Chapitre III

Formalisation - Le "langage" *XML*

Dans ce chapitre, on considèrera l'exemple de la figure III.1 :

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Bagagerie -->
<bagagesPassagers
    xmlns="http://www.timc.fr/nicolas.glade/bagages"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.timc.fr/nicolas.glade/bagages bagages.xsd">
    <compagnie>Air France</compagnie>
    <date>2012-01-06</date>
    <bagage>
        <ref>AF677793</ref>
        <type format="sac à dos" couleur="rouge"/>
        <poids>9.8</poids>
    </bagage>
    <bagage>
        <ref>AF67840</ref>
        <type format="valise" couleur="noir"/>
        <poids>22.5</poids>
    </bagage>
</bagagesPassagers>
```

Figure III.1: Exemple de document XML (*instance xml*).

1 Éléments et attributs

XML est un langage à balises. On distingue les balises, les éléments et les attributs.

1.a Balise

Une balise est un marqueur. Elle commence par le signe inférieur < et se termine par le signe supérieur >. Exemples: Une balise peut être ouvrante <nom>, fermante </nom> ou ouvrante/fermante (auto-fermante) <nom/>.

En XML, **une balise ouverte doit TOUJOURS être refermée**.

1.b Élément

Un élément est un ensemble:

- balise ouvrante + contenu + balise fermante
  ou bien

- balise ouvrante/fermante (ou auto-fermante)

Exemples:
`<ref>AF677793</ref>` est un élément. Son contenu est `AF677793` et son nom `ref`.

Le contenu d'un élément peut aussi être un sous élément. Par exemple

```
<bagage>
        <ref>AF677793</ref>
        <type format="sac à dos" couleur="rouge"/>
</bagage>
```

est un élément dont le contenu est 3 sous-éléments (`<ref>...</ref>` , `<type.../>` et `<poids>...</poids>`).

L'élément `<type format="valise" couleur="noir"/>` est une balise ouvrante/fermante (ou auto-fermante) sans contenu. Un élement auto-fermant ne contient pas de texte, mais il peut cependant contenir des attributs ; c'est le cas ici.

1.c Attributs

Un attribut est un couple `nom = "valeur"`. Il est associé à une balise ouvrante ou une balise ouvrante/fermante. Il a un nom et une valeur. Par exemple dans l'exemple de la figure III.1, couleur="rouge" est un attribut de la balise ouvrante/fermante <type/>

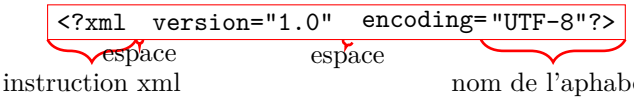
Vous remarquerez également que l'élément `bagagesPassagers` possède plusieurs attributs comme `xmlns` par exemple. Ces attributs, nous le verrons dans le chapitre suivant, servent à indiquer à quel vocabulaire appartiennent les différents mots (noms d'éléments) utilisés ici. Il s'agit ni plus ni moins de ce que l'on appelle **namespaces** (espaces de nom).

2 Les 8 points clés de XML

NB : ce qui est indiqué ici est valable pour TOUT document XML, à savoir les documents XML (instances), mais aussi les Schemas XML, les feuilles de transformation XSLT, ...

1. Prologue

Le prologue est la première ligne d'un document XML. Elle doit commencer au premier caractère du document (c'est à dire qu'il ne doit pas y avoir de caractères avant: ni caractères visibles, ni même de caractères d'échappement, espaces ...). Il indique au processeur du document (c'est-à-dire le logiciel qui va lire le document) qu'il s'agit d'un document XML. Il indique également la version de XML ainsi que l'alphabet (ASCII, UTF-8 ...) utilisé dans le document.



Il existe plusieurs alphabets numériques (encodages) que l'on peut utiliser dans un document XML. Si l'on utilise l'alphabet *ASCII* par exemple, on ne pourra pas utiliser d'accents ni de caractères spéciaux dans le document (nom des balises, mais également contenu des éléments). Ici, on utilisera majoritairement l'alphabet *UTF-8*, mais on pourrait également utiliser *Unicode*, latin-1, ISO-8859-1, etc.

NB : il n'est en général pas recommandé d'utiliser d'autres encodages que l'*ASCII* de base ! L'usage d'accents et de caractères spéciaux implique que votre système est bien configuré et surtout peut poser des problèmes de compatibilité lorsque les fichiers sont utilisés par d'autres personnes.

2. Arborescence et balises

Un document XML est une hiérarchie d'éléments représentable sous forme d'arbre. Il n'y a qu'un seul élément racine dans un document XML. Tout élément doit être soit l'élément racine, soit inclus dans un et un seul autre élément aussi appelé élément parent. Il ne doit donc pas y avoir de recouvrement.

L'exemple de la figure III.1 peut être représenté sous forme d'arbre de la façon suivante:

Un contre-exemple est donné dans la figure suivante:

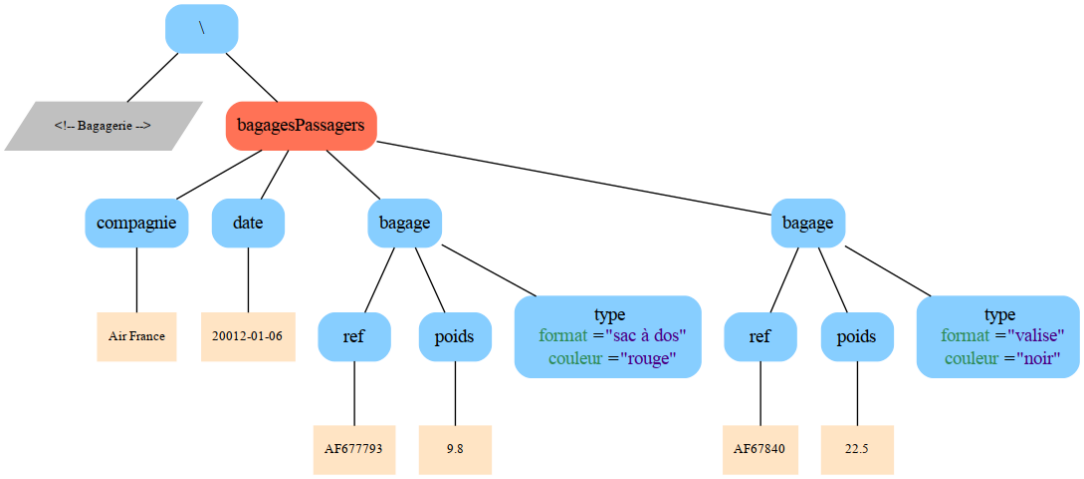
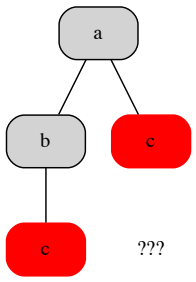


Figure III.2: représentation sous forme d'arbre du document XML de la figure II.7. .

