XML Schema

Dan VODISLAV

CY Cergy Paris Université

Licence Informatique L3

Plan

- XML Schema
  - DTD vs. XML Schema
  - Types simples
  - Types complexes
  - Contraintes d'intégrité

XML Schema

- Recommandation W3C pour le typage des documents XML
  - Séparation types – éléments
    - Définition séparée des types et des éléments
    - Plusieurs éléments peuvent avoir le même nom mais des types différents
  - Richesse de types
    - Types simples très variés, types complexes, types locaux et anonymes
    - Héritage de types: extension, restriction
  - Contraintes d'intégrité: unicité, clés, clés étrangères
- Autres différence avec les DTD: schéma exprimé en format XML

Exemple

- Type complexe CinemaType
  - Nom (string)
  - Adresse (AdresseType, complexe)
  - Séances (SeanceType, complexe)

DTD

```
<!ELEMENT Cinema (Nom, Adresse, (Seance)*)>
<!ELEMENT Nom (#PCDATA)>
<!ELEMENT Adresse (Ville, Rue, Numero)>
...
```

Exemple (suite)

XML Schema

```
<xsd:element name='Cinema' type='CinemaType'/>

<xsd:complexType name='CinemaType'>
  <xsd:sequence>
    <xsd:element name='Nom' type='xsd:string'/>
    <xsd:element name='Adresse' type='AdresseType'/>
    <xsd:element name='Seance' type='SeanceType'
                 minOccurs='0' maxOccurs='unbounded'/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name='AdresseType'>
  <xsd:sequence>
    <xsd:element name='ville' type='xsd:string'/>
    <xsd:element name='rue' type='xsd:string'/>
    <xsd:element name='numero' type='xsd:decimal'/>
  </xsd:sequence>
</xsd:complexType>
```


Types simples

- Type simple: ensemble de valeurs (pas d'élément)
  - DTD: un seul type simple (#PCDATA), 10 types d'attributs
  - XML Schema: 43 types simples
    - xsd:string, xsd:byte, ...
    - xsd:integer, xsd:long, xsd:float, xsd:double, ...
    - xsd:boolean
    - xsd:anyType
    - xsd:time, xsd:timeDuration, xsd:Date, xsd:year, xsd:month, ...
    - xsd:language, xsd:uriReference
    - xsd:ID, xsd:IDREF, xsd:NMTOKEN, ...

Restrictions de types simples

- Définition de nouveaux types simples par rajout de *restrictions*
- Restriction par domaine

```
<xsd:simpleType name='anneeFilm'>
  <xsd:restriction base='xsd:integer'>
    <xsd:minInclusive value='1900'/>
    <xsd:maxInclusive value='2002'/>
  </xsd:restriction>
</xsd:simpleType>
```

- Restriction par motif
  - Numéro de téléphone: +33-(0), suivi d'un chiffre, ensuite quatre fois tiret - deux chiffres

```
<xsd:simpleType name='numTel'>
  <xsd:restriction base='xsd:string'>
    <xsd:pattern value='+33-\(0\)\d(-\d{2}){4}'/>
  </xsd:restriction>
</xsd:simpleType>
```

- Restriction par énumération

```
<xsd:simpleType name='marques'>
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="Audi"/>
    <xsd:enumeration value="Golf"/>
    <xsd:enumeration value="BMW"/>
  </xsd:restriction>
</xsd:simpleType>
```


Déclaration d'éléments et d'attributs

- Éléments

```
<xsd:element name='…' type='…' contraintes ...>
<xsd:element ref='…' contraintes>
```

  - Contraintes: minOccurs, maxOccurs, fixed, ...
  - Ex:

```
<xsd:element   name='nom' type='xsd:string'
               minOccurs='0' maxOccurs='2'/>
```

- Attributs

```
<xsd:attribute name use type ...>
```

  - Placé à la fin de la définition d'un élément / type d'élément
  - Use: *required*, *optional*, …
  - Ex:

```
<xsd:attribute name='langue'
               type='xsd:language'
               use='optional'/>
```

Types complexes

- Type complexe: constitué d'autres éléments/valeurs
- Constructeurs de types
  - `xsd:sequence` : séquence ordonnée d'éléments (DTD: ',')
  - `xsd:all` : séquence non-ordonnée d'éléments
  - `xsd:choice` : choix d'éléments (DTD: '|')
  - `xsd:group` : regroupement d'éléments (DTD: '(…)')

Exemple: séquence

- DTD: **(titre, année)**
- XML Schema

```
<xsd:complexType name='FilmType'>
  <xsd:sequence>
    <xsd:element name='titre' type='xsd:string'/>
    <xsd:element name='année' type='xsd:year'/>
  </xsd:sequence>
</xsd:complexType>
```

Contenu mixte

- DTD: **(#PCDATA | cinéma | film)***
  - Contenu mixte: texte et éléments mélangés au même niveau
  - DTD: restrictions sur les contenus mixtes (seulement choix et *)
- XML Schema
  - Les restrictions disparaissent
  - Le texte n'est pas à déclarer comme un « élément »
  - On décrit seulement le contenu une fois que tout le texte a été éliminé

```
<xsd:complexType name='OfficielType' mixed='true'>
  <xsd:choice minOccurs='0' maxOccurs='unbounded'>
    <xsd:element name='cinema' type='CinemaType' />
    <xsd:element name='film' type='FilmType' />
  </xsd:choice>
</xsd:complexType>
```


