L'Altruiste : Le guide des langages Web

Le langage XLL

1 / Introduction

Le XLL (eXtended Linking Language) est une nouvelle fois un dérivé du langage XML.

Il apporte à ce-dernier **un support très puissant pour la gestion des liens** dans un document XML.

Le langage étendu des liens se divisent en deux parties complémentaires :

- **Les XLinks** (Liens XML)
- **Les XPointers** (Pointeurs XML)

Les XLinks permettent d'accéder à un document par l'intermédiaire d'un URI (Uniform Resource Identifier) à l'image de la balise *<a href="URL">* du langage HTML.

Les Xpointers ajoutent des fonctionnalités aux liens étendus par l'intermédiaire d'une expression qui apporte plus de précisions sur la cible à atteindre.

Ainsi, **le XLL avec ses deux composantes offre une gestion plus efficace des liens** vers des ressources externes.

Dans HTML, la gestion des liens est limitée :

- il n'est possible d'accèder qu'à **un unique document** à la fois,
- l'accès à une section particulière d'un document n'est opérable que si, auparavant, **l'ancrage *...* a été soigneusement opérée** lors de la conception de ce document,
- **aucune prise en charge de l'historique des liens** n'est pris en compte par le langage HTML, pour cela il est nécessaire de d'utiliser les langages de script,
- les liens et ancres sont déterminés par avance et ne laissent donc plus aucune possbilité d'extension dynamique.

Ne se contentant pas de reprendre en charge les fonctions des liens HTML, **le langage XLL propose de les améliorer et de les étendre** à un niveau d'efficacité remarquable.

- **N'importe quel élément XML peut devenir un lien.**
- Tous les types de ressources peuvent être non seulement accédés, mais également **pointés en des points précis**.
- Les **liens multidirectionnels** sont réalisables.
- Les liens peuvent être organisés en **groupe de connexions**.
- La gestion des **arcs de liens** devient effective.

Le langage XLL faisant encore aujourd'hui l'objet d'études par le W3C, les spécifications évoluent et ne sont pas encore parfaitement opérationnelles au sein de toutes les applications XML. Cependant, cet outil prometteur devrait à terme constituer l'ossature de la navigation sur le Web.

2 / Les X-Liens

Le langage XLink fournit un support très efficace pour la création de liens unidirectionnels (liens simples), à l'instar de ceux du HTML, mais également de liens bien plus complexes (liens étendus) permettant d'atteindre n'importe quel type de ressources dans un document XML.

Subséquemment, **les documents XML peuvent prendre en charge des relations entre plusieurs ressources**, soit des liaisons multidirectionnelles.

Ils **associent aux liens, des méta-données**, en l'occurrence des données informatives ou identifiantes explicitant d'autres données, justement ciblées par ces liens.

Enfin, ils peuvent **utiliser des liens qui résident dans un emplacement différent et séparé des ressources pointées**.

En premier lieu, **tous les éléments XLink doivent posséder un attribut d'espace de noms *xmlns*** indiquant un URI (Uniform Resource Identifier) identifiant le suffixe *xlink:*.

```
<xlink:element_XLL xmlns="http://www.w3.org/1999/xlink">
  ...
</xlink:element_XLL>
```

Eventuellement, **cet espace de noms peut être spécifié une seule fois, à condition qu'il soit localisé dans l'élément racine** du document.

Les liens simples peuvent être mis en oeuvre selon deux méthodes distinctes.

Tous les éléments peuvent devenir des liens, soit en leurs incluant un attribut *xlink:type*, soit en les encadrant par un élément *<xlink:simple>*.

Les éléments contenant l'attribut *xlink:type* sont appelés des éléments liants, car la valeur de cet attribut définit un type de liens, et partant, permet à l'hôte d'adopter des possiblités de liaison vers d'autres ressources.

Le marqueur *<xlink:simple>*, un x-lien simplifié, agît comme une balise *<a>* du HTML.
L'élément XML êtant encadré par ce marqueur, devient dès lors, un lien d'où il est aussi possible, là aussi, d'accèder à n'importe quelle ressource.

D'aucuns préféreront utiliser la première méthode estimant que la seconde demeure moins pratique.

Les liens étendus autorisent des fonctionnalités plus riches permettant la création de liaisons multidirectionnelles, d'arcs de liens ou encore de groupes de liens.

Dans ce cas, tout **un jeu d'attributs et d'éléments spécifiques** doivent être employés pour la mise en oeuvre de tels liens.

2.1 / Les espaces de noms

Les espaces de noms (namespace) permettent aux éléments et attributs XLink de s'associer à un préfixe spécial *xlink:* les distinguant des autres composants d'un document XML et d'ailleurs marquant leur appartenance au langage des liens XML.

Cette appartenance est **symbolisé par l'intermédiaire d'un adresse URI** (Uniform Resource Identifier) de référence, en outre standardisée par le World Wide Web Consortium (W3C).

http://www.w3.org/1999/xlink

L'URI de l'espace de noms XLink est mentionné par le truchement de l'attribut *xmlns:xlink* dans soit :

- l'élément racine du document XML,
- l'élément contenant les composants XLinks,
- tous les éléments XLinks.

```
<element xmlns:xlink="http://www.w3.org/1999/xlink">
  ...
</element>
```

Enfin, **plusieurs espaces de noms peuvent cohabiter ensemble dans un document XML** sans que cela ne soit problématique.

```
<monns:image xmlns:monns="http://www.site.com/ns"
             xmlns:xlink="http://www.w3.org/1999/xlink"
             xlink:type="simple"
             xlink:show="embed"
             xlink:actuate="onLoad"
             xlink:href="image.gif"/>
```

2.2 / Les liens simples

Les liens simples sont semblables aux liaisons standards du HTML.

Deux possiblités permettent de mettre en oeuvre ce genre de liens.

Le premier moyen s'effectue par l'intermédiaire de **l'élément *<xlink:simple>* encadrant l'élément liant**.

```
<xlink:simple
        xmlns:xlink="http://www.w3.org/1999/xlink"
        href="fichier.xml"/>
  <element/>
</xlink:simple>
```

Le second moyen est accompli par l'utilisation de **l'attribut *xlink:type="simple"* dans l'élément liant** lui-même.

```
<element
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xlink:type="simple"
        xlink:href="fichier.xml"/>
```

Dans ce cas, **les attributs XLinks doivent être impérativement accompagnés du préfixe *xlink:***.

D'autres attributs peuvent être employés dans ces éléments liants simples.

Attribut	Description
xlink:role	définit la nature des ressources jointes.
xlink:title	affecte un titre aux ressources jointes.
xlink:show	détermine le type d'affichage des ressources jointes.
xlink:actuate	définit le comportement du lien au moment de son activation.

```
<element
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xlink:role="document"
        xlink:title="Document comptable"
        xlink:show="replace"
        xlink:actuate="onRequest"
        xlink:type="simple"
        xlink:href="compte.xml"/>
```

Les composants des X-Liens simples doivent être, au préalable déclarés dans la Définition de Type de Document (DTD) au même titre que tous les éléments présents dans le document XML.

```
<!ELEMENT element ANY>
<!ATTLIST element
        xmlns:xlink CDATA #FIXED "http://www.w3.org/1999/xlink"
        xlink:type (simple) #FIXED "simple"
        xlink:href CDATA #IMPLIED
        xlink:role CDATA #IMPLIED
        xlink:title CDATA #IMPLIED
        xlink:show (new
                |replace
                |embed
                |other
                |none) #IMPLIED
        xlink:actuate (onLoad
                 |onRequest
                 |other
                 |none) #IMPLIED>
```

Cette déclaration est **valable pour les éléments liants comportant les attributs XLinks**.

Dans le cas de l'**utilisation d'un élément *<xlink:simple>***, la déclaration peut prendre la forme suivante :

```
<!ELEMENT xlink:simple ANY>
<!ATTLIST xlink:simple
        xmlns:xlink CDATA #FIXED "http://www.w3.org/1999/xlink"
        href CDATA #IMPLIED
        role CDATA #IMPLIED
        title CDATA #IMPLIED
        show (new
            |replace
            |embed
            |other
            |none) #IMPLIED
        actuate (onLoad
            |onRequest
            |other
            |none) #IMPLIED>
```

Les liens simples ne permettent donc que de joindre une unique ressource distante et partant, demeure très limités.

