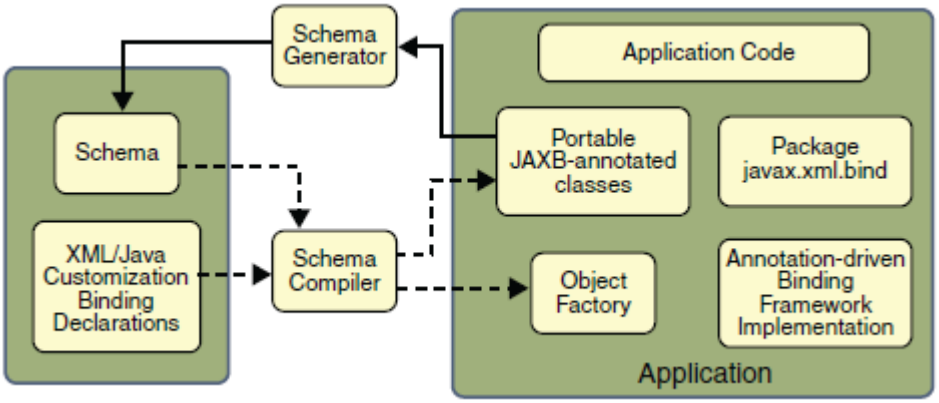
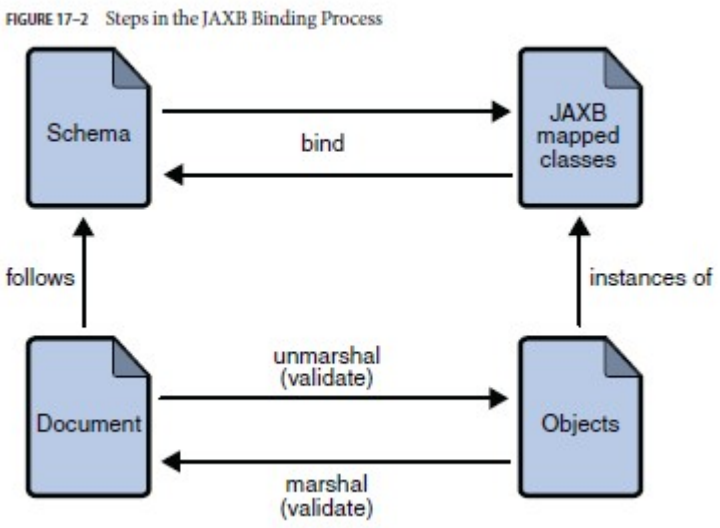
LIYANA ARACHCHIGE RANIL

SHORT NOTES / JAXB 2.0

1. JAXB stands for JAVA API for XML Binding
2. JAXB allows XML marshalling and un marshalling – JAVA-2-XML (Marshalling) , XML-2-JAVA (UnMarshalling)
3. JAXB is used heavily by JAX-WS
4. JAXB provides ways to generate XML content from a Java representation , to generate a Java representation from XML file , to generate XML schema from Java Objects
5. JAXB supports all W3C XML schema features (JAXB1.0 did not)
6. JAXB ARCHITECUTRE
   a.



7. JAXB contains , SCHEMA COMPILER , SCHEMA GENERATOR and BINDING RUNTIME FRAMEWORK
8. Steps in JAXB binding process

a. 

9. The GENERAL steps in JAXB data binding process are
    a. Generate Classes
    b. Compile Classes
    c. Un marshal
    d. Generate Content Tree
    e. Validate (Optional)
    f. Process Content (client application can modify the tree)
    g. Marshall
10. JAXB schema generator “schemagen” , schema compiler XJC
11. JAXB generated JAVA Content tree can be MODIFIED/UPDATED as well
12. UNMARSHALLING can be done not only from an XML document , but also from a InputStream object, a URL , or a DOM node
13. You can do SAX parsing and pass the event to JAXB for UNMARSHALLING

14. JAXB allows access to DATA in a NON-SEQUENTIAL manner. Unlike DOM based processing it doe not force you to navigate through the TREE
15. JAXB binding behavior can also be CUSTOMIZED
16. JAXB UNMARSHALLING
    a. JAXBContext context = JAXBContext.newInstance(“com.text.something”);
    b. Unmarshaller unmarshaller = contrext.createUnmarshaller();
    c. MyObject o = (MyObject) unmarshaller.unmarshal(new file(“foo.xml”));
17. JAXB MARSHALLER
    a. Marshaller marshaller = context.createMarshaller();
    b. Marshaller.marshal(o,new FileOutputStream(“foo.xml”));
18. JAXB VALIDATOR
    a. Validator v = context.createValidateor();
    b. If(v.validate(o)) System.out.println(“Error!!!”);