jInfer BasicXSDExporter Module Description

Michal Klempa, Mário Mikula, Robert Smetana, Michal Švirec, Matej Vitásek
Advisors: RNDr. Irena Mlýnková, Ph.D., Martin Nečaský, Ph.D.

Praha, 2011

Target audience: developers willing to extend jInfer, specifically hack the XSD export.

Responsible developer:	Mário Mikula
Required tokens:	none
Provided tokens:	cz.cuni.mff.ksi.jinfer.base.interfaces.inference.SchemaGenerator
Module dependencies:	Base
Public packages:	none

1 Introduction

This is an implementation of a *`SchemaGenerator`* exporting the inferred schema to XSD, providing basic features of the language.

2 Structure

The main class implementing `SchemaGenerator` inference interface and simultaneously registered as its service provider is `SchemaGeneratorImpl`. Process of export consists of two phases described in detail in later sections:

1. Preprocessing
2. Own export to string representation of XSD

Method `start` first creates instance of `Preprocessor` class supplied by rules (elements) it got in the simplified grammar on input. Phase of preprocessing is done by creating that instance (calling its constructor) and its purpose is to discover information such which elements should be globally defined and which element is the top level element (TODO rio how is this element called in XML??).

Afterwards, it recursively traverses global elements followed by other elements starting at the top level element and for each it creates element's XSD string representation.

2.1 Preprocessing

TODO rio

2.2 Own export

TODO rio

Code exporting attributes is in `attributeToString()`. First thing this method does is to assess the domain of a particular atribute: this is a map indexed by attribute values containing number of occurences for each such attribute. Type definition of an attribute is generated in the `DomainUtils.getAttributeType()` method. Based on a user setting, this might decide to enumerate all possible values of this attribute using the `(a|b|c)` notation, otherwise it just returns `#CDATA`.
Attribute requiredness is assessed based on `required` metadata presence. If an attribute is not deemed required, it

might have a default value: if a certain value is prominent in the attribute domain (based on user setting again), it is declared default.

2.3 Preferences

All settings provided by *BasicDTDExporter* are project-wide, the preferences panel is in `cz.cuni.mff.ksi.jinfer.basicdtd.p` package. As mentioned before, it is possible to set the following.

- Maximum attribute domain size which is exported as a list of all values (`(a|b|c)` notation).
- Minimal ratio an attribute value in the domain needs to have in order to be declared default.

3 Data flow

Flow of data in this module is following.

1. `SchemaGeneratorImpl` topologically sorts elements (rules) it got on input.
2. For each element, relevant portion of DTD schema is generated.
3. String representation of the schema is returned along with the information that file extension should be "dtd".

[Aho96]

References

[Aho96] H. Ahonen. *Generating grammars for structured documents using grammatical inference methods*. PhD thesis, Department of Computer Science, University of Helsinki, Series of Publications A, Report A-1996-4, 1996.