TU Vienna

2007 Winter Semester

End of Semester Examination

ESW

Einführung ins Semantic Web

Time: 2.0 Hours
Date of Examination: March 10th, 2008

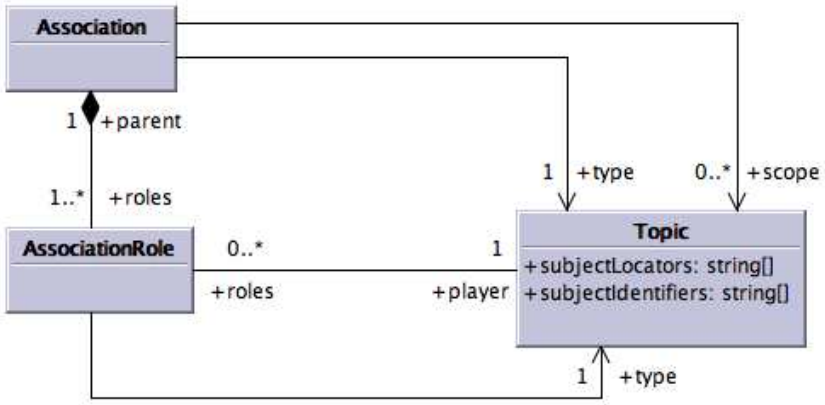
SID: .....

Instructions to the Candidate:

- **Please ensure that your student id number is filled in on this cover!**
- Answer all questions in this booklet in the spaces provided.
- The size of the empty space reflects roughly the expectation on the length of your answer.
- You may use any course material.
- For multiple-choice questions any number of correct answers is possible (zero, 1, ..., all).
- Text questions must be answered in a few sentences.
- If you run out of space, you can use the empty page on the left.

Question 1 (11%)

Write a Java (Python, Ruby, Perl) method which creates RDF triples in N3 text format from a Topic Map association which is handed in as a parameter.



For this purpose you can (should) assume that your language infrastructure is so clever that it allows you to navigate through an association object `a` like this:

`a.type` gives you the type topic

`a.roles` gives you the list of roles

`a.roles[1].player` gives you the player for the 2nd role

`a.roles[1].type` gives you the type topic of that role

For a given topic `t` you can use `t.subjectLocators` and `t.subjectIdentifiers` to get a list of URIs for subject locators and identifiers, respectively. You may ignore the scope component in an association.

Use one example association of type `employment` and roles `position`, `employer` and `employee` to illustrate your expected output.

Question 2 (15%)

Given an XML document (together with its RelaxNG schema):

```
<students>
  <student ID='8182838'>
     <name>James Bond</name>
     <shoesize>42</shoesize>
  </student>
  <student ID='8182839'>
     <name>Maxwell Smart</name>
     <haircolor>green</haircolor>
     <works-with IDREF='8182838'/>
  </student>
</students>
```

```
element students {
  element student {
     element name { text }
     ( element shoesize  { text }
     | element haircolor { text } )
     element works-with { }?
  }*
}
```

Accordingly, a `students` container can contain any number of `student` nodes. Each must have one `name` and alternatively a `shoesize` or a `haircolor`. Optionally there is a `works-with` subelement which links to another student.

- Create an RDF(S) schema in N3 to reflect this structure.

- Create (on the left page) a graph for the RDF instance data.

- Write a SPARQL query (including proper namespaces) to find all student pairs where one student cooperated with another student. Can you suppress in SPARQL duplicates, i.e. those pairs where there is a reciprocal `works-with` relationship in both students?

Question 3 (10%)

Formalize each of the following statements into description logic (DL). When a particular statement (or a part of it) cannot be fully expressed in DL, explain why not.

- `sacklpicker` is not a `CAT`.

- Everyone who does not `love(s)` at least one `CAT` is a `DOGLOVER`.

- If a `PERSON` `owns` a particular `CAT`, then that person is also owned by that cat.

- `CAT`s only love things which are `part-of` a `CHICKEN`.

- Some `CAT`s are also `DOG`s.

Question 4 (15%)

Given the following RDF(S) definition, create a database schema for a relational database (UML, Entity-Relationship diagrams, ...). Record any incompatibilities in semantics between the two.

```
:Person    rdf:type    rdfs:Class .
:birthdate rdf:type    rdf:Property, rdfs:range :Person .
:name      rdf:type    rdf:Property, rdfs:range :Person .

:Course    rdf:type    rdfs:Class .
:title     rdf:type    rdf:Property, rdfs:range :Course .
:code      rdf:type    rdf:Property, rdfs:range :Course .

:Student   rdfs:subClassOf :Person .
:matrikel  rdf:type    rdf:Property, rdfs:range :Student .

:enrolled  rdf:type    rdf:Property,
           rdfs:range  :Student,
           rdfs:domain :Course .
```

Question 5 (12%)

For which of the following T-boxes is it true that $\texttt{BusDriver} \sqsubseteq \texttt{Driver}$ if we also know that $\texttt{Bus} \sqsubseteq \texttt{Vehicle}$?

- $\texttt{BusDriver} \equiv \texttt{Person} \sqcap \exists \texttt{drives.Bus}$
  $\texttt{Driver} \equiv \texttt{Person} \sqcap \exists \texttt{drives.Vehicle}$

- $\texttt{BusDriver} \sqsubseteq \texttt{Person} \sqcap \exists \texttt{drives.Bus}$
  $\texttt{Driver} \equiv \texttt{Person} \sqcap \exists \texttt{drives.Vehicle}$

- $\texttt{BusDriver} \equiv \texttt{Person} \sqcap \exists \texttt{drives.Bus}$
  $\texttt{Driver} \sqsubseteq \texttt{Person} \sqcap \exists \texttt{drives.Vehicle}$

Question 6 (15%)

Translate the following FOL formulas into DL:

- $\forall d \in \texttt{DOG} : \forall c \in \texttt{CAT} : d \neq c$

- $\forall v \in \texttt{VEGETARIAN} : \forall f : \texttt{eats}(v, f) \Rightarrow \neg(\texttt{ANIMAL}(f) \wedge \exists f' \in \texttt{ANIMAL} : \texttt{part}(f, f'))$

Translate the following DL formulas into FOL (predicate logic):

- $\texttt{CATLOVER} \equiv \exists \texttt{loves.CAT} \sqcap \texttt{PERSON}$

- $\texttt{CATHATERS} \sqsubseteq \texttt{PERSON} \sqcap \neg\texttt{CATLOVER}$

- $\forall \texttt{loves.CAT} \sqsupseteq \exists \texttt{hates.DOG}$

Question 7 (10%)

Given the following RDF graph, the fact that `s` is a subproperty of `r` and that `r` is a subproperty of `t`, and the fact that `r` is symmetric and transitive, what is the graph an RDF(S) reasoner would 'see' (i.e. the graph which is entailed by the RDF(S) semantics)?

```
:a :r :b .
:b :r :c .
:c :r :d .
:e :s :d .
:e :t :f .
```

Question 8 (12%)

The following T-box has some highly undesirable characteristic. Which is it? Explain what happens. Note: `Wien` is just an arbitrary instance, nothing wrong with that.

$$\begin{aligned} \texttt{WIENER} &\equiv \exists \texttt{hauptGemeldet}.\{:Wien\} \sqcap \texttt{PERSON} \\ \texttt{NICHT}-\texttt{WIENER} &\equiv \neg \texttt{WIENER} \\ \texttt{NICHT}-\texttt{WIENER} &\sqsubseteq \texttt{PERSON} \end{aligned}$$

End of Paper.