Éléments simples avec attributs

- Les types simples n'ont pas d'attributs
  - Définition par extension d'un type complexe avec contenu simple
- Exemple

```
<titre langue='français'>Le Goût des Autres</titre>
```

XML Schema

```
<xsd:element name='titre' type='titreType'/>
<xsd:complexType name='titreType'>
  <xsd:simpleContent>
    <xsd:extension base='xsd:string'>
      <xsd:attribute name='langue' type='xsd:string'/>
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>
```

Éléments vides avec attributs

- Exemple : `<film titre='If' année='1976'/>`

XML Schema

```
<xsd:element name='film' type='filmType'/>
<xsd:complexType name='filmType'>
  <xsd:attribute name='titre' type='xsd:string'/>
  <xsd:attribute name='année' type='annéeFilm'/>
</xsd:complexType>
```


Styles de déclaration de schéma

- Trois styles principaux
  - Avec types complexes nommés et séparés (utilisé dans ce cours)
  - Avec types complexes anonymes et inclus dans les éléments
  - Avec référence à des éléments
- Exemple

```
<!ELEMENT Cinema (Nom, Adresse, (Seance)*)>
<!ELEMENT Nom (#PCDATA)>
<!ELEMENT Adresse (Ville, Rue, Numero)>
<!ELEMENT Ville (#PCDATA)>
<!ELEMENT Rue (#PCDATA)>
<!ELEMENT Numero (#PCDATA)>
<!ELEMENT Seance (#PCDATA)>
```

- Style avec types complexes nommés et séparés déjà présenté
  - Seule différence: `Seance` est défini comme une chaîne de caractères (remplacer SeanceType par xsd:string)

Style avec types complexes anonymes

```
<xsd:element name='Cinema'>
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name='Nom' type='xsd:string'/>
      <xsd:element name='Adresse'>
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name='ville' type='xsd:string'/>
            <xsd:element name='rue' type='xsd:string'/>
            <xsd:element name='numero' type='xsd:decimal'/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
      <xsd:element name='Seance' type='xsd:string'
                   minOccurs='0' maxOccurs='unbounded'/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```


Style avec référence aux éléments

```
<xsd:element name='Nom' type='xsd:string'/>
<xsd:element name='ville' type='xsd:string'/>
<xsd:element name='rue' type='xsd:string'/>
<xsd:element name='numero' type='xsd:decimal'/>
<xsd:element name='Seance' type='xsd:string'/>
<xsd:element name='Adresse'>
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref='ville'/>
      <xsd:element ref='rue'/>
      <xsd:element ref='numero'/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
<xsd:element name='Cinema'/>
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref='Nom'/>
      <xsd:element ref='Adresse'/>
      <xsd:element ref='Seance'
                   minOccurs='0' maxOccurs='unbounded'/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

Contraintes d'intégrité

- DTD: intégrité référentielle (ID/IDREF)
- XML Schema: valeurs uniques, clés et références
- Exemple

```
<exemple>
  <film film_id="f23"> Gladiator </film>
  <film film_id="f12"> Avatar </film>
  ...
  <salle nom="St André des Arts">
    <seance ref_film="f12"> 14:30 </seance>
    <seance ref_film="f23"> 17:00 </seance>
    ...
  </salle>
  <salle nom="Kinorama">
    <seance ref_film="f12"> 15:30 </seance>
    ...
  </salle>
</exemple>
```


Valeurs uniques

- Exemple: l'attribut `film_id` d'un élément `film` doit être unique
  - Déclaration à l'intérieur de l'élément E du schéma qui contient les valeurs contraintes (ici E = `exemple`)

```
<xsd:unique name="idfilm">
  <xsd:selector xpath='film' />
  <xsd:field xpath='@film_id' />
</xsd:unique>
```

  - `selector:` chemin relatif par rapport à `exemple`
  - `field:` chemin relatif par rapport au `selector`
- Signification: pour tout sous-élément de E sélectionné par `selector`, la valeur de `field` doit être unique
  - Il peut y avoir plusieurs `field` pour un `selector`

Clés et références

- Clé d'un élément: similaire à la déclaration d'unicité
  - Nommée, la valeur doit toujours exister et ne peut pas être *nil*

```
<xsd:key name='filmclé'>
  <xsd:selector xpath='film' />
  <xsd:field xpath='@film_id' />
</xsd:key>
```

- Référence: clé étrangère, fait référence à une clé définie
  - Doit être placé de façon à ce que la déclaration de clé soit visible (dans un élément du schéma à l'intérieur de celui qui déclare la clé)
    - L'inclusion des éléments de schéma dépend du style
    - Le plus simple: la déclarer dans le même élément que la clé (`exemple`)
  - Dans l'exemple: l'attribut `ref_film` de `seance` est une clé étrangère

```
<xsd:keyref name='filmref' refer='filmclé'>
  <xsd:selector xpath='salle/seance' />
  <xsd:field xpath='@ref_film' />
</xsd:keyref>
```


Utilisation d'un XML Schema

- Fichier *cinema.xsd* contenant le schéma

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="Cinema" type="CinemaType"/>
  ...
</xsd:schema>
```

- Fichier *cinema.xml* basé sur le schéma *cinema.xsd*

```
<?xml version="1.0"?>
<Cinema xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="cinema.xsd">
   ...
</Cinema>
```