2.3 / Les liens étendus

Les liens étendus donnent la pleine mesure de XLINK puisqu'ils apportent des fonctionnalités supplémentaires à l'image des liens multidirectionnels ou du comportement des ressources entre elles par rapport à des liaisons.

La structure des liens étendus semble très complexe en comparaison à celle des liens HTML ou des liens simples.

En effet, **les liens sont multidirectionnels**, c'est-à-dire qu'ils ont la faculté de pouvoir joindre simultanément, un nombre quelconque de ressources.

Par exemple, un lien étendu pourrait construire un document résultant à partir de plusieurs portions de documents situés à des emplacements différents.

A cet effet, **un lien étendu est composé d'une ressources et de localisateurs**.

L'élément du type ressource

<xlink:resource> **est une ressource locale qui représente la source du lien.**

Lorsqu'une ressouce est spécifié, alors le lien étendu sera qualifié de *inline* (en ligne) puisque le lien provient de la ressource énoncée.

Si aucune ressource locale n'est indiquée, alors le lien étendu sera qualifié de *out of line* (hors ligne) car le lien ne fait parti d'aucun document source.

Les éléments de type localisateur

<xlink:locator> **sont des ressources distantes qui représentent les cibles du lien.**

Les arcs de liens

<xlink:arc> **permettent de définir le comportement des liaisons entre les ressources participant au lien étendu.**

Les localisateurs peuvent être aussi bien des cibles que des sources selon les caractéristiques énoncées par les arcs et la qualification du lien étendu qui les contient.

Les liens étendus ne comportant pas de ressources locales peuvent être stockés dans une base de liens. De cette façon, il devient possible de regrouper des liens étendus dans un fichier différent de ceux des documents XML auxquels ils font références.

Dans l'exemple ci-dessous, un lien étendu contient une ressource locale et quatre localisateurs. Tous sont repérés par un label d'identification.
L'arc de lien spécifie quatre traversées, toutes allant de la ressource locale vers chacune des ressources cibles.
Le dernier arc de traversée affichera le document dans une nouvelle fenêtre, les autres se contentant d'un affichage dans la fenêtre d'origine.

```
<menu xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="extended">
 <titre xlink:href="http://www.site.com/titre.xml"
        xlink:type="locator"
        xlink:title="Titre du livre en cours"
        xlink:label="tit">
  Titre
 </titre>
 <auteur xlink:href="http://www.site.com/auteur.xml"
         xlink:type="locator"
         xlink:title="Auteur du livre en cours"
         xlink:label="aut">
  Auteur
 </auteur>
 <editeur xlink:href="http://www.site.com/editeur.xml"
          xlink:type="locator"
          xlink:title="Editeur du livre en cours"
          xlink:label="edi">
  Editeur
 </editeur>
 <livre xlink:href="http://www.site.com/livre.xml"
        xlink:type="locator"
        xlink:title="Informations sur le livre en cours"
        xlink:label="liv">
  Livre
 </livre>
 <info xlink:type="resource"
       xlink:title="Interroger la base de la bibliothèque"
       xlink:label="inf">
  <details>Informations</details>
 </info>
 <xlink:arc from="inf" to="tit" show="replace" actuate="onRequest"/>
 <xlink:arc from="inf" to="aut" show="replace" actuate="onRequest"/>
 <xlink:arc from="inf" to="edi" show="replace" actuate="onRequest"/>
 <xlink:arc from="inf" to="liv" show="new" actuate="onRequest"/>
</menu>
```

Le second exemple ci-dessous, montre un lien étendu hors ligne (out of line) avec quatre ressources cibles et un arc de lien qui autorise toutes les traversées possibles entre n'importe quelles localisateurs énumérés.

```
<bibliotheque xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="extended">
 <titre xlink:href="http://www.site.com/titre.xml"
        xlink:type="locator"
        xlink:title="Titres des livres de la bibliothèque"
        xlink:label="tit">
  Titre
 </titre>
 <auteur xlink:href="http://www.site.com/auteur.xml"
         xlink:type="locator"
         xlink:title="Auteurs des livres de la bibliothèque"
         xlink:label="aut">
  Auteur
 </auteur>
 <editeur xlink:href="http://www.site.com/editeur.xml"
          xlink:type="locator"
          xlink:title="Editeurs des livres de la bibliothèque"
          xlink:label="edi">
  Editeur
 </editeur>
 <resume xlink:href="http://www.site.com/resume.xml"
         xlink:type="locator"
         xlink:title="Résumé des livres de la bibliothèque"
         xlink:label="res">
  Résumé
 </resume>
 <xlink:arc show="replace" actuate="onRequest"/>
</bibliotheque>
```

La déclaration des composants tels que les éléments et attributs XLinks de l'exemple ci-dessus, dans la Définition de Type de Document (DTD) peut se faire comme suit :

```
<!ELEMENT bibliotheque (titre, auteur, editeur, resume, xlink:arc)>
<!ATTLIST bibliotheque
            xmlns:xlink CDATA #FIXED "http://www.w3.org/1999/xlink"
            xlink:type (extended) #FIXED "extended">
<!ENTITY % attributs-xlink
            "xlink:href CDATA #REQUIRED
             xlink:label NMTOKEN #REQUIRED
             xlink:role CDATA #IMPLIED
             xlink:title CDATA #IMPLIED
             xlink:type CDATA (locator|resource) #REQUIRED 'locator'">
<!ELEMENT titre (#PCDATA)>
  <!ATTLIST titre %attributs-xlink;>
<!ELEMENT auteur (#PCDATA)>
  <!ATTLIST auteur %attributs-xlink;>
<!ELEMENT editeur (#PCDATA)>
  <!ATTLIST editeur %attributs-xlink;>
<!ELEMENT resume (#PCDATA)>
  <!ATTLIST resume %attributs-xlink;>
<!ELEMENT xlink:arc EMPTY>
  <!ATTLIST xlink:arc
            show (new|replace|embed|other|none) #REQUIRED 'replace'
            actuate (onLoad|onRequest|other|none) #REQUIRED 'onRequest'
            from NMTOKEN #IMPLIED
            to NMTOKEN #IMPLIED
            title CDATA #IMPLIED
  >
```

Dans cet extrait de DTD, on peut remarquer que **plusieurs attributs possèdent une valeur fixe ou par défaut**.
De cette façon, il est possible de de ne pas citer les attributs en question dans le document puisqu'ils sont automatiquement activés par défaut.
Cela permet de raccourcir considérablement la rédaction des liens dans un document XML.

```
<bibliotheque>
 <titre xlink:href="http://www.site.com/titre.xml"
        xlink:title="Titres des livres de la bibliothèque"
        xlink:label="tit">
  Titre
 </titre>
 <auteur xlink:href="http://www.site.com/auteur.xml"
         xlink:title="Auteurs des livres de la bibliothèque"
         xlink:label="aut">
  Auteur
 </auteur>
 <editeur xlink:href="http://www.site.com/editeur.xml"
          xlink:title="Editeurs des livres de la bibliothèque"
          xlink:label="edi">
  Editeur
 </editeur>
 <resume xlink:href="http://www.site.com/resume.xml"
         xlink:title="Résumé des livres de la bibliothèque"
         xlink:label="res">
  Résumé
 </resume>
 <xlink:arc/>
</bibliotheque>
```

L'interaction des sources, des cibles et des comportements de liaisons, respectivement représentés par les ressources, les localisateurs et les arcs de liens offrent au document XML un support de liens très puissant et efficace.

2.4 / Les bases de liens

Les base de liens (baselink) regroupent des liens étendus de type hors ligne (out of line), c'est-à-dire n'étant pas composés d'éléments de type ressource locale *<xlink:resource>*.

Par conséquent, **les ressources présentes dans ces liens étendus ne doivent être que du type localisateur *<xlink:locator>***.

De cette façon, **toutes les ressources incluses dans une base de liens ont la possibilité de devenir soit une source, soit une cible dans les arcs de traversée** définis par les éléments de type arc de liens *<xlink:arc>*.

