1. Introduction

Un document XML doit d'abord être bien formé, c'est à dire respecter la syntaxe XML décrivant l'écriture et l'imbrication des balises. Cependant, même en étant bien formé, un document peut avoir un contenu sémantique incorrect : des balises inconnues, manquantes ou avec le mauvais contenu pour représenter les informations. Les DTD, les XMLSchemas et les feuilles RelaxNG permettent de spécifier ce que doit contenir un document XML : balises, structuration, attributs et types. Les DTD sont simples, valables pour une première validation. Les schémas sont plus complets, avec une notion de types de données très forte, mais ils sont peu lisibles. RelaxNG présenté dans le CM n°3 est à la fois simple et complet.

Commencez par créer un dossier `tp2` pour ce TP et travaillez dedans.

2. Document Type Definitions

2.1. Processus de validation

Téléchargez deux fichiers : `ordinateurs_dtd.xml` et `ordinateurs.dtd` (clic droit sur ces liens, enregistrer sous...).

- Vérifier que `ordinateurs.xml` est valide, en ligne de commande par :
  - `xmlstarlet val --err --dtd ordinateurs.dtd -e ordinateurs_dtd.xml`
  - `xmllint --dtdvalid ordinateurs.dtd --noout ordinateurs_dtd.xml`
- Ouvrir les deux documents dans XML Copy Editor. Sélectionner l'onglet de `ordinateurs_dtd.xml` et appuyez sur la touche `F5` ou le menu `XML`, `Valider`, `DTD/XML Schema`. Le message en bas signale que le document est valide. La DTD a été mentionnée à l'intérieur du document par l'instruction `<!DOCTYPE` *racine* `SYSTEM "`*fichier.dtd*`">`.
- Regardez comment la feuille DTD `ordinateurs.dtd` est spécifiée dans `ordinateurs_dtd.xml`.

Voici maintenant des manipulations où on va altérer le document `ordinateurs_dtd.xml`. Le but est de vérifier s'il est encore bien formé par `F2` (*« ordinateur.xml est bien formé »*) puis s'il est toujours valide avec `F5` (*« ordinateurs.xml est valide »*). Après chaque manip, vous remettrez tout comme c'était avant en appuyant sur `CTRL Z` autant de fois que nécessaire.

- Rajouter un attribut `prix="649.90"` dans l'un des éléments `ordinateur`.
- Rajouter un sous-élément `<modèle>ultra mega speed</modèle>` dans l'un des éléments `ordinateur`.
- Changer le nom de l'élément `description` en `discription`.
- Enlever l'attribut `marque` de l'élément `description`.
- Enlever l'un des éléments `cpu`.
- Ajouter un second élément `cpu` dans l'un des ordinateurs.
- Permuter l'élément `cpu` avec `description`.
- Rajouter du texte sous un élément `cpu`. Pour cela, il faut enlever le `/` avant son `>`, taper du texte puis mettre la balise fermante `</cpu>`.
- Mettez la même valeur pour l'attribut `nom` des deux éléments `ordinateur`.

Avez-vous vu tous les messages d'erreurs possibles ? En résumé, c'est « chose en trop » ou « chose manquante », sauf le dernier changement qui est un peu plus subtil. Voici par contre, deux changements, qu'on peut considérer comme des erreurs à notre niveau et qui ne seront pas

détectées avec cette DTD.

- Changer la marque de cpu, mettre `nimportekoi` ou la chaîne vide à la place de `intel`.
- Vider le texte de l'une des descriptions mais en laissant les balises.

Constatez que les DTD sont trop simples, ne vérifient pas assez de choses.

2.2. Amélioration de la DTD

Maintenant, on va compléter la DTD et le document pour représenter de nouvelles informations. Ouvrir le cours de la semaine 2 dans le navigateur afin d'avoir la syntaxe.

Pour chacune des modifications proposées, il faut éditer à la fois le document et sa DTD.

- Rajouter un nouvel attribut optionnel appelé `prix` pour l'élément `ordinateur`. Sa valeur est un nombre avec deux chiffres après le point, par exemple : `"349.90"`. Cependant, une DTD ne permet pas de tester si la valeur est correcte.
- Rajouter un nouvel élément appelé `ram` qui est un enfant obligatoire de `ordinateur`. Cet élément doit être vide. Il doit avoir un attribut `taille` indiquant la quantité de mémoire en Go, ex: `<ram taille="6Go"/>`.
- Rajouter un nouvel élément appelé `disque`, enfant présent en au moins un exemplaire dans l'élément `ordinateur` (il peut y avoir plusieurs disques dans le même ordinateur). L'élément `disque` doit être vide et avoir un attribut `taille`, comme l'élément `ram`.
- Rajouter un attribut optionnel `date` dans l'élément `disque`. C'est la date d'achat du disque[1] qui sera au format `AAAA-MM-JJ` mais croyez-vous qu'il soit possible de le vérifier avec une DTD ?
- Rajouter un attribut `type` dans l'élément `disque`. La valeur de cet attribut devra être parmi `SSD`, `HDD` ou `HD` (il faut interdire les autres valeurs) et il peut être optionnel, par défaut c'est `HD` (attention à bien écrire les valeurs de cette manière en majuscules).
- Faire en sorte que la marque du `cpu` ne puisse être que `intel` ou `amd` (écrits ainsi en minuscules).

2.3. Écrire une DTD pour un document existant

Télécharger le fichier `chevaux.xml`. Votre travail consiste à écrire sa DTD, appelée `chevaux.dtd`, qui soit la plus précise possible. Par exemple, si un attribut semble n'avoir que deux valeurs, alors ça doit apparaître dans la DTD. Si un élément semble unique dans son parent, alors ça doit être défini dans la DTD.

