Birkbeck
University of London

MSc Electronic Commerce

Development of Internet Applications—Solutions

Friday 7th June 2002 (10:00–12:00)

1. (a) XML allows one to use more appropriate and meaningful element names to describe the information being exchanged. XML allows the use of a DTD to enforce constraints on the stucture of information and to specify many more data types than in HTML. A validating XML parser can then check that the information conforms to the given constaints - there is thus no need to write a separate program to check information integrity as would be the case for HTML. (6 marks)

   (b) Loosely speaking the expression returns the titles of sections which contain images. More specifically, the expression returns all title child elements of section elements which occur at any level in the document, as long as the section element has an image element nested at some level within it. (4 marks)

   (c)
```
<xsl:for-each select="author[not(position()=last())]">
  <xsl:value-of select="."/>,
</xsl:for-each>
<xsl:value-of select="author[position()=last()]"/>.
```
   (7 marks)

   (d) DOM defines an API for HTML and XML documents with the purpose of providing portability across web browsers. DOM defines a logical structure (model) of documents in which a document is modelled as a tree (or forest) of elements. Using DOM, programmers can build documents, navigate their structure and add, modify and delete elements and content. DOM is platform-neutral and language-neutral, with language bindings being defined for a number of programming languages. (8 marks)

2. (a) XML schema uses XML syntax while DTDs do not; it provides extensive data typing, especially for element content, while DTDs provide essentially only a string type for element contents; it is compatible with namespaces while DTDs are not really; it can use mixed content and enforce the order and number of child elements while DTDs cannot; it can enforce the number of child elements without also enforcing order in a simple way while this is complicated with DTDs. (5 marks)

   (b) Diagram should be of a tree of nodes with the sets of nodes comprising each of the 5 axes being indicated. "Partitioning the tree in 5 subtrees" means that the sets of nodes for each axis do not intersect and the union of the 5 sets gives all the nodes in the tree. (8 marks)

   (c)
```
<script language = "JavaScript">
   var num;
   // Display a prompt with a zero in it
   num = window.prompt("Enter an integer", "0");
   document.writeln("<table border = '1' align='center'>");
   var count = 0;
   while (count <= num)
   {
      // Display each line of the table,
      // each line is an integer and its square
      document.writeln("<tr><td>", count, "</td>
                        <td>", count*count, "</td></tr>");
      count++;
```

```
  }
  document.writeln("</table>");
</script>
```

(12 marks)

3. (a) Namespaces allow elements which have been defined with the same name but with different meanings to be disambiguated. This is especially useful and important in the area of e-commerce where data from multiple sources may be integrated. The different sources may use the same element names to mean different things. (4 marks)

   (b)
```
<xsl:template match="/supplier-list">
  <part-list>
  <xsl:for-each select="supplier/part">
    <part>
      <number><xsl:value-of select="number"/></number>
      <description>
      <xsl:choose>
        <xsl:when test="description">
          <xsl:value-of select="description"/>
        </xsl:when>
        <xsl:otherwise>
          n/a
        </xsl:otherwise>
      </xsl:choose>
      </description>
      <supplier><xsl:value-of select="../name"/></supplier>
    </part>
  </xsl:for-each>
  </part-list>
</xsl:template>
```

   (16 marks)

   (c) Active Server Pages (ASP), Java Server Pages (JSP), Common Gateway Interface (CGI), Java servlets, System-Side Includes (SSI). (5 marks)

4. (a) An XML entity is a named part of a document, irrespective of structural considerations. Essentially it provides a mechanism for naming and including document fragments. General entities are used in the body of a document, whereas parameter entities are used only within XML DTDs. An example of a general entity declaration is:

   ```
   <!ENTITY BBK "Birkbeck College, University of London">
   ```

   A reference to this entity is: `&BBK;`. (10 marks)

   (b) Responses in HTTP caching are assumed to have a certain lifetime. A response is fresh before its lifetime has expired and stale after its lifetime has expired. Expiry of responses is usually set by the origin server using either the Expires header (date and time) or the max-age directive of the Cache-Control header. Assigning an expiry time in the past, suggests tha every request should be revalidated. The origin server can force the revalidation of stale reponses by using the must-revalidate directive of the Cache-Control header. It can disallow caching by using the no-cache directive (e.g., for copyrighted material, or pay-per-view documents). Origin servers are not required to set expiry times so caches need to employ heuristic algorithms to assign them. For this, they can use the Last-Modified header in a resource and set the expiry, e.g., to 10% of the interval since the Last-Modified time. (15 marks)

5. (a) i.
```
<CD>
  <composer>Johannes Brahms</composer>
  <performance>
    <composition>Piano Concerto No. 2</composition>
  </performance>
</CD>
```
(4 marks)

ii.
```
<xsd:element name="CD">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="composer"     type="xsd:string"/>
      <xsd:element name="performance"  maxOccurs="unbounded">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="composition"  type="xsd:string"/>
            <xsd:sequence minOccurs="0">
              <xsd:element name="orchestra"  type="xsd:string"/>
              <xsd:element name="conductor"  type="xsd:string"/>
            </xsd:sequence>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
      <xsd:element name="length" type="xsd:timeDuration" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```
(12 marks)

(b) A URL is dependent on the host name where the resource is located and on the file structure on that host. The URL can easily be invalidated by host names changing, file structure changing, or the resource being moved. A URN should persist until it is explicitly deleted and should not be affected by the relocation or deletion of a particular resource. This is done by using an extra level of indirection. A URN comprises a Namespace IDentifier (NID) and a Namespace Specific String (NSS). A browser uses a Resolver Discovery Service (RDS) to find a resolver for a particular NID. The browser then sends the NSS to the resolver which returns the URL of the resource. The browser then used DNS in the usual way to map the URL to an IP address. (9 marks)

6. (a) i.
```
<!ELEMENT name (((first-initial|first-name),middle-initial?)?, last-name)>
```
(7 marks)

ii. This cannot be done using a DTD. In a DTD each element name can have only one content model associated with it, so there is no way that an element can have one content model when it is a child of one element and a different content model when it is the child of another element. (3 marks)

(b) Persistent connections send multiple request/response interactions over a single HTTP connection. They were introduced in HTTP/1.1 because of poor performance of HTTP/1.0 due to its use of separate connection for each request/response. The advantages of persistent connections include: saves CPU time and memory because of fewer TCP connections and fewer TCP control blocks, requests and responses can be pipelined (see below), network load is reduced because fewer packets are sent, and there is greater tolerance to HTTP version inconsistencies since errors can be reported to the client without the penalty of closing the TCP connection. In pipelining a client can make multiple requests without waiting for each response from the server. This results in better utilisation of the connection. A consequence of pipelining is that the server must send responses back in the same order as requests were made (since HTTP is stateless). (15 marks)