Pour qu'une application XLink puisse entamer la traversée d'une ressource source vers une ressource cible, cela nécessite de localiser soit la ressource de départ, soit le lien.
Localiser ces deux derniers ne présente aucun problème dans le cas des arcs *outbound* (allant vers l'extérieur) car la ressource de départ est soit l'élément liant lui-même, soit un enfant de l'élément liant.
Toutefois, dans le cas d'arcs de type *inbound* (allant vers l'intérieur) et *third-party* (tiers d'arc), l'application XLink a besoin d'être capable de rechercher par n'importe quels moyens la ressource source ou la ressource cible.

Les bases de liens sont souvent utilisées dans le but de faciliter l'administration des liens par l'intermédiaire d'un regroupement d'éléments liants d'une même famille.
XLink fournit un chemin pour informer les applications XLinks afin d'accéder potentiellement aux bases de liens appropriées.
L'instruction prend la forme d'une spécification d'arc (si explicite dans un lien étendu, ou implicite dans un lien simple) qui a la valeur suivante pour son attribut *xlink:arcole*.

```
<xlink:arc
      from="source"
      to="cible"
      actuate="événement"
      arcole="http://www.w3.org/1999/xlink/properties/linkbase">
```

L'utilisation de cette URI (Uniform Resource Identifier) implique que la base de liens ciblée par ce genre d'arc, doit être impérativement un document de type XML.

Les applications XLink peuvent également utiliser n'importe quels autres moyens pour localiser et traiter les bases de liens supplémentaires.

La manipulation d'un arc d'une base de liens s'effectue comme celle d'un arc standard, excepté que la traversée entraîne le chargement de la ressource cible en l'occurrence, la base de liens, afin d'extraire ses liens pour une utilisation postérieure, plutôt que de la présenter à un utilisateur ou d'exécuter quelqu'autre processus.
Sa manipulation est aussi spéciale dans cette application XLink qui doit suspendre la traversée d'arcs de base de liens à une option utilisateur.

Spécifiquement, sur le chargement d'un arc de base de liens, une application XLink devrait conserver la trace de la ressource de départ. Chaque fois qu'un document contenant cette ressource de départ est chargée et la traversée de l'arc de la base de liens est activée, l'application devrait accéder à la base de liens et extraire n'importe lequel des liens étendus trouvé à l'intérieur.
Dans le cas, d'une portion d'un document XML complet extrait de la ressource, à l'image d'un intervalle de noeuds ou de chaîne de caractères, seulement ces liens étendus là entièrement contenu dans la portion extraite devrait être assuré d'être disponible.

La synchronisation de la traversée de l'arc de la base de liens dépend de la valeur de l'attribut *xlink:actuate* de l'arc. Par exemple, si la valeur est *onLoad*, la base de liens est chargée et ses liens sont extraits aussitôt que la ressource de départ est chargée.
N'importe quelle valeur d'attribut *xlink:show* d'un arc de base de liens doit être ignorée, car la traversée n'entraîne pas de présentation dans ce cas.

Les bases de liens peuvent être enchaînées à condition qu'elles puissent servir de ressources de départ pour d'autres arcs de bases de liens. L'application interprétant un arc de base de lien initial peut choisir de limiter le nombre d'étapes traitées dans la chaîne.

Une application devrait conserver une liste de liens étendus sauvée comme un résultat de

traîtement d'une base de liens, et devrait ne pas préserver les ressources identiques ou les liens dans le cas ou une dépendance cyclique existe. Dans le but de soulager le traitement de XLink, il est souhaitable que **la déclaration des arcs de base de liens se situe au début d'un document**.

Exemple :

```
<!-- Document appel_base.xml -->
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE appel_base [
 <!ELEMENT appel_base ((origine|base_lien|chargement)*)>
 <!ATTLIST appel_base
          xmlns:xlink CDATA #FIXED "http://www.w3.org/1999/xlink"
          xlink:type (extended) #FIXED "extended"
          xlink:title CDATA #IMPLIED>
 <!ELEMENT origine EMPTY>
 <!ATTLIST origine
          xlink:type (locator) #FIXED "locator"
          xlink:href CDATA #REQUIRED
          xlink:label NMTOKEN #IMPLIED>
 <!ELEMENT base_lien EMPTY>
 <!ATTLIST base_lien
          xlink:type (locator) #FIXED "locator"
          xlink:href CDATA #REQUIRED
          xlink:label NMTOKEN #IMPLIED>
 <!ELEMENT chargement EMPTY>
 <!ATTLIST chargement
          xlink:type (arc) #FIXED "arc"
          xlink:arcrole CDATA #FIXED
                    "http://www.w3.org/1999/xlink/properties/linkbase"
          xlink:actuate (onLoad|onRequest|other|none) #IMPLIED
          xlink:from NMTOKEN #IMPLIED
          xlink:to NMTOKEN #IMPLIED>
]>
<appel_base
      xlink:title="Chargement de la base de liens : lien_sprinter.xml">
 <origine xlink:label="doc" xlink:href="sprinter.xml"/>
 <base_lien xlink:label="base" xlink:href="lien_sprinter.xml"/>
 <chargement xlink:from="doc" xlink:to="base" actuate="onLoad"/>
</appel_base>
<!-- Base de liens lien_sprinter.xml -->
<?xml version="1.0" ISO-8859-1"?>
<!DOCTYPE xlink:extended SYSTEM "definition.dtd">
<xlink:extended>
 <xlink:locator label="ffred" href="#fredericks"/>
 <xlink:locator label="dbail" href="#bailey"/>
 ...
 <xlink:locator label="vit3" href="cent_metre.xml#fredericks"/>
 <xlink:locator label="vit9" href="cent_metre.xml#bailey"/>
 ...
 <xlink:arc from="ffred" to="vit3" actuate="onLoad" show="embed"/>
 <xlink:arc from="dbail" to="vit9" actuate="onLoad" show="embed"/>
 ...
</xlink:extended>
<!-- Document sprinter.xml -->
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE coureur [
 <!ELEMENT coureur (nom+)>
 <!ELEMENT nom ANY>
 <,!ATTLIST nom id ID #REQUIRED>
]>
<coureur>
 <nom id="fredericks">
 Franckie Fredericks
 </nom>
 <nom id="bailey">
 Donovan Bailey
 </nom>
 ...
</coureur>
```

Dans l'exemple ci-dessus, le chargement initial du document intermédiaire *appel_base.xml*, entraîne

l'activation de son lien étendu et partant de son arc de lien. Subséquemment, la ressource de départ *sprinter.xml* est alors associée à la ressource de fin, soit la base de liens *lien_sprinter.xml*. Le document *sprinter.xml* est alors parcouru en réinterprétant ses données conformément à la base de liens. Cette-dernière, finalement, active ses arcs de liens et affiche dans le document *sprinter.xml*, à la place des éléments *<nom id="...">* (ressources de départ), les ressources finales provenant du document *cent_metre.xml*.

```
<appel_base
     xlink:title="Détermination des liens dans le document courant">
  <origine
       xlink:label="doc"
       xlink:href="sprinter.xml#string-range(//*,'Consulter sa fiche.')"/>
  <base_lien xlink:label="base" xlink:href="lien_sprinter.xml"/>
  <chargement xlink:from="doc" xlink:to="base" actuate="onRequest"/>
</appel_base>
<!-- Document sprinter.xml -->
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE coureur [
  <!ELEMENT coureur (nom+)>
  <!ELEMENT nom ANY>
  <,!ATTLIST nom id ID #REQUIRED>
]>
<coureur>
  <nom id="fredericks">
  Franckie Fredericks
  Consulter sa fiche.
  </nom>
  <nom id="bailey">
  Donovan Bailey
  Consulter sa fiche.
  </nom>
  ...
</coureur>
```

Dans cet exemple, l'activation d'un lien s'effectue sur requête de l'utilisateur. L'intervalle de chaîne de caractères *Consulter sa fiche.* est la ressource de départ de l'arc de liens initial et après interprétation fera office de lien dans le document *sprinter.xml* pour chaque coureur.