Des points seront retirés si la DTD ne s'appelle pas exactement `chevaux.dtd` ou si ça ne valide pas le document XML fourni, avec ses particularités.

3. Les schémas XML

Les schémas sont plus précis que les DTD mais sont nettement plus complexes. Affichez cette partie du cours 2 dans un onglet du navigateur et relisez-la pour vous rappeler les concepts. Il

[1] Sachant que la durée de vie moyenne avant panne (MTBF) d'un disque dur est d'environ 4 ans selon l'usage, ça permet de savoir quand le changer.

faut définir des types pour les valeurs d'attributs et les contenus des éléments.

3.1. Processus de validation

Téléchargez deux fichiers : `ordinateurs_xsd.xml` et `ordinateurs.xsd` (clic droit sur ces liens, enregistrer sous...).

Le second est un schéma qui valide `ordinateurs_xsd.xml` ; ce dernier est lié au premier par l'attribut `xmlns:xsi` de sa racine. Les commandes à essayer sont :

- `xmlstarlet val --xsd ordinateurs.xsd -e ordinateurs_xsd.xml`
- `xmllint --schema ordinateurs.xsd --noout ordinateurs_xsd.xml`

Ouvrir les deux fichiers dans XML Copy Editor, puis appuyer sur `F5` dans `ordinateurs_xsd.xml`.

3.2. Modification du schéma

Le but de l'exercice est d'écrire le schéma correspondant à l'état final de votre document `ordinateurs_dtd.xml`, en partant du schéma actuel qui ne valide que quelques éléments. Commencez par recopier tout le contenu de `ordinateurs_dtd.xml` sauf la ligne de la DTD dans `ordinateurs_xsd.xml` et sans changer les attributs de la racine, puis modifiez le schéma.

Dans un premier temps, utilisez `xsd:string` pour tous les types. Quand vous aurez fini l'écriture du schéma, vous pourrez essayer de préciser les types. Relire le cours sur les types et les restrictions.

- Par exemple, restreindre le prix d'un ordinateur à un nombre à virgule, avec deux chiffres à droite (`xsd:decimal` avec une restriction `xsd:fractionDigits`).
- Question : est-il possible que la date d'achat d'un disque dur soit au format *JJ/MM/AAAA*. Dans ce cas, son type ne peut pas être `xsd:date` qui exige le format *AAAA-MM-JJ*, voir cette page. Donc vous devrez définir un type spécifique basé sur une restriction de type *pattern* sur la base `xsd:string`. Par contre, ce type ne pourra pas vérifier la validité des dates.

3.3. Schéma de `chevaux.xml`

Écrire une feuille XSD de validation du fichier `chevaux.xml`. En point de départ, vous pouvez vous servir de l'outil intégré à XML Copy Editor, voir `DTD→schema` dans le menu `XML` et l'appliquer à `chevaux.dtd`. **Attention** : il faut changer le namespace `xs:` en `xsd:`

Votre feuille de style devra être stricte concernant les valeurs des attributs et contenus des balises. Par exemple la taille sur 3 chiffres décimaux dont le premier est un 1, le poids est un entier entre 100 et 1500, la proportion un pourcentage correct, etc.

3.4. Schémas compacts (optionnel)

Un schéma peut être écrit en définissant des types comme jusqu'à présent, et c'est assez facile à lire surtout si c'est commenté. Voici un exemple :

```
<?xml version="1.0" encoding="utf-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
```

```
  <!-- racine du document : un élément <courrier> -->
  <xsd:element name="courrier" type="ElemCourrier"/>

  <!-- définition de l'élément <courrier> -->
  <xsd:complexType name="ElemCourrier">
    <xsd:sequence>
      <xsd:element name="message" type="ElemMessage" minOccurs="1" maxOccurs="unbounded"/
    </xsd:sequence>
  </xsd:complexType>

  <!-- définition de l'élément <message> -->
  <xsd:complexType name="ElemMessage">
    <xsd:simpleContent>
      <xsd:extension base="xsd:string">
        <xsd:attribute name="date" type="xsd:date" use="required"/>
        <xsd:attribute name="cc" type="xsd:string"/>
      </xsd:extension>
    </xsd:simpleContent>
  </xsd:complexType>

</xsd:schema>
```

Créer un document non trivial que cette feuille peut valider : `exemple.xml`.

Cela peut aussi s'écrire ainsi, en forme *compacte*, sans aucune séparation :

```
<?xml version="1.0" encoding="utf-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <!-- racine du document : un élément <courrier> -->
  <xsd:element name="courrier">
    <!-- contenu de l'élément <courrier> -->
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="message" minOccurs="1" maxOccurs="unbounded">
          <!-- contenu de l'élément <message> -->
          <xsd:complexType>
            <xsd:simpleContent>
              <xsd:extension base="xsd:string">
                <xsd:attribute name="date" type="xsd:date" use="required"/>
                <xsd:attribute name="cc" type="xsd:string"/>
              </xsd:extension>
            </xsd:simpleContent>
          </xsd:complexType>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
```

```
  </xsd:element>

</xsd:schema>
```

Vérifiez que cette feuille valide aussi votre `exemple.xml`.

4. Travail à rendre

Vous avez travaillé dans le dossier `tp2`. Remontez au dessus avec le navigateur de fichiers. Cliquez droit sur le dossier `tp2`, choisissez `Compresser...`, cliquez sur `Créer`. Ça va créer une archive `tp2.tar.gz`. Déposez cette archive sur Moodle, dans la page de cours dédiée L4IN121T Formats et traitements de données internet.