Distributed Programming II

A.Y. 2019/20

Lab 1

The material for this lab is in the *.zip* archive where you have found this file. Extract the archive to an empty directory that you will use as your working area and that we will call `[root]`.

In the archive you will find the files of the bank account and bibliography exercises as presented in the classroom: files `account.xsd`, `account.xml`, `biblio.xsd`, `biblio.xml` in the `[root]/xsd` folder.

Purpose

The aim of this Lab is to experiment with XML data format design. The exercises give the opportunity to apply the knowledge acquired about XML schema and the design guidelines presented in the course.

Exercise 1

1. Starting from the bank account sample schema (file `[root]/xsd/account.xsd`), design an extended version of it (to be saved as `[root]/xsd/accounts.xsd`) that satisfies the following additional requirements:
   - A single instance XML document must be able to store the information of *multiple* bank accounts, each one with all the information considered for the original bank account exercise.
   - Accounts must be uniquely identified in the document by their account number.
   - Account holders must be uniquely identified in the document by their fiscal ID, which is an additional attribute of each account holder, made of exactly 16 alphanumeric characters.
   - Within each account there cannot be two `yearOperation` elements referring to the same year (but this must be possible for `yearOperation` elements belonging to different accounts).
   - A single account holder can be a holder for multiple accounts.
2. Check that the new schema is valid. Then check that it works as expected by generating an XML file (to be saved as `[root]/xsd/accounts.xml`) that references the designed schema locally and that is valid with respect to the new schema. Then, test the schema by trying to introduce errors into the XML file and by checking that they are properly detected by the validation engine. Fix any errors that you found in the schema.

Exercise 2

1. Create another different solution of the bibliography exercise solved in the classroom, by exploiting the *complex type derivation mechanism (extension/restriction)* of the XML schema language. The new schema, which may use a different organization of XML

elements, must be saved as [root]/xsd/biblio_e.xsd.

2. Check that the new schema is valid. Then check that it works as expected by generating an XML file (to be saved as [root]/xsd/biblio_e.xml) that references the designed schema locally and that is valid with respect to the new schema. Then, test the schema by trying to introduce errors into the XML file and by checking that they are properly detected by the validation engine. Fix any errors that you found in the schema.

Exercise 3

1. Starting from the solution of the bibliography exercise developed in the classroom, or from the solution of exercise 2 (at your choice), create a new schema that re-uses the old one but that also satisfies the following additional requirements:
   - The title of a journal is constrained to be unique in the whole document, i.e. there cannot be two journals with the same title in the same XML document
   - In addition to article and book, an item can also be a book chapter, which is characterized by:
     - The book this chapter is part of. If the bibliography contains a book chapter, it must also contain the book this chapter is part of.
     - The number that uniquely identifies the chapter inside the book (there cannot be two book chapters that belong to the same book and that have the same number)
   - The title of a book chapter is unique inside the book it belongs to, i.e. there cannot be another book chapter in the same book with the same title.
2. The new schema must be saved as [root]/xsd/biblio_3.xsd.
3. Check that the new schema is valid. Then check that it works as expected by generating an XML file (to be saved as [root]/xsd/biblio_3.xml) that references the designed schema locally and that is valid with respect to the new schema. Then, test the schema by trying to introduce errors into the XML file and by checking that they are properly detected by the validation engine. Fix any errors that you found in the schema.

Submission instructions

The solutions of the exercises in this Lab will be submitted along with the solutions of Labs 2 and 3. For exemption, exercises 1 and 2 are mandatory, while exercise 3 is optional.

A submission will be accepted as valid only if:

- the files [root]/xsd/accounts.xsd and [root]/xsd/biblio_e.xsd exist and are valid XML Schema documents;
- the file [root]/xsd/accounts.xml exists, it references the schema [root]/xsd/accounts.xsd locally, and it is valid against such schema.
- The file [root]/xsd/biblio_e.xml exists, it references the schema [root]/xsd/biblio_e.xsd locally, and it is valid against such schema.

The validity of an XML file or schema can be checked by any XML validation program. For example, it can be checked by the Eclipse validate command.