2.5 / Les éléments XLINK

Les éléments XLinks permettent de créer des liens plus ou moins complexes vers des ressources.

Elément	Attributs	Description
<xlink:arc>	xlink:type="arc" from="source" to="cible" show="affichage" actuate="événement"	crée un élément de type arc de lien.
<xlink:extended>	xmlns:xlink ="http://www.w3.org/1999/xlink" xlink:type="extended" xlink:role="texte" xlink:title="texte"	crée un lien étendu.
<xlink:locator>	xlink:type="locator" xlink:href=*"URI"* xlink:role="texte" xlink:title="texte" xlink:label="texte"	crée un élément de type localisateur (ressource distante).
<xlink:resource>	xlink:type="resource" xlink:role="texte" xlink:title="texte" xlink:label="texte"	crée un élément de type ressource (ressource locale).
<xlink:simple>	xmlns:xlink ="http://www.w3.org/1999/xlink" xlink:type="simple" xlink:href=*"URI"* xlink:role="texte" xlink:title="texte" xlink:label="texte" xlink:show="affichage" xlink:actuate="événement"	crée un élément liant simple à l'instar des liens HTML.
<xlink:title>		crée un élément de titrage des éléments X-Liens.

2.5.1 / L'élément *<arc>*

Les éléments *<xlink:arc>* fournissent des règles de traversée entre les ressources participantes d'un lien étendu.

La formulation d'un arc de liens s'applique selon deux méthodes distinctes, soit par l'**utilisation de l'élément XLink *<xlink:arc>***,

```
<xlink:arc xmlns:xlink="http://www.w3.org/1999/xlink"
           from="source"
           to="cible"
           show="valeur"
           actuate="valeur"/>
```

soit par l'**insertion de l'attribut XLink *xlink:type="arc"* dans l'élément récepteur**.

```
<elt_liaison xmlns:xlink="http://www.w3.org/1999/xlink"
             xlink:type="arc"
             from="source"
             to="cible"
             show="valeur"
             actuate="valeur"/>
```

Un lien étendu *<xlink:extended>* peut contenir plusieurs éléments de type arc de liens.

L'élément de type d'arc peut avoir n'importe quel contenu, mais se trouve la plupart des cas, en tant qu'élément vide dans les documents XML.

Il ne doit pas exister deux arcs de liens identiques dans un lien étendu.

Les attributs

Attribut	Description
xlink:type	spécifie le type de l'élément arc de lien, soit *arc*.
xlink:from	indique la source de la traversée.
xlink:to	indique la cible de la traversée.
xlink:title	affecte un titre à la traversée.
xlink:show	détermine le type d'affichage de la ressource cible.
xlink:actuate	définit le comportement du lien au moment de son activation.

Si aucune valeur n'est fournie dans les attributs *xlink:from* et *xlink:to*, alors toutes les liaisons seront possibles entre les différentes ressources spécifiées dans le lien étendu, car **la valeur par défaut est égale à *all*** (tous).

Déclaration dans la DTD

L'utilisation des éléments et attributs XLink nécessite une déclaration préalable dans la Définition de Type de Document (DTD).

Dans le cas de l'**utilisation des attributs XLinks directement à l'intérieur de l'élément de type arc de lien**, la déclaration prend cette forme :

```
<!ELEMENT element EMPTY>
<!ATTLIST element
  xlink:show (new|replace|embed|other|none) #REQUIRED 'replace'
  xlink:actuate (onLoad|onRequest|other|none) #REQUIRED 'onRequest'
  xlink:from NMTOKEN #IMPLIED
  xlink:to NMTOKEN #IMPLIED
  xlink:title CDATA #IMPLIED
>
```

Dans le cas de l'**utilisation de l'élément *<xlink:arc>* et de ses attributs**, la déclaration devient :

```
<!ELEMENT xlink:arc EMPTY>
<!ATTLIST xlink:arc
  show (new|replace|embed|other|none) #REQUIRED 'replace'
  actuate (onLoad|onRequest|other|none) #REQUIRED 'onRequest'
  from NMTOKEN #IMPLIED
  to NMTOKEN #IMPLIED
  title CDATA #IMPLIED
>
```

Exemple :

```
<annee_scolaire
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xlink:title="Année scolaire 2001-2002">
  <professeur
    xlink:type="locator"
    xlink:href="http://www.site.com/prof_01-02.xml"
    xlink:role="http://www.site.com/professeur"
    xlink:title="Professeur pour la rentrée 2001-2002"
    xlink:label="prof">
  </professeur>
  <eleve
    xlink:type="locator"
    xlink:href="http://www.site.com/eleve_01-02.xml"
    xlink:role="http://www.site.com/eleve"
    xlink:title="Liste des élèves pour l'année scolaire 2001-2002"
    xlink:label="elev">
  </eleve>
  <cours
    xlink:type="locator"
    xlink:href="http://www.site.com/cours_01-02.xml"
    xlink:role="http://www.site.com/cours"
    xlink:title="Liste des cours pour l'année scolaire 2001-2002"
    xlink:label="cour">
  </cours>
  <xlink:arc from="cour" to="prof"/>
  <xlink:arc from="cour" to="elev"/>
</annee_scolaire>
```

2.5.2 / L'élément *<extended>*

Un lien étendu est une liaison qui associe un nombre arbitraire de ressources.

La formulation d'un lien étendu s'applique selon deux méthodes distinctes, soit par l'**utilisation de l'élément XLink *<xlink:extended>***,

```
<xlink:extended xmlns:xlink="http://www.w3.org/1999/xlink">
  <element>
    ...
  </element>
</xlink:extended>
```

soit par l'**insertion de l'attribut XLink *xlink:type="extended"* dans l'élément liant**.

```
<element xmlns:xlink="http://www.w3.org/1999/xlink"
          xlink:type="extended">
  ...
</element>
```

Les ressources participantes peuvent être une combinaison de ressources distantes ou locales, soit des éléments distinguéspar l'attribut *xlink:type* dont la valeur est respectivement *resource* ou *locator*.

Lorsque le lien étendu ne regroupe que des ressources distantes, appelées localisateurs (locator), il est qualifié de hors ligne (out of line), et partant, possède la capacité de pouvoir être regroupé avec d'autres liens du même type dans un document séparé, dénommé la base de liens.

Les liens étendus possèdent également un ou plusieurs éléments *<xlink:arc>* indiquant les caractéristiques des liaisons entre les ressources.

Les attributs

Attribut	Description
xlink:type	indique le type du lien étendu : *extended*.
xmlns:xlink	affecte l'URI *http://www.w3.org/1999/xlink* à l'espace de noms *xlink:*.
xlink:role	définit la nature du lien étendu.
xlink:title	affecte un titre au lien étendu.

Déclaration dans la DTD

L'utilisation des éléments et attributs XLink nécessite une déclaration préalable dans la Définition de Type de Document (DTD).

Dans le cas de l'**utilisation des attributs XLinks directement à l'intérieur de l'élément liant**, la déclaration prend cette forme :

```
<!ELEMENT element ANY>
<!ATTLIST element
        xmlns:xlink CDATA #FIXED "http://www.w3.org/1999/xlink"
        xlink:type (extended) #FIXED "extended"
        xlink:role CDATA #IMPLIED
        xlink:title CDATA #IMPLIED
>
```

Dans le cas de l'**utilisation de l'élément *<xlink:extended>* et de ses attributs**, la déclaration devient :

```
<!ELEMENT xlink:extended ANY>
<!ATTLIST xlink:extended
        xmlns:xlink CDATA #FIXED "http://www.w3.org/1999/xlink"
        role CDATA #IMPLIED
        title CDATA #IMPLIED
>
```

Exemple :

```
<bibliotheque xmlns:xlink="http://www.w3.org/1999/xlink"
                    xlink:type="extended">
 <info xlink:type="resource"
           xlink:title="Informations sur le livre en cours"
           xlink:label="inf">
  Aperçu du Livre
 <titre xlink:href="http://www.site.com/titre.xml"
       xlink:type="locator"
       xlink:title="Titre du livre en cours"
       xlink:label="tit">
  Titre
 </titre>
 <resume xlink:href="http://www.site.com/resume.xml"
        xlink:type="locator"
        xlink:title="Résumé du livre en cours"
        xlink:label="res">
  Résumé
 </resume>
 <xlink:arc show="replace" actuate="onRequest"/>
</bibliotheque>
```

2.5.3 / L'élément *<locator>*

Les éléments *<xlink:locator>* représentent les coordonnées des ressources distantes participant au lien étendu *<xlink:extended>*.

