Problem	1	2	3	4	5	6	7	Total
Points:	14	36	10	16	20	8	10	114
Score:								

This homework assignment has 7 problems, for a total of 114 points.

1. (14 points) Of the following statements, identify all that hold about governance.
   A. Governance refers to the administration of a system according to the requirements imposed by a governing body
   B. Even changing the URL through which students register for courses at our university is an example of what could be a governance decision
   C. In abstract terms, a policy of a participant in a system reflects the autonomy of the participant
   D. Effective governance presumes an ability to conduct voting to elect the stakeholders who would take system-wide decisions on behalf of everyone else
   E. Governance, in principle, applies to the entire life cycle of an IT system
   F. Governance does not entail that all decisions are made through negotiation at run time; several important decisions could be "configured" ahead of time so little or no negotiation is needed at run time
   G. The so-called SOA governance is an element of IT governance
2. (36 points) Problems on architecture, governance, norms, and contracts.
   A. Policies make the most sense in settings where we have to make sure that whatever the system manager wants is carried out and users are prevented from acting against the manager's wishes
   B. To implement an organizational policy we must employ a policy engine
   C. Norms as we treat them in this course capture business relationships between autonomous parties as a way to specify their interactions in a meaningful manner
   D. A physical capability to perform some action is the most basic kind of norm
   E. An authorization is violated only if in its terminal state, its antecedent is true and its consequent is false
   F. A well-designed Org would not specify a commitment for a role that an agent playing that role is not planning to satisfy
   G. A well-designed Org would not specify that a role be both committed to doing something and prohibited from doing it
   H. We use prohibitions to specify restrictions on a party's behavior that *cannot* be enforced through the computational system alone and require some external process, potentially leading to sanctions
   I. Prohibitions as we treat them relate to an optimistic strategy for enforcing norms in that we assume each party will respect its prohibitions, so we give each party freedom to proceed though we may subsequently detect violations and penalize violators
   J. Often, a liability in one role's façade is a privilege in another role's façade
   K. A contract must include the specification of a resolution clause
   L. A contract can include the specification of a resolution clause
   M. A penalty clause in a contract is a kind of a sanction clause
   N. Once a contract clause is violated, the contract may either terminate in failure if it cannot be resolved or continue in its execution phase if it can be resolved
   O. The main advantage of using an architectural framework such as DoDAF is that it helps produce unique architectures for IT systems of nontrivial complexity

P. In DoDAF terms, our normative specifications would fall into the category of specifications called the Operational Views

Q. A policy expresses how a participant in a contract chooses its interactions with others, including whether it decides to break the terms of the given contract

R. A compensation clause in a contract describes how potential problems in carrying out the contract may be resolved

3. (10 points) Problems on metadata and XML.

A. Metadata is the general category of data about data, and includes markup as an important subclass

B. Metadata can be potentially valuable trying to correctly interpret documents that are not exchanged across enterprises but are merely stored and retrieved

C. If someone changes the namespace to which a URI refers, that change could potentially invalidate an XML document that references that namespace

D. Once a URI for a namespace becomes established, its meaning is irrelevant except that it is a well-known identifier

E. Using XML for representing all documents has the benefit that one enterprise can understand the documents produced by another enterprise

4. (16 points) [Worth 4 points each.] Of the following statements, identify all that hold about XQuery.

A. The following XQuery function computes all the leaves of a subtree rooted at its argument, $aNode

```
declare function local:leaf($aNode)
{
 $aNode//*[count(*)=0]
};
```

B. When invoked with $root as the document root, local:ancestor computes the ancestor-or-self axis for $aNode

```
declare function local:ancestor($root, $aNode) {
 local:anc($root, $aNode, ())
};

declare function local:anc($root, $this, $ancs) {
if ($this = $root)
 then ($this, $ancs)
 else local:anc($root,
                $this/parent::element(),
                ($this, $ancs))
};
```

