ID2208 Homework 1 - XML Processing

Andreas Hallberg
KTH Royal Institute of Technology
CINTE2010 / TSEDM 2013
Email: anhallbe@kth.se

I. Introduction

This report presents the concepts and the implementation of a simulated ”Employment Service Company”, which will be referred to as ESC. The task of the company is to create an *applicant profile* of a job seeker, which can be used to match him/her with companies that are looking to fill available positions. The service will gather information about the applicant from different services, such as the CV, university transcript, employment service etc. This simulation will assume that such services exist and they provide information in the form of XML documents, which will be parsed and processed in order to generate the applicant profile. XML sample documents are generated and validated against their respective schema documents. The processing of the documents (”The Profiler”) is performed using four different java implementations of XML processing: DOM, SAX, JAXB and XSLT.

II. XML Schemas

An XML schema describes how a document should be constructed, which elements it should contain, in which order they should appear and the format of their values and attributes. There are four schemas for the documents provided as input to the simulation:

- **Transcript.xsd** This describes a university transcript that contains the elements: *name, university, degree, year of graduation*, and a list of courses. A course has the elements *name, code, credits* and *grade*.
- **CV.xsd** Represents a CV that the applicant provides. It contains the name and contact information of the applicant, as well as a short presentation (personal letter) and a list of references. Restrictions are used to make sure that elements such as the *email* is correctly formatted.
- **EmploymentRecord.xsd** The Employment Record contains a list of previous (or current) employments of the applicant. An employment consists of the *name* of the company, *start*- and *end* dates of the employment and the monthly *salary* of the employee.
- **CompanyInfo.xsd** This document contains information about a company (as the name implies...) that the applicant may or may not have worked for. Among other elements it contains a list of available positions which the ESC can use to find suitable matches for the applicant.

The **ApplicantProfile** schema describes the resulting XML document. It contains almost all of the elements of the input documents and with similar structures. It also contains the GPA of the applicant that has been calculated by the ECS.

III. Sample XML documents

To simulate the ESC a number of sample documents were created, formatted and validated according to their respective schema. The documents contain information about a job applicant named Ronald Weasley. Ronald provides the ESC with a CV, the rest of documents (A transcript from Hogwarts School of Witchcraft and Wizardry, company information about Cauldrons Inc. and Ronald’s employment record) are provided as files in the local file system. All of the sample documents and schemas are located in the **src/pws/hw1/xml** directory.

IV. The Profiler

The Profiler is implemented in the pws.hw1.Profiler class (this is where the main method is implemented). At this time the program takes no arguments, and the location of the XML sources is chosen at compile-time. When the program is compiled it can be executed with ”java Profiler”, and the resulting applicant profile will be written to the XML file **src/pws/hw1/xml/applicant-profile-ronald.xml**. The Profiler will process different parts of the profile through the use of static parsing methods provided in *pws.hw1.dom/sax/jaxb/xslt*. DOM is used to parse the CV. SAX is used to parse the transcript and employment record. JAXB is used to parse (unmarshal) the company info and to marshal (generate) the resulting applicant profile.

Once the documents have been transformed into appropriate java objects, which are processed in the Profiler, the resulting profile is marshalled into a new document. This document is transformed using XSLT, where the GPA of the applicant is calculated and inserted in a new element, the result can be seen in the file **applicant-profile-ronald.xml**.

The implementation of the actual XML parsing does not make for an interesting report, but the code and documentation is pretty self-explanatory.