La formulation d'un lien étendu s'applique selon deux méthodes distinctes, soit par l'**utilisation de l'élément XLink *<xlink:locator>***,

```
<xlink:locator xmlns:xlink="http://www.w3.org/1999/xlink">
  <elt_localisateur>...</elt_localisateur>
</xlink:locator>
```

soit par l'**insertion de l'attribut XLink *xlink:type="extended"* dans l'élément de type localisateur**.

```
<elt_localisateur xmlns:xlink="http://www.w3.org/1999/xlink"
                  xlink:type="locator">
   ...
</elt_localisateur>
```

L'élément *<xlink:locator>* présente l'adresse URI (Uniform Resource Identifier) de la ressource distante. C'est elle qui permettra au lien étendu de joindre la cible.

Par ailleurs, **la ressource distante est représentée par une valeur identificatrice donnée par l'attribut *xlink:label***.
Les attributs *xlink:from* et *xlink:to* des éléments de type arc de liens peuvent **faire référence aux éléments de type localisateur**, par l'intermédiaire de ces labels d'identification.

Lorsqu'un lien étendu ne regroupe que des ressources distantes, appelées localisateurs (locator), il est qualifié de hors ligne (out of line), et partant, possède la capacité de pouvoir être regroupé avec d'autres liens du même type dans un document séparé, dénommé la base de liens.

L'élément de type de localisateur peut avoir n'importe quel contenu.

Les attributs

Attribut	Description
xlink:type	spécifie le type de l'élément localisateur, soit *locator*.
xlink:label	affecte un label identificateur à la ressource distante.
xlink:role	définit la nature de la ressource distante.
xlink:title	affecte un titre à la ressource distante.
xlink:href	spécifie l'adresse URI de la ressource à joindre.

La déclaration dans la DTD

L'utilisation des éléments et attributs XLink nécessite une déclaration préalable dans la Définition de Type de Document (DTD).

Dans le cas de l'**utilisation des attributs XLinks directement à l'intérieur de l'élément localisateur**, la déclaration prend cette forme :

```
<!ELEMENT element ANY>
<!ATTLIST element
        xlink:type (locator) #FIXED "locator"
        xlink:href CDATA #REQUIRED
        xlink:role CDATA #IMPLIED
        xlink:title CDATA #IMPLIED
        xlink:label NMTOKEN #IMPLIED
>
```

Dans le cas de l'**utilisation de l'élément *<xlink:locator>* et de ses attributs**, la déclaration devient :

```
<!ELEMENT xlink:locator ANY>
<!ATTLIST xlink:locator
        href CDATA #REQUIRED
        label NMTOKEN #IMPLIED
        role CDATA #IMPLIED
        title CDATA #IMPLIED
>
```

Exemple :

```
<xlink:extended xmlns:xlink="http://www.w3.org/1999/xlink"
  <xlink:locator
        xlink:title="Rapport de stage"
        xlink:href="http://www.site.com/rapport.xml">
    Rapports de stage 1999
  </xlink:locator>
  <xlink:locator
        xlink:title="Stagiaires"
        xlink:href="http://www.site.com/stagiaire.xml">
    Stagiaires de l'année 1999
  </xlink:locator>
  <xlink:arc show="replace" actuate="onRequest"/>
 </xlink:extended>
```

2.5.4 / L'élément *<resource>*

Les éléments *<xlink:resource>* fournissent des ressources locales participant à un lien étendu.

La formulation d'un lien étendu s'applique selon deux méthodes distinctes, soit par l'**utilisation de l'élément XLink *<xlink:resource>***,

```
<xlink:resource xmlns:xlink="http://www.w3.org/1999/xlink">
  <elt_ressource>...</elt_ressource>
</xlink:resource>
```

soit par l'**insertion de l'attribut XLink *xlink:type="resource"* dans l'élément de type ressource locale**.

```
<elt_ressource xmlns:xlink="http://www.w3.org/1999/xlink"
                xlink:type="resource">
  ...
</elt_ressource>
```

Le contenu de l'élément de type ressource peut avoir n'importe quel contenu au sein de ses marqueurs, toutefois il peut être aussi vide.

```
<info xlink:type="resource"
      xlink:title="Interroger la base de la bibliothèque"
      xlink:label="inf">
  <details>Informations</details>
</info>
```

Le contenu peut être représenté par une adresse URI (Uniform Resource Identifier) fournie par l'intermédiaire de l'attribut *xlink:role*.

```
<info
    xlink:type="resource"
    xlink:title="Interroger la base de la bibliothèque"
    xlink:role="http://www.site.com/livre.xml#xpointer(/*[1]/*[3]*/[1])">
    xlink:label="inf"/>
```

Les attributs

Attribut	Description
xlink:type	spécifie le type de l'élément ressource, soit *resource*.
xlink:label	affecte un label identificateur à la ressource locale.
xlink:role	définit la nature de la ressource locale.
xlink:title	affecte un titre à la ressource locale.

La déclaration dans la DTD

L'utilisation des éléments et attributs XLink nécessite une déclaration préalable dans la Définition de Type de Document (DTD).

Dans le cas de l'**utilisation des attributs XLinks directement à l'intérieur de l'élément de type ressource**, la déclaration prend cette forme :

```
<!ELEMENT element ANY>
<!ATTLIST element
                xlink:type (resource) #FIXED "resource"
                xlink:role CDATA #IMPLIED
                xlink:title CDATA #IMPLIED
                xlink:label NMTOKEN #IMPLIED
>
```

Dans le cas de l'**utilisation de l'élément *<xlink:resource>* et de ses attributs**, la déclaration devient :

```
<!ELEMENT xlink:resource ANY>
```

```
<!ATTLIST xlink:resource
                label NMTOKEN #IMPLIED
                role CDATA #IMPLIED
                title CDATA #IMPLIED
>
```

Exemple :

```
<xlink:extended xmlns:xlink="http://www.w3.org/1999/xlink">
  <livre xlink:href="http://www.site.com/livre.xml"
         xlink:type="locator"
         xlink:title="Informations sur le livre en cours"
         xlink:label="liv">
   Livre
  </livre>
  <info xlink:type="resource"
        xlink:title="Interroger la base de la bibliothèque"
        xlink:label="inf">
   <details>Informations</details>
  </info>
  <xlink:arc from="inf" to="liv" show="replace" actuate="onRequest"/>
</xlink:extended>
```

2.5.5 / L'élément *<simple>*

Un lien simple est une liaison qui associe exactement deux ressources, une locale et une autre éloignée, avec un arc allant de l'ancien au dernier.

Les attributs

Attribut	Description
xlink:type	spécifie le type de l'élément localisateur, soit *locator*.
xlink:label	affecte un label identificateur à la ressource distante.
xlink:role	définit la nature de la ressource distante.
xlink:title	affecte un titre à la ressource distante.
xlink:href	spécifie l'adresse URI de la ressource à joindre.
xlink:show	détermine le type d'affichage de la ressource à joindre.
xlink:actuate	définit le comportement du lien au moment de son activation.

La déclaration dans la DTD

L'utilisation des éléments et attributs XLink nécessite une déclaration préalable dans la Définition de Type de Document (DTD).