C. When invoked with $root as the document root, local:ancestorish, which invokes local:anc as declared in Part B, computes the ancestor-or-self axis for its argument, $aNode

```
declare function local:ancestorish($root, $aNode) {
 local:anc($root, $aNode, ($aNode))
};
```

D. When invoked with $root set to the document root, local:ancestorr computes the ancestor axis for $aNode

```
declare function local:ance($root, $this, $ancs) {
if ($this = $root)
 then $ancs
 else local:ance($root,
                 $this/parent::element(),
                 ($this/parent::element(), $ancs))
};

declare function local:ancestorr($root, $aNode)
{
 local:ance($root, $aNode, ())
};
```

Listing 1: Singers nested within songs grouped by genre

```
<Songs>
 <rock>
  <Song lg="en">Hotel California<Sgr name="Eagles"/>
  </Song>
 </rock>
 <folk>
  <Song lg="cpe">Day O<Sgr name="Harry␣Belafonte"/>
  </Song>
  <Song lg="pa">Mera Dil Darda<Sgr name="Jagdish␣Prasad"/>
  </Song>
 </folk>
 <calypso>
  <Song lg="en">Jamaica Farewell<Sgr name="Harry␣Belafonte"/>
  </Song>
 </calypso>
</Songs>
```

5. (20 points) Problems on XPath (refer to Listing 1 as needed).
   A. At the document root of Listing 1, evaluating the XPath expression Songs/rock/Song[@lg='en'] finds the Song nodes for Hotel California and Jamaica Farewell
   B. At the document root of Listing 1, evaluating the XPath expression Songs/rock/Song[@lg='cpe']/text()[1] finds the text node containing "Day O"
   C. At the document root of Listing 1, evaluating the XPath expression Songs/rock/Song[@lg or text()] finds a Song node that has an attribute lg or encloses some text
   D. The XPath expression //text()/text() finds the longest words within a text node; when evaluated at the document root of Listing 1, its outputs would include "California," "Day," and so on
   E. At the text node containing "Mera Dil Darda" in Listing 1, evaluating the XPath expression ./Sgr finds the Sgr node for Jagdish Prasad
   F. Evaluating the XPath expression Songs//Song[@lg='cpe']/descendant::node() at the document root of Listing 1, finds an empty node sequence
   G. At the rock node in Listing 1, the expression following::element()[1] finds the folk node
   H. At the calypso node in Listing 1, the expression preceding-sibling::text()[3] finds a text node containing whitespace
   I. At the Songs node in Listing 1, the expression //Song/*[3.9] yields no result
   J. At the Songs node in Listing 1, the expression //Song/*[1.9] yields the folk node and nothing else

6. (8 points) Problems on XQuery.
   A. Evaluating the following XQuery query on Listing 1 identifies both the rock and calypso genres

```
for $song in doc($input)//Song
where every $songg in $song/../Song satisfies ($songg/@lg='en')
return <enGenre gName='{name($song/..)}'/>
```

   B. Evaluating the following XQuery query on Listing 1 identifies only the rock genre

```
for $genre in doc($input)/Songs/*
where some $song in $genre/Song satisfies
                                  (fn:contains($song/Sgr/@name, "s"))
return $genre
```

C. Any XQuery query that includes nested FLWOR expressions must have exactly one return for each such nested FLWOR expression

D. The following XQuery function yields the same result as computing self::node() on $aNode

```
declare function local:selflike($aNode)
{
 $aNode
};
```

7. (10 points) Problems on XPath.

A. At any context node, evaluating parent::node()/child::node() produces the same result as evaluating child::node()/parent::node()

B. At any context node that is an element, evaluating parent::element()/child::element() produces no larger a node sequence than evaluating child::element()/parent::element()

C. At any context node that is a comment, evaluating parent::node()/child::node() always produces a strictly larger set of nodes than evaluating child::node()/parent::node()

D. For any context node, self::element() equals itself

E. An XPath query such as parent::node()/child::node() may never terminate, depending on the particular XML document and context node at which it is evaluated