```
1 <a>
2   <b>
3     <c>
4   </b>
5     </c>
6 </a>
```

(a) ce document



(b) ne peut pas être représenté comme un arbre

Figure III.3: Exemple de document ne vérifiant pas l'arborescence des balises: l'élément `c` n'est pas totalement inclus dans l'élément `b`.

3. Complétion

Tous les éléments d'un document XML doivent être complets. C'est-à-dire que:

- toute balise ouverte doit être fermée
- soit il y a un contenu (ex. `<balise>contenu</balise>`)
- soit l'élément est vide de contenu texte (ex. `<balise/>` équivalent à `<balise></balise>`)

4. Règles de nommage

L'alphabet est défini dans le prologue. L'encodage (`encoding`) indique les caractères qu'il est possible d'utiliser dans le document XML (donc y compris dans le nom des balises).

Par ailleurs, il existe des règles additionnelles pour définir le nom des balises et des attributs:

- tout nom d'élément commence par un caractère alphabétique non accentué (ou bien le caractère _ (souligné / underscore))
- les caractères qui suivent sont alphanumériques, accentués ou non (selon l'alphabet) ou le caractère _ (souligné / underscore) ou . (point / dot).
- il ne peut pas y avoir d'espace dans les noms de balise ou d'attribut

- le nom d'une balise ou d'un attribut ne peut pas commencer par `xml` (quelque soit la casse (minuscules ou majuscules))

Par ailleurs, XML est sensible à la *casse*[1]. La balise `<nomdefamille>` est donc différente de la balise `<nomDeFamille>`

Exemples:

- `<compteClient>` ⟶ **OK**
- `<ma balise>` ⟶ **KO** : *contient un espace dans le nom*
- `<maBalise nom="espace">` ⟶ **OK**
- `<Tom&Jerry>` ⟶ **KO** : *contient le caractère & qu'il faut remplacer comme nous le verrons dans le point 7*
- `<12j>` ⟶ **KO** : *commence par un caractère numérique et non alphabétique*
- `<base16>` ⟶ **OK**
- `<prénom>` ⟶ **OK** : *à condition que l'alphabet accepte les accents*

NB: Encore une fois, s'il est évidemment normal de pouvoir enregistrer n'importe quel symbole de n'importe quel encodage (sauf les caractères interdits qui doivent être remplacés par leurs substituts, *cf* section susbtitutions ci-dessous) dans les données contenues dans les éléments ou dans les valeurs des attributs, il n'est vraiment pas recommandé d'utiliser d'autre alphabet que l'alphabet ASCII pour les noms des éléments et des attributs.

5. Attributs

Les attributs sont associés aux **balises ouvrantes** ou **balises sans contenu**. Les noms des attributs suivent les même règles que les noms des éléments. La syntaxe des attributs est la suivante:

- nomAttribut="valeur"
- les guillemets sont obligatoires
- On peut remplacer " (double quote) par ' (single quote) si nécessaire
  (exemple: <type format='valise "à roulettes"'/>)

Sous-élément ou attribut ?

Il y a équivalence entre

- <type format='valise "à roulettes"'/> et
- <type><format>valise "à roulettes"</format></type>

Le choix entre l'utilisation d'attributs ou de sous-éléments est laissé au concepteur. Les critères de choix sont variés: la clarté de la présentation, la longueur du document, les conséquences sur la programmation, etc. Attention cependant, on ne peut pas avoir de sous-élément dans un attribut, alors que l'on peut *empiler* les hiérarchies dans un élément.

Remarque: en pratique une différenciation entre éléments et attributs se fait souvent sur l'utilisation associée aux contenus. Les éléments seront davantage utilisés pour stocker les données d'intérêt, tandis que les attribut seront utilisés pour caractériser ces données et éventuellement les retrouver lors d'une recherche spécifique dans le document. Dans l'exemple des bagages ci-dessous, la donnée stockée est la référence, les attributs de couleur et de format pouvant servir à retrouver des références correspondantes (associées à un passager). Ici, une recherche faite sur les sac à dos donnera une liste de 2 références.

[1] XML fait la différence entre les lettres minuscules et majuscules.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!-- Références bagagerie -->
<bagagesPassagers
        xmlns="http://www.timc.fr/nicolas.glade/bagages"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.timc.fr/nicolas.glade/bagages bagages.xsd">
    <compagnie>Air France</compagnie>
    <date>2012-01-06</date>
    <bagage>
        <ref>AF677793</ref>
        <type format="sac à dos" couleur="rouge"/>
        <poids>9.8</poids>
    </bagage>
    <bagage>
        <ref>AF67840</ref>
        <type format="valise" couleur="noir"/>
        <poids>22.5</poids>
    </bagage>
    <bagage>
                <ref>AF48717</ref>
                <type format="sac à dos" couleur="noir"/>
        <poids>7.3</poids>
    </bagage>
</bagagesPassagers>
```