Dans le cas de l'**utilisation des attributs XLinks directement à l'intérieur de l'élément localisateur**, la déclaration prend cette forme :

```
<!ELEMENT element ANY>
<!ATTLIST element
        xlink:type (simple) #FIXED "simple"
        xlink:href CDATA #REQUIRED
        xlink:role CDATA #IMPLIED
        xlink:title CDATA #IMPLIED
        xlink:label NMTOKEN #IMPLIED
        xlink:show (new|replace|embed|other|none) #IMPLIED 'replace'
        xlink:actuate (onLoad|onRequest|other|none) #IMPLIED 'onRequest'
>
```

Dans le cas de l'**utilisation de l'élément *<xlink:locator>* et de ses attributs**, la déclaration devient :

```
<!ELEMENT xlink:simple ANY>
<!ATTLIST xlink:simple
        xlink:href CDATA #REQUIRED
        xlink:role CDATA #IMPLIED
        xlink:title CDATA #IMPLIED
        xlink:label NMTOKEN #IMPLIED
        xlink:show (new|replace|embed|other|none) #IMPLIED 'replace'
        xlink:actuate (onLoad|onRequest|other|none) #IMPLIED 'onRequest'
>
```

Exemple :

```
<!-- Affiche une image à l'instar de IMG du HTML -->
<image
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xlink:type="simple"
        xlink:href="illustration.gif"
        xlink:show="embed"
        xlink:actuate="onLoad"/>
```

2.5.6 / L'élément *<title>*

Ce type d'élément fournit des informations textuelles lisibles à propos du lien.

La formulation d'un élément de titrage s'applique selon deux méthodes distinctes, soit par l'**utilisation de l'élément XLink *<xlink:title>***,

```
<xlink:extended xmlns:xlink="http://www.w3.org/1999/xlink">
  <element>
    ...
  </element>
</xlink:extended>
```

soit par l'**insertion de l'attribut XLink *xlink:title* dans l'élément récepteur**.

```
<element xmlns:xlink="http://www.w3.org/1999/xlink"
          xlink:type="extended">
  ...
</element>
```

Les éléments *<xlink:extended>*, *<xlink:locator>*, *<xlink:arc>* et *<xlink:simple>* peuvent **posséder aussi bien l'attribut *xlink:title* que l'élément *<xlink:title>* au sein de leurs marqueurs** d'ouverture et de fermeture.

De tels éléments sont utiles, afin de **disposer d'informations lisibles sur les liens** ou parfois afin d'**identifier les différentes ressources dans les liens étendus**.

Déclaration dans la DTD

L'utilisation des éléments et attributs XLink nécessite une déclaration préalable dans la Définition de Type de Document (DTD).

Dans le cas de l'**utilisation des attributs XLinks directement à l'intérieur de l'élément récepteur**, la déclaration prend cette forme :

```
<ELEMENT element ANY>
<aTTLIST element
          xlink:title CDATA #IMPLIED
          xmlns:xlink CDATA #FIXED "http://www.w3.org/1999/xlink"
```

Dans le cas de l'**utilisation de l'élément *<xlink:title>***, la déclaration devient :

```
<ELEMENT xlink:title (#PCDATA)>
<aTTLIST xlink:title
          xmlns:xlink CDATA #FIXED "http://www.w3.org/1999/xlink">
```

Exemple :

```
<info xmlns:xlink="http://www.w3.org/1999/xlink"
      xlink:type="simple"
      xlink:title="Informations relatives au budget 2001">
  Budget 2001
</info>
```

2.6 / Les attributs XLINK

Les attributs XLinks permettent de fournir des informations à propos du lien, telles que l'adresse URI (Uniform Resource identifier), le type et la signification de la ressource.

Elément	Valeurs	Description
xlink:actuate	onLoad onRequest other none	détermine le type d'activation d'un lien.
xlink:arcole	CDATA URI	donne la signification des ressources.
xlink:from	NMTOKEN	fait référence à l'élément ressource ou localisateur à l'origine de la traversée.
xlink:href	URI	spécifie l'adresse URI de la ressource distante.
xlink:label	CDATA NMTOKEN	affecte un label à un élément de type ressource ou localisateur.
xlink:role	CDATA	affecte un titre à un élément.
xlink:show	new replace embed other none	détermine le type d'affichage de la ressource cible.
xlink:title	CDATA	affecte un titre à un élément.
xlink:to	NMTOKEN	fait référence à l'élément localisateur à l'extrémité de la traversée.
xlink:type	arc extended locator resource simple	indique le type de l'élément.
xmlns:xlink	http://www.w3.org/1999/xlink	indique l'adresse URI de référence de l'espace de noms *xlink:*.

2.6.1 / L'attribut *actuate*

L'attribut *xlink:actuate* permet d'indiquer le type d'activation d'un lien.

Cet attribut peut être présent dans les éléments :

- <xlink:arc>
- <xlink:simple>

Les valeurs

Attribut	Description
onLoad	provoque l'affichage de la ressource cible lors du chargement du document à l'instar de la balise *<img>* du HTML.
onRequest	déclenche la traversée de la source vers la cible, seulement sur un événement intentionnel comme un clic de souris sur un lien par l'utilisateur.
other	entraîne la recherche d'un autre marqueur présent dans le lien afin de déterminer le comportement approprié.
none	définit qu'aucun marqueur n'est présent, afin d'aider l'application à déterminer le comportement approprié.

Exemple :

```
<xlink:extended xmlns:xlink="http://www.w3.org/1999/xlink"
  <xlink:locator
                xlink:title="Rapport de stage"
                xlink:href="http://www.site.com/rapport.xml">
    Rapports de stage 1999
  </xlink:locator>
  <xlink:locator
                xlink:title="Stagiaires"
                xlink:href="http://www.site.com/stagiaire.xml">
    Stagiaires de l'année 1999
  </xlink:locator>
  <xlink:arc show="replace" actuate="onRequest"/>
 </xlink:extended>
```

2.6.2 / L'attribut *arcole*

L'attribut *xlink:arcole* décrit la signification des ressources à l'intérieur d'un lien, par l'intermédiaire d'une référence URI (Uniform Resource Identifier).

Cet attribut peut être présent dans les éléments :

- <xlink:arc>
- <xlink:simple>

Les valeurs

Attribut	Description
URI	Une adresse URI de forme relative ou absolue.

Exemple :

```
<appel_base
      xlink:title="Chargement de la base de liens : lien_sprinter.xml">
 <origine
          xlink:type="locator"
          xlink:label="doc"
          xlink:href="sprinter.xml"/>
 <base_lien
          xlink:type="locator"
          xlink:label="base"
          xlink:href="lien_sprinter.xml"/>
 <chargement
          xlink:type="arc"
          xlink:arcole="http://www.w3.org/1999/xlink/properties/linkbase"
          xlink:from="doc"
          xlink:to="base"
          actuate="onLoad"/>
</appel_base>
```

Dans cet exemple, l'attribut *xlink:arcole* indique que l'arc de lien fait appel aux propriétés énoncées par l'URI afin de résoudre la traversée entre les ressources concernée.

2.6.3 / L'attribut *from*

L'attribut *xlink:from* fait référence à l'extrémité d'un arc de traversé, en l'occurrence une ressource cible, représentée par un élément de type localisateur à l'intérieur d'un lien étendu.

Cet attribut peut être présent dans l'élément suivant :

- <xlink:arc>

Les valeurs

Attribut	Description
NMTOKEN	correspond à un mot unique identifiant une resssource de type *<xlink:locator>*.

Exemple :

```
<annee_scolaire
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xlink:title="Année scolaire 2001-2002">
  <professeur
    xlink:type="locator"
    xlink:href="http://www.site.com/prof_01-02.xml"
    xlink:role="http://www.site.com/professeur"
    xlink:title="Professeur pour la rentrée 2001-2002"
    xlink:label="prof">
  </professeur>
  <eleve
    xlink:type="locator"
    xlink:href="http://www.site.com/eleve_01-02.xml"
    xlink:role="http://www.site.com/eleve"
    xlink:title="Liste des élèves pour l'année scolaire 2001-2002"
    xlink:label="elev">
  </eleve>
  <cours
    xlink:type="locator"
    xlink:href="http://www.site.com/cours_01-02.xml"
    xlink:role="http://www.site.com/cours"
    xlink:title="Liste des cours pour l'année scolaire 2001-2002"
    xlink:label="cour">
  </cours>
  <xlink:arc from="cour" to="prof"/>
  <xlink:arc from="cour" to="elev"/>
</annee_scolaire>
```

2.6.4 / L'attribut *href*

L'attribut *xlink:href* spécifie l'adresse URI (Uniform Resource Identifier) de la ressource cible à laquelle un lien fait référence.

