitfalls</p> <ul style="list-style-type: none"> • Incorrect Granularity of Services • SOA does not solve complexity automatically • Big Design Upfront • Incorrectly applied Canonical Data Model • - Missing skills <p>Organizational pitfalls:</p>	C405.4	BTLI


	<ul style="list-style-type: none"> -Unclear ownership/Project based funding Ignoring culture when introducing SOA 		
PART B			
1	Write short notes on a. Service descriptions. (6) b. Atomic transaction .(5) c. Choreography.(5)	C405.4	BTL5
2	Discuss about different service layer in detail.(16)	C405.4	BTL5
3	Discuss about three service layers in detail.(16)	C405.4	BTL5
4	Describe in detail about messaging with SOAP.(16)	C405.4	BTL5
5	a. Write short notes on orchestration and choreography.(10) b. Briefly discuss on application service layer.(6)	C405.4	BTL 5
6	Explain briefly about technical requirements for orchestration and choreography.(16) Discuss in detail about Orchestration and Choreography [NOV/DEC 2016]	C405.4	BTL6
7	Briefly explain about: a. Service layer abstraction .(8) b. Application service layer.(8)	C405.4	BTL 6
8	Describe the process of messaging with SOAP and Atomic transaction.(16) Explain in detail about Atomic Transaction Process with suitable diagrams. [NOV/DEC 2016]	C405.4	BTL5
9	Explain about service layer configuration scenarios.(16)	C405.4	BTL6

10	Explain about service layer abstraction and orchestration service layer	C405.4	BTL6
11	Discuss on how SOA is related to the layers of the J2EE platform APR/MAY 2018	C405.4	BTL5
12	Mention the benefits of JAX-RPC in detail APR/MAY 2018	C405.4	BTL5
13	a) Show the WSDL document consisting of abstract and concrete parts that collectively describe a service endpoint. NOV/DEC 2017	C405.4	BTL5
14	How the challenge of coordinating messages in accomplished by Message exchange patterns ? NOV/DEC 2017	C405.4	BTL5

**UNIT V
PART A**

S. No	Question	Course Outcome	Blooms Taxonomy Level
1	<p>What is Service-oriented analysis?</p> <p>Service-oriented analysis establishes a formal analysis process completed jointly by business analysts and technology architects. Service modeling, a sub-process of service-oriented analysis, produces conceptual service definitions called service candidates. Iterations through the service-oriented analysis and service modeling processes result in the gradual creation of a service inventory blueprint.</p>	C405.5	BTL1

2	<p>Write the business-centric entry points? These business-centric entry points are: People—Productivity through people collaboration Process—Business process management for continuous innovation Information—Delivering information as a service.</p>	C405.5	BTL1
3	<p>Write the IT-centric entry points? These IT-centric entry points are: Reuse—Creating reusable functionality Connectivity—Underlying connectivity to support business-centric SOA</p>	C405.5	BTL1
4	<p>Define entity-centric business service. The entity-centric business service is agnostic to any one business process, they achieve "process logic independence" and therefore become highly reusable. And, because they represent a well-defined set of logic and data, they establish a level of abstraction and governance over a distinct business domain. This can significantly increase the agility with which business processes that rely on the composition of services can be altered in response to change.”</p>	C405.5	BTL1
5	<p>List out the component specification in Service modeling</p> <ul style="list-style-type: none"> Data Rules Services Configurable profile Variations 	C405.5	BTL1
6	<p>Define SOMA - Service Oriented Modeling and Architecture Service Oriented Modeling and Architecture refers to the general domain of service modeling that is necessary for the creation and design of Service Oriented Architecture. Service Oriented Modeling and Architecture covers a much larger scope and implements service oriented analysis and design via a specification, designation, and actualization of services, as well as the components that help realize those services and the flows that can aid in the process of building such services.</p>	C405.5	BTL1
7	<p>Define Resource and services The XML Web services architecture defines a standard mechanism for making resources available via XML messaging. Being able to access a resource by simply transmitting XML messages over standard protocols like TCP, HTTP, or SMTP</p>	C405.5	BTL1