6. Commentaires

Il est possible d'insérer des commentaires dans un document XML grâce à une balise spéciale:

```xml
        <!-- Ceci est un commentaire
                (possible sur plusieurs lignes)
        -->
```

Par exemple, la ligne 2 de l'exemple III.1 il n'y a pas d'élément, mais un commentaire.

On ne peut pas utiliser `--` à l'intérieur d'un commentaire. On ne peut pas insérer un commentaire à l'intérieur d'une balise (mais on peut évidemment insérer un commentaire à l'intérieur d'un élément).

7. Substitutions

Il existe des caractères interdits dans un document XML. par exemple, le caractère `<` indique le début d'une balise. On ne peut donc pas écrire

```xml
        <equation> x<y </equation>
```

dans un document XML.

On substitue alors les caractères interdits par un code commençant par &.

- substitutions obligatoires:
  - `<` doit être remplacé par < (*Lesser Than*)
  - `&` doit être remplacé par & (*AMPersand*, en français *esperluette*)
- substitutions conseillées:
  - `>` doit être remplacé par > (*Greater Than*)
  - `"` doit être remplacé par " (*QUOTation mark*)
  - `'` doit être remplacé par ' (*APOStroph*)

8. Rubriques CDATA

Dans une rubrique *Character Data*, les caractères ne sont pas analysés comme du XML par le processeur qui analyse le document. Ils peuvent en revanche être utilisés par d'autres applications (nous verrons des exemples dans le projet). Dans une telle section, tous les caractères sont autorisés, y compris <, & ... La rubrique CDATA est généralement utilisée pour inclure des portions de code non (nécessairement) XML (comme du Javascript, du C++, ... ou du code XML bien entendu).

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Du XML qui contient du C ... -->
<code
    xmlns="http://www.timc.fr/nicolas.glade/code"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.timc.fr/nicolas.glade/code code.xsd">
        <script>
                <![CDATA[
                        int comparer(int a, int b) {
                                if ((a < b) && (a < 0)) {
                                        return b-a;
                                } else {
                                        return a-b;
                                }
                        }
        ]]>
        </script>
</code>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Du XML qui contient du XML ... -->
<code
    xmlns="http://www.timc.fr/nicolas.glade/code"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.timc.fr/nicolas.glade/code code.xsd">
        <script>
                <![CDATA[
                        <?xml version="1.0" encoding="UTF-8"?>
                        <!-- Références bagagerie -->
                        <bagagesPassagers
                                xmlns="http://www.timc.fr/nicolas.glade/bagages
                                    "
                                xmlns:xsi="http://www.w3.org/2001/XMLSchema-
                                    instance"
                                xsi:schemaLocation="http://www.timc.fr/nicolas.
                                    glade/bagages bagages.xsd">
                                <compagnie>Air France</compagnie>
                                <date>2012-01-06</date>
                                <bagage>
                                        <ref>AF677793</ref>
                                        <type format="sac à dos" couleur="rouge
                                            "/>
                                        <poids>9.8</poids>
                                </bagage>
                                <bagage>
                                        <ref>AF67840</ref>
                                        <type format="valise" couleur="noir"/>
                                        <poids>22.5</poids>
                                </bagage>
                        </bagagesPassagers>
        ]]>
        </script>
</code>
```

3 Document bien formé

Un document XML est dit **bien formé** (*well formed*) si et seulement s'il respecte les règles de nommage, il n'a qu'un seul élément racine, il respecte les règles de complétude et de non-recouvrement.

Pour résumer, un document est bien formé, s'il respecte les 8 points clé du XML.