Homework 1

Mattias Cederlund, mcede@kth.se

Task 1 – Schema and document generation

I created the 5 schemas, TranscriptSchema.xsd, ShortCvSchema.xsd, EmploymentRecordSchema.xsd, CompanyInfoSchema.xsd and ApplicantProfileSchema.xsd by hand. Then I used various xml web services to generate sample documents from the schemas (http://www.xsd2xml.com/). I filled the sample documents with data and used yet another web service to validate the document to the schemas to make sure they were valid (http://www.utilities-online.info/xsdvalidation).

All schemas specify a targetNamespace and use elementFormDefault=”qualified” to enforce use of namespace in the sample documents. Although I did not find any way to define multiple namespaces in a schema, so the ApplicantProfile use only one namespace for the entire document.

Task 2 – Mapping program

My plan when creating the mapping program was to parse one document with each method and create intermediary documents of a format which I could merge into a single document, the finished ApplicantProfile.

I parsed the ShortCv using DOM and then constructed a new document with DOM of the correct structure used in the Applicant profile. I then saved the intermediary xml document to a file.

The Transcript was parsed using SAX and then rebuilt with the correct structure using DOM (since SAX is only for reading documents and not for creating them). The intermediary document was saved to an xml file.

The employment record used in the ApplicantProfile consists of a mix of both the EmploymentRecord and the CompanyInfo, which I previously had in two separate documents. I generated Java models using JAXB for both schemas and I also created an additional schema for the desired employment record output and created a Java model for it using JAXB. I then created a Java object of the desired output and populated it from the Java objects unmarshaled from the xml files. The output was then marshaled to a xml file, the third intermediary document.

The three documents was then merged using XSLT. After merging I checked the document against the schema I created in the first task, and it validated successfully.

Task 3 – Only use XSLT

The mapping using XSLT was the most straight forward of all. The only two times it wasn't was when calculating the GPA and when mapping the companies to the employment record.

The GPA was calculated using

```
<xsl:value-of select="sum(document('Transcript.xml')//t:Grade) div
count(document('Transcript.xml')//t:Grade)" />
```

Initially I tried using avg(), but it wouldn't work.

To match companies with the employment record I quickly realized I would need to find the company entry with the same name as the employment record entry. Although, to make the matching work I had to declare a variable using

```
<xsl:variable name="currentCompanyName" select="e:CompanyName" />
```

and then select the appropriate values by using

```
<xsl:value-of select="document('CompanyInfo.xml')//c:Company[c:Name = $currentCompanyName]/c:CompanyNumber" />
```

After transforming the documents using XSLT I was left with a document that validated successfully with the ApplicantProfile schema.

Running instructions

The source code is provided as an Eclipse project. To run it, import it to Eclipse and run normally.

Finding the deliverables

The schemas and documents from task 1 is located in the xml folder.

The result from task 2 is provided in xml/ApplicantProfileMergedResult.xml.

All intermediary xml documents are prefixed with Intermediary and are located in the xml folder.

The result from task 3 is provided in ApplicantProfileResult.xml.