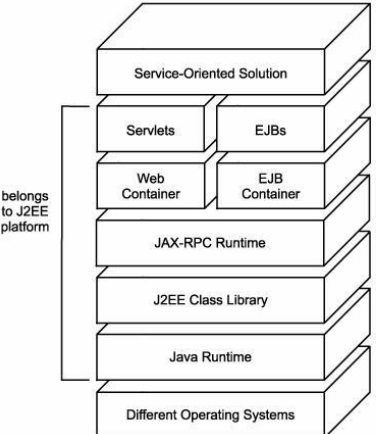
	greatly lowers the bar for potential consumers. The term "Web service" (or simply "service") typically refers to the piece of code implementing the XML interface to a resource, which may otherwise be difficult to access		
8	<p>Define Messages and operations</p> <p>A message exchange is also referred to as an operation. Operations are what consumers care about most since they're the focal point of interacting with the service.</p>	C405.5	BTL1
9	<p>Define WSDL</p> <p>The Web Services Description Language (WSDL) provides an XML grammar for describing these details. WSDL picks up where XML Schema left off by providing a way to group messages into operations and operations into interfaces. It also provides a way to define bindings for each interface and protocol combination along with the endpoint address for each one. WSDL plays an important role in the overall Web services architecture since it describes the complete contract for application communication. Although other techniques exist for describing Web services, the WS-I Basic Profile Version 1.0 mandates the use of WSDL and XML Schema for describing Web services. This helps ensure interoperability at the service description layer.</p> 	C405.5	BTL1
10	<p>What is the responsibility of the service?</p> <p>This provides the basic description of WHY the service should exist. Services are expensive. If you cannot stand in front of an executive and make a purely business-oriented case for the existence of a service, you need to rethink it. You are doing something wrong.</p>	C405.5	BTL1

11	<p>What rules does the service own?</p> <p>This provides the basic scope of the service. In a well devised, Enterprise SOA, you will have a rule implemented in a relatively small number of services (hopefully in one service) which makes it easier to change that rule. This means that you need to describe the collection of rules owned by a service.</p>	C405.5 BTL1	which
12	<p>What style of EAI are you implementing?</p> <p>The alternative is the "RPC (Remote Procedure Call)" style. If you are passing a block of self-describing and complete data to a service, and all that returns is "thanks... got it," then you are using the messaging style. If you are passing a command (with or without parameters) and are expecting either a set of data in response or an "OK... operation complete" message back, then you are using the RPC style. Note that RPC style services are more typical but, IMHO, less powerful because they assume a real-time binding between the interacting systems</p>	C405.5	BTL1
13	<p>Is the service idempotent?</p> <p>In other words, if a call to the service is duplicated, and instead of being called once, the service is called twice with identical parameters/payload, will the service detect the duplication and prevent any effects on the underlying systems? This is very important in messaging style services, but it turns up in RPC services as well. A service that provides idempotency is more loosely coupled than one that does not, but it also adds to the complexity of the service implementation.</p>	C405.5	BTL1
14	<p>What preconditions and postconditions apply to this service?</p> <p>Just as in use case development, you must be able to describe the factors that must be in place for this service to be used. Unlike typical use case development, however, you must describe the behavior of the service when these preconditions are not met. You also must describe the limitations and constraints of the service. For example, if a service is designed to be used only during a specific business process, then this must be described and included in the service design.</p>	C405.5	BTL1
15	<p>What actors may use this service and how will they be authenticated?</p> <p>This is an optimistic statement, because you (a) may not know, and (b) may not want to limit your implementation. However, you need to consider all of the actors who can use the business rule that you are encapsulating. If one of those actors use your service, you need to either find a suitable interaction where that actor can use the rule, or create another service that meets that actors needs. Even within the firewalls of the data center, it is imperative that the communications between systems be understood to be secure from mal-intentioned people.</p>	C405.5 BTL1	cannot