Cet attribut peut être présent dans les éléments :

- <xlink:locator>
- <xlink:simple>

Les valeurs

Attribut	Description
URI	Une adresse URI de forme relative ou absolue. Elle peut également contenir un fragment de locatisation XPointer.

Exemple :

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE poesie [
  <!ELEMENT poesie (titre, texte, auteur)>
  <!ELEMENT titre (#PCDATA)>
  <!ELEMENT texte (#PCDATA)>
  <!ELEMENT auteur (#PCDATA)>
  <!ATTLIST auteur
          xmlns:xlink CDATA #FIXED "http://www.w3.org/1999/xlink"
          xlink:type (simple) #FIXED "simple"
          xlink:href CDATA #REQUIRED
          xlink:title CDATA #IMPLIED
          xlink:show (new) #FIXED "new"
          xlink:actuate (onRequest) #FIXED "onRequest">
]>
<poesie>
  <titre>Locution des pierrots</titre>
  <texte>
  Je ne suis qu'un viveur lunaire
  Qui fait des ronds dans le bassin
  Et cela, sans autre dessein
  Que de devenir légendaire.

  Retroussant d'un air de défin
  Mes manches de Mandarin pâle,
  J'arrondis ma bouche et - j'exhale
  Des conseils doux de Crucifix

  Ah! oui, devenir légendaire,
  Au seuil des siècles charlatans !
  Mais où sont les Lunes d'antan ?
  Et que Dieu n'est-il à refaire ?
  </texte>
  <auteur xlink:href="auteur.xml#jlaforg"
          xlink:title="Voir la fiche de Jules Laforgue">
  Jules Laforgue
  </auteur>
</poesie>
```

2.6.5 / L'attribut *label*

L'attribut *xlink:label* affecte à une ressource, un label d'identification utilisable par les attributs *from* et *to* des éléments de type arc de liens.

Cet attribut peut être présent dans les éléments :

- <xlink:locator>
- <xlink:resource>

Les valeurs

Attribut	Description
NMTOKEN	correspond à un mot unique identifiant la resssource.

Exemple :

```
<annee_scolaire
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xlink:title="Année scolaire 2001-2002">
  <professeur
    xlink:type="locator"
    xlink:href="http://www.site.com/prof_01-02.xml"
    xlink:role="http://www.site.com/professeur"
    xlink:title="Professeur pour la rentrée 2001-2002"
    xlink:label="prof">
  </professeur>
  <eleve
    xlink:type="locator"
    xlink:href="http://www.site.com/eleve_01-02.xml"
    xlink:role="http://www.site.com/eleve"
    xlink:title="Liste des élèves pour l'année scolaire 2001-2002"
    xlink:label="elev">
  </eleve>
  <cours
    xlink:type="locator"
    xlink:href="http://www.site.com/cours_01-02.xml"
    xlink:role="http://www.site.com/cours"
    xlink:title="Liste des cours pour l'année scolaire 2001-2002"
    xlink:label="cour">
  </cours>
  <xlink:arc from="cour" to="prof"/>
  <xlink:arc from="cour" to="elev"/>
</annee_scolaire>
```

2.6.6 / L'attribut *role*

L'attribut *xlink:role* décrit la signification des ressources à l'intérieur d'un lien, par l'intermédiaire d'une référence URI (Uniform Resource Identifier).

Cet attribut peut être présent dans les éléments :

- <xlink:extended>
- <xlink:locator>
- <xlink:resource>
- <xlink:simple>

Les valeurs

Attribut	Description
URI	Une adresse URI de forme relative ou absolue.

Exemple :

```
<livre xmlns:xlink="http://www.w3.org/1999/xlink">
  <titre
   xlink:type="locator"
   xlink:href="http://www.librairie.com/database/2724267737.xml"
   xlink:role="http://www.librairie.com/livre"
   xlink:title="La tragédie Cathare"/>
  <auteur
   xlink:type="locator"
   xlink:href="http://www.greece.antique/database/gbordonove.xml"
   xlink:role="http://www.librairie.com/auteur"
   xlink:title="Georges Bordonove"/>
  <editeur
   xlink:type="locator"
   xlink:href="http://www.librairie.com/database/pygmalion.xml"
   xlink:role="http://www.librairie.com/editeur"
   xlink:title="Editions Pygmalion"/>
</livre>
```

2.6.7 / L'attribut *show*

L'attribut *xlink:show* détermine le type d'affichage de la ressource cible.

Cet attribut peut être présent dans les éléments :

- <xlink:arc>
- <xlink:simple>

Les valeurs

Attribut	Description
new	provoque l'affichage de la ressource cible dans une nouvelle fenêtre à l'image de l'attribut *target="_blank"* du HTML.
replace	occasionne le remplacement de l'affichage d'origine du lien par celui de la ressource cible dans la même fenêtre ou cadre, à l'instar de l'attibut *target="_self"* du HTML.
embed	insère la ressource cible dans la fenêtre courante comme le ferait la balise *<img>* du HTML.
other	entraîne la recherche d'un autre marqueur présent dans le lien afin de déterminer le comportement approprié.
none	définit qu'aucun marqueur n'est présent, afin d'aider l'application à déterminer le comportement approprié.

Exemple :

```
<xlink:extended xmlns:xlink="http://www.w3.org/1999/xlink"
  <xlink:locator
                xlink:title="Rapport de stage"
                xlink:href="http://www.site.com/rapport.xml">
    Rapports de stage 1999
  </xlink:locator>
  <xlink:locator
                xlink:title="Stagiaires"
                xlink:href="http://www.site.com/stagiaire.xml">
    Stagiaires de l'année 1999
  </xlink:locator>
  <xlink:arc show="replace" actuate="onRequest"/>
 </xlink:extended>
```

2.6.8 / L'attribut *title*

L'attribut *xlink:title* affecte une information textuelle compréhensible à un élément XLINK.

Cet attribut peut être présent dans les éléments :

- <xlink:arc>
- <xlink:extended>
- <xlink:locator>
- <xlink:resource>
- <xlink:simple>

Les valeurs

Attribut	Description
CDATA	accepte toutes données textuelles analysées.

Exemple :

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE carte [
  <!ELEMENT carte (image,zone*,legende?)>
  <!ELEMENT image EMPTY>
  <!ATTLIST image src CDATA #REQUIRED>
  <!ELEMENT zone (localisateur+)>
  <!ATTLIST zone
            xmlns:xlink CDATA #FIXED "http://www.w3.org/1999/xlink"
            xlink:type (extended) #FIXED "extended"
            inline (true|false) "true"
            xlink:label ID #REQUIRED
            forme (rect|circle|polygon) #IMPLIED
            coords CDATA #REQUIRED
            xlink:title CDATA #IMPLIED>
  <!ELEMENT localisateur EMPTY>
  <!ATTLIST localisateur
            xmlns:xlink CDATA #FIXED "http://www.w3.org/1999/xlink"
            xlink:type (locator) #FIXED "locator"
            xlink:href CDATA #REQUIRED
            xlink:role CDATA #IMPLIED
            xlink:title CDATA #IMPLIED
            xlink:show (embed|replace|new) "replace"
            xlink:actuate (auto|user) "user">
  <!ELEMENT legende (#PCDATA)>
]>
<carte>
  <image src="../coursxll/lemonde.jpg"/>
  <zone xlink:label="europe"
       xlink:title="Le continent européenne"
       forme="circle"
       coords="405,105,70">
  <localisateur xlink:href="europe.xml"
            xlink:title="voir l'histoire de l'Europe"/>
  </zone>
  <zone xlink:label="oceanie"
       xlink:title="Le continent océanique"
       forme="circle"
       coords="680,380,120">
  <localisateur xlink:href="oceanie.xml"
            xlink:title="voir l'Histoire de l'Océanie"/>
  </zone>
  <zone xlink:label="afrique"
       xlink:title="Le continent africain"
       forme="rect"
       coords="325,175,505,430">
  <localisateur xlink:href="afrique.xml"
            xlink:title="voir l'Histoire de l'Afrique"/>
  </zone>
  <zone xlink:label="amerique"
       xlink:title="Le continent américain"
       forme="polygon"
       coords="250,0,120,40,55,110,55,110,120,180,230,
                520,280,500,320,310,250,200,300,100">
  <localisateur xlink:href="amerique.xml"
            xlink:title="voir l'Histoire de l'Amérique"/>
  </zone>
  <zone xlink:label="asie"
       xlink:title="Le continent américain"
       forme="polygon"
       coords="450,30,510,180,540,270,550,280,630,
                260,695,237,710,70,657,20,550,0">
  <localisateur xlink:href="asie.xml"
            xlink:title="voir l'Histoire de l'Asie"/>
  </zone>
  <legende>Histoire du Monde</legende>
</carte>
```

2.6.9 / L'attribut *to*

L'attribut *xlink:to* fait référence à l'origine d'un arc de traversé, en l'occurrence une ressource source, représentée par un élément de type ressource ou localisateur à l'intérieur d'un lien étendu.

