Interopérabilité des données et des connaissances

Principes généraux du langage XML

Mauro Gaio

Université de Pau et des Pays de l'Adour
UFR S&T de Pau
mauro.gaio@univ-pau.fr

Plan

Récupérer le document resume_DTD.pdf sur e-learn …

Forme Générale

- XML « (meta-)langage extensible de balisage » : langage informatique de balisage générique

Forme Générale

- XML « (meta-)langage extensible de balisage » : langage informatique de balisage générique
- Il est utiliser pour stocker/représenter/transférer des données dans un format : « texte parsable » i.e. analysable par un parseur autonome

Forme Générale

- XML « (meta-)langage extensible de balisage » : langage informatique de balisage générique
- Il est utiliser pour stocker/représenter/transférer des données dans un format : « texte parsable » i.e. analysable par un parseur autonome
- afin de permettre une interopérabilité étendue

Forme Générale

- XML « (meta-)langage extensible de balisage » : langage informatique de balisage générique
- Il est utiliser pour stocker/représenter/transférer des données dans un format : « texte parsable » i.e. analysable par un parseur autonome
- afin de permettre une interopérabilité étendue
- les données sont obligatoirement organisées dans des structures arborescents

Forme Générale

```
<?xml version=''1.0'' encoding=''utf-8''?>
<!-- ici un prologue peut être ajouter -->
<C>
    <H/>
<H> Du texte simple
        <P chemin="http://www.oxygenxml.com/">
            encore du texte simple, mais comme ces 2 zones
            de texte sont  'parsables' certains caractères
            doivent être échappés, par exemple :
            le caractère 'supérieur' s'écrit &gt; ou &#62;
        </P>
    </H>
</C>
```

Spécification d'un DSL données en XML

Document Type Definition (DTD)

- spécifier explicitement l'alphabet auxiliaire
- représenter certaines règles de dérivation indépendamment de l'application

```
    <?xml version="1.0" encoding="UTF-8"?>
<!ENTITY titre "Un␣exemple␣simple␣de␣DTD" >
<!-- décaration d'une constante (abondonner en xml-schema)-->
<!ELEMENT C (H)+>
<!ELEMENT H (#PCDATA|H|P)*>
<!ELEMENT P (#PCDATA)>
<!ATTLIST P chemin CDATA #REQUIRED>
```

Une instance générée à partir du DSL

```
<?xml version=''1.0'' encoding=''utf-8''?>
<!-- <!DOCTYPE C SYSTEM "ex-simple.dtd"> -->

<!DOCTYPE C PUBLIC "http://erig.univ-pau.fr/DTD/" "ex-simple.dtd">
<C>
    <H/>
<H> &titre;
        <P chemin="http://www.oxygenxml.com/">
            &titre;
        </P>
    </H>
</C>
```

Une instance générée à partir du DSL

- la spécification avec une DTD permet : de **désigner** des parties d'un contenu (chaîne de caractère) avec des noms et de **hiérarchiser** ces noms
- la spécification avec une DTD consiste à considérer que quelque soit la structure initiale de l'information elle doit être traduite en un **arbre**

Une instance générée à partir du DSL

- la spécification avec une DTD permet : de **désigner** des parties d'un contenu (chaîne de caractère) avec des noms et de **hiérarchiser** ces noms
- la spécification avec une DTD consiste à considérer que quelque soit la structure initiale de l'information elle doit être traduite en un **arbre**

à retenir :

- un élément est caractérisé à la fois par son nom et par sa place dans l'arbre
- tout traitement se fonde sur des outils permettant de choisir des éléments par :
  - leur nom
  - ou leur position
  - ou par les 2 dimensions

Une instance générée à partir du DSL

- la spécification avec une DTD permet : de **désigner** des parties d'un contenu (chaîne de caractère) avec des noms et de **hiérarchiser** ces noms
- la spécification avec une DTD consiste à considérer que quelque soit la structure initiale de l'information elle doit être traduite en un **arbre**

à retenir :

- un élément est caractérisé à la fois par son nom et par sa place dans l'arbre
- tout traitement se fonde sur des outils permettant de choisir des éléments par :
  - leur nom
  - ou leur position
  - ou par les 2 dimensions

Cela suffit pour le moment . . .

Document XML bien formé et valide

- la structure d'un document XML est définissable et "validable" par un **schéma**
- un document XML est **entièrement transformable** dans un autre document XML.

- *Document Type Definition* DTD
- XML-Schema (W3C)
- RelaxNG
- schematron
- ...

Structure définissable et “validable” par un *schéma*

```
 <?xml version=''1.0'' encoding=''utf-8''?>
<!ELEMENT cinemas (cinema)+>
<!ELEMENT cinema (nom, adresse, ville,departement, (salle)+)>
<!ELEMENT nom (#PCDATA)>
<!ELEMENT salle ((titre, seances)+)>
<!ATTLIST salle num CDATA #REQUIRED
                place CDATA #IMPLIED>
<!ELEMENT seances (seance)+>
<!ELEMENT seance (#PCDATA)>
<!ELEMENT titre (#PCDATA)>
<!ELEMENT ville (#PCDATA)>
<!ELEMENT departement (#PCDATA)>
<!ATTLIST departement num CDATA #REQUIRED>
<!ELEMENT adresse (#PCDATA)>
```

Structure définissable et “validable” par un *schéma*

La même chose mais en xml-schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
  <xs:element name="cinemas">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" ref="cinema"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="cinema">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="nom"/>
        <xs:element ref="adresse"/>
        <xs:element ref="ville"/>
        <xs:element ref="departement"/>
        <xs:element maxOccurs="unbounded" ref="salle"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
```

Structure définissable et “validable” par un *schéma*

La même chose mais en xml-schema

```
<xs:element name="nom" type="xs:string"/>
<xs:element name="adresse" type="xs:string"/>
<xs:element name="ville" type="xs:string"/>
<xs:element name="departement">
  <xs:complexType mixed="true">
    <xs:attributeGroup ref="attlist.departement"/>
  </xs:complexType>
</xs:element>
<xs:attributeGroup name="attlist.departement">
  <xs:attribute name="num" use="required"/>
</xs:attributeGroup>
<xs:element name="salle">
  <xs:complexType>
    <xs:sequence maxOccurs="unbounded">
      <xs:element ref="titre"/>
      <xs:element ref="seances"/>
    </xs:sequence>
    <xs:attributeGroup ref="attlist.salle"/>
  </xs:complexType>
</xs:element>
```

Structure définissable et “validable” par un *schéma*

La même chose mais en xml-schema

```
    <xs:element name="titre" type="xs:string"/>
  <xs:attributeGroup name="attlist.salle">
    <xs:attribute name="num" use="required"/>
    <xs:attribute name="place"/>
  </xs:attributeGroup>
  <xs:element name="seances">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" ref="seance"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="seance" type="xs:string"/>
</xs:schema>
```

Structure définissable et “validable” par un *schéma*

Les schémas ont plusieurs avantages, comme par exemple :

- un éditeur se sert du schéma pour faciliter l'édition d'un doc. XML et/ou vérifier sa conformité pour rapport à son schéma
- Un programme XSLT se base sur le schéma pour l'origine et la destination de la transformation et/ou à la vérification des règles XSLT
- En combinaison avec une analyse validante, le processeur XSLT vérifie avant l'exécution de la transformation si le doc. à transformer est conforme

Exercice (suite)

Rédiger une DTD, puis une instance XML du document sur les eaux.

- les éléments suivants doivent être formalisés :
  - l'alphabet terminal
  - l'alphabet auxiliaire
  - la structure arborescente
    (définissant certaines règles de dérivation)