	If your answer is "pray," then you may want to consider a new line of work		
16	<p>List the stages of WS-Coordination</p> <p>Instantiation (or activation) of a new coordinator for the specific coordination protocol, for a particular application instance Registration of participants with the coordinator, such that they will receive that coordinator's protocol messages during (some part of) the application's lifetime. Propagation of contextual information between Web services that comprise the application. An entity to drive the coordination protocol through to completion</p>	C405.5	BTL1
17	<p>Define WS-Coordination context</p> <p>A coordination identifier with guaranteed global uniqueness for an individual coordinator in the form of a URI An address of a registration service endpoint where parties receiving a context can register participants into the protocol A TTL value which indicates for how long the context should be considered valid Extensible protocol-specific information particular to the actual coordination protocol supported by the coordinator</p>	C405.5	BTL1
18	<p>What data elements will be required in order to call the service?</p> <p>Do not define the format of the calling sequence. Define the semantics of the data element itself</p>	C405.5	BTL1
19	<p>What data elements will be returned by the service in its acknowledgement / receipt / return?</p> <p>Think of these questions as the services "data dictionary" but with more constraints. Data dictionaries describe data at rest. These points describe data in process.</p>	C405.5	BTL1
20	<p>Define task service</p> <p>A task service is a form of business service with a functional context based on a specific business process. As a result, task services are not generally considered agnostic and therefore have less reuse potential than other service models.</p>	C405.5	BTL1
21	<p>List out the stages to develop applications in WS-BPEL</p> <p>Denial - don't need it Coercion - management says we have to use it Elation - realization that it will solve all our enterprise application problems Depression - realization that it will not solve all our enterprise application problems yet</p>	C405.5	BTL1

	Enlightenment - understanding how - and when - to leverage the advantages of SOA, via BPEL (in this case, using your Java skills as a guide)		
22	<p>Define WS-Policy APR/MAY2018</p> <p>The Web Services Policy Framework (WS-Policy) provides a general purpose model and corresponding syntax to describe the policies of a Web Service. WS-Policy defines a base set of constructs that can be used and extended by other Web services specifications to describe a broad range of service requirements and capabilities.</p>	C405.5	BTL1
23	<p>Define Composable Architecture</p> <p>The Web service specifications (WS*) are designed to be composed with each other to provide a rich set of tools for secure, reliable, and/or transacted Web services. WS-Policy by itself does not provide a negotiation solution for Web services. WS-Policy is a building block that is used in conjunction with other Web service and application-specific protocols to accommodate a wide variety of policy exchange models.</p>	C405.5	BTL1
24	<p>Define Compact Policy Expression</p> <p>To express a policy in a more compact form while still using the XML Infoset, this specification defines three constructs: an attribute to decorate an assertion, semantics for recursively nested policy operators, and a policy reference/inclusion mechanism</p>	C405.5	BTL1
25	<p>Define Policy Expression</p> <p>To convey policy in an interoperable form, a policy expression is an XML Infoset representation of a policy. The normal form policy expression is the most straightforward Infoset; equivalent, alternative Infosets allow compactly expressing a policy through a number of constructs</p>	C405.5	BTL1
26	<p>List the Security Threads APR/MAY 2018</p> <ul style="list-style-type: none"> • Confidentiality • Man-in-the-middle • Spoofing • Denial of Service • Replay Attacks 	C405.5	BTL1
27	<p>List the Web Services Security Requirements APR/MAY 2018</p> <ul style="list-style-type: none"> • Authorization • Data Integrity and Data Confidentiality • Integrity of Transactions and Communications 	C405.5	BTL1

- Non-Repudiation
- End-to-End Integrity and Confidentiality of Messages
- Audit Trails
- Distributed Enforcement of Security Policies

<p>28 Define Secure Messaging</p>	<p>Secure Messaging ensures privacy, confidentiality and integrity of interactions. Digital signatures techniques can be used to help ensure non-repudiation. Techniques that ensure channel security can be used for securing messages. However, such techniques are applicable in a few limited cases. Examples include a static direct connection between a requester agent and a provider agent. For some applications, such mechanisms can be appropriate. However, in the general case, message security techniques such as encryption and signing of the message payload can be used in routing and reliable messaging.</p>	<p>C405.5</p>	<p>BTL1</p>
<p>29 List the layers of the J2EE platform as they relate to SOA.</p>		<p>C405.5</p>	<p>BTL1</p>
<p>30 List the layers of the J2EE platform as they relate to SOA.</p>	<p>J2EE solutions inherently are distributed and therefore componentized. The following types of components can be used to build J2EE Web applications:</p> <p>Java Server Pages (JSPs) Dynamically generated Web pages hosted by the Web server. JSPs exist as text files comprised of code interspersed with HTML.</p> <p>Struts An extension to J2EE that allows for the development of Web applications with sophisticated user -interfaces and navigation.</p> <p>Java Servlets These components also reside on the Web server and are used to process HTTP request and response exchanges. Unlike JSPs, servlets are compiled programs.</p>	<p>C405.5</p>	<p>BTL1</p>

	<p>Enterprise JavaBeans (EJBs) The business components that perform the bulk of the processing within enterprise solution environments. They are deployed on dedicated application servers and can therefore leverage middleware features, such as transaction support.</p> <p>While the first two components are of more relevance to establishing the presentation layer of a service-oriented solution, the latter two commonly are used to realize Web services</p>		
	<p>31 Give the step by step process in the service oriented analysis. [NOV/DEC 2016]</p> <ul style="list-style-type: none"> – Define a preliminary set of service operation candidates. – Group service operation candidates into logical contexts. These contexts represent service candidates. – Define preliminary service boundaries so that they do not overlap with any existing or planned services. – Identify encapsulated logic with reuse potential. – Ensure that the context of encapsulated logic is appropriate for its intended use. – Define any known preliminary composition models. 	C405.5	BTL1
32	<p>Write the syntax for getVariableData function in WS BPEL. [NOV/DEC 2016]</p> <p>getVariableData(variable name, part name, location path)</p>	C405.5	BTL1
33	<p>What is service modeling process NOV/DEC 2017</p> <p>SOMF is a service-oriented development life cycle methodology, a discipline-specific modeling process. It offers a number of modeling practices and disciplines that contribute to a successful service-oriented life cycle development and modeling during a project</p>	C405.5	BTL1
34	<p>Write any four attributes of invoke elements of BPEL NOV/DEC 2017</p>	C405.5	BTL1
1	Briefly explain about WSDL & SOAP basics in service oriented design.(16)	C405.5	BTL6
2	Describe in detail about entity-centric business service design in a step by step process.(16)	C405.5	BTL5

3	Explain the steps involved in service oriented design in detail .(16)	C405.5	BTL6
4	Explain the basics of web services description language in detail.(16)	C405.5	BTL6
5	Explain briefly about business-centric SOA	C405.5	BTL6
6	Explain in detail about service modeling guidelines.(16)	C405.5	BTL6
7	Describe application service design in a step by step process.(16)	C405.5	BTL1
8	.a.List out the objectives of service oriented design.(6) b.Explain about service oriented design process.(10)	C405.5	BTL1 BTL6
9	Explain entity-cenric and task-centric business service design in detail.(16)	C405.5	BTL6
10	a.Discuss about entity-centric business service design .(8) b.List out the steps for composing SOA.(8)	C405.5	BTL5
11	Identify the various steps involved in service oriented modeling elaborate them in detail. (16) [NOV/DEC 2016]	C405.5	BTL5
12	Illustrate in detail about the WS-BPEL with code snippets (16) [NOV/DEC 2016]	C405.5	BTL5
13	Describe the Web Services Security Requirements in detail. APR/MAY 2018	C405.5	BTL5
14	List out the security Threads in detail APR/MAY 2018	C405.5	BTL5
15	Classify service model logic as service operation candidates and service candidates with basic building block activities. NOV/DEC 2017	C405.5	BTL5
16	Demonstrate WS-Security framework in terms of the 'security' element with an example. NOV/DEC 2017	C405.5	BTL5