Cet attribut peut être présent dans l'élément suivant :

- <xlink:arc>

Les valeurs

Attribut	Description
NMTOKEN	correspond à un mot unique identifiant les resssources d'un lien étendu.

Exemple :

```
<annee_scolaire
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xlink:title="Année scolaire 2001-2002">
  <professeur
    xlink:type="locator"
    xlink:href="http://www.site.com/prof_01-02.xml"
    xlink:role="http://www.site.com/professeur"
    xlink:title="Professeur pour la rentrée 2001-2002"
    xlink:label="prof">
  </professeur>
  <eleve
    xlink:type="locator"
    xlink:href="http://www.site.com/eleve_01-02.xml"
    xlink:role="http://www.site.com/eleve"
    xlink:title="Liste des élèves pour l'année scolaire 2001-2002"
    xlink:label="elev">
  </eleve>
  <cours
    xlink:type="locator"
    xlink:href="http://www.site.com/cours_01-02.xml"
    xlink:role="http://www.site.com/cours"
    xlink:title="Liste des cours pour l'année scolaire 2001-2002"
    xlink:label="cour">
  </cours>
  <xlink:arc from="cour" to="prof"/>
  <xlink:arc from="cour" to="elev"/>
</annee_scolaire>
```

2.6.10 / L'attribut *type*

L'attribut *xlink:type* définit le type de l'élément.

Cet attribut peut être présent dans les éléments de type :

- <xlink:arc>
- <xlink:extended>
- <xlink:locator>
- <xlink:resource>
- <xlink:simple>
- <xlink:title>

Les valeurs

Attribut	Description
arc	crée un élément de type arc de lien.
extended	crée un lien étendu.
locator	crée un élément localisateur.
resource	crée un élément de type ressource.
simple	crée un lien simple.
none	signifie que l'élément n'a pas de signification XLink.

Exemple :

```
<lien
      xlink:type="extended"
      xmlns:xlink="http://www.w3.org/1999/xlink">
 <livre xlink:href="http://www.site.com/livre.xml"
        xlink:type="locator"
        xlink:title="Fiche du livre en cours"
        xlink:label="liv"/>
 <info xlink:type="resource"
        xlink:title="Information sur le livre en cours"
        xlink:label="inf">
  <details>Informations</details>
 </info>
 <traversee
         xlink:type="arc"
         from="inf"
         to="liv"
         show="replace"
         actuate="onRequest"/>
</lien>
```

2.6.11 / Les X-Pointeurs

Les X-Pointeurs (XPointers), en conjonction avec les X-Liens, permettent de cibler plus précisément une ressource quelconque dans l'arborescence d'un document XML.

Le langage XPointer est destiné à être la base d'identificateurs de fragments seulement pour les documents ayant pour types de média : text/xml ou application/xml.

Les pointeurs étendus sont des expressions **utilisées dans les valeurs des attributs *xlink:href*** des éléments liants.

Les X-Pointeurs permettent d'**adresser des points précis, des intervalles ainsi que des noeuds complets** à l'intérieur de documents.

Egalement, ils permettent de **localiser des informations par l'itnermédiaire de chaînes de caractères de mises en correspondance**.

Enfin, **ils utilisent les expressions d'adressage dans des références d'URI** (Uniform Resource Identifier) comme fragment d'identificateur.

L'avantage indéniable de ces pointeurs est de pouvoir **cibler n'importe quel composant d'un document XML** sans que celui-ci soit obligé de subir une préparation en conséquence comme ce serait le cas dans le langage HTML.

En effet, **chacun des noeuds d'un document XML peut être pointé par un expression XPointer appropriée**.

En fait, **les pointeurs sont capables d'établir une connexion sur des noeuds, un point précis, mais aussi sur un intervalle d'un document XML**.

L'expression, ci-dessous, sélectionne l'élément dont son attribut *id* possédant la valeur *introduction*. Ainsi, un noeud précis du document est atteint.

```
xpointer(id("introduction"))
```

Ce deuxième pointeur cible un point au sein de données textuelles appartenant à un élément XML.

```
xpointer(start-point(string-range(//*, "introduction")))
```

Cette troisième expression sélectionne tous les éléments possédant un attribut *id* dont la valeur est comprise dans l'intervalle *de 1 à 10*. Ici, les dix premiers chapîtres sont récupérés.

```
xpointer(id("chapitre1")/range-to(id("chapitre10")))
```

3 / La syntaxe des X-Pointeurs

Un expression X-Pointeur obéit à des règles particulières de syntaxe.

Premièrement, **un X-Pointeur doit être formulé comme suit** :

```
xpointeur(expression)
```

Deuxièmement, **l'expression doit se trouver à la suite d'un URI** (Uniform Resource identifier) compris lui-même dans l'attribut *xlink:href* d'un élément liant.

```
<element xlink:type="type" xlink:href="uri#xpointer(expression)">
  ...
</element>
```

L'expression XPointer doit être séparé de l'URI par un caractère # (dièse) comme ci-dessus.

Un X-Pointeur peut également avoir **une forme abrégée appelé *child sequence***.

```
<element xlink:type="type" xlink:href="uri#/n/n/n...">
  ...
</element>
```

Une valeur d'attribut identificateur peut être également spécifié. Cette seconde forme abrégée est appelé *bare name*.

```
<element xlink:type="type" xlink:href="uri#valeur_id">
  ...
</element>
```

Il est possible d'utiliser à la fois les deux formes abrégées, *child sequence* et *bare name*.

```
<element xlink:type="type" xlink:href="uri#valeur_attribut/n/n...">
  ...
</element>
```

Les axes nodaux sont tous utilisables dans les expressions XPointers.

```
<element xlink:type="type" xlink:href="uri#epression/axes_nodaux">
  ...
</element>
```

Un X-Pointeur peut être composé de ses fonctions propres ainsi que de celles de XPath.

```
<element xlink:type="type" xlink:href="uri#epression/fonctions">
  ...
</element>
```

Les XPointers acceptent **les prédicats**.

```
<element xlink:type="type" xlink:href="uri#expression[prédicat]">
  ...
</element>
```

Exemple

```
<index xlink:type="locator"
        xlink:href="http://www.site.com/livre.xml#xpointer(id("chap5"))">
Chapître 5
</index>

<index xlink:type="locator"
        xlink:href="livre.xml#xpointer((id("chap9"))range-to(id("chap10")))">
Chapître 9 et 10
</index>

<index xlink:type="locator"
        xlink:href="livre.xml#crime_et_chatiment/6">
Chapître 6
</index>

<index xlink:type="locator"
         xlink:href="../livre.xml#/1/7">
Chapître 7
</index>

<index xlink:type="locator"
        xlink:href="livre.xml#xpointer(child::crime_et_chatiment[position(8)])">
Chapître 8
</index>
```

3.1 / Les caractères d'échappement

Les caractères d'échappement permettent d'insérer dans des expressions X-Pointers des caractères réservés ou diverses lettres, accentuées notamment. Cela permet d'éviter toutes ambiguités quant aux délimiteurs de l'expressions.

Une paranthèse ne peut être incluse en l'état. Il faut faire appel à un caractère d'échappement "^" (accent circonflexe).

```
xpointer(string-range("^(une valeur^)"))
```

D'ailleurs, **l'accent circonflexe doit être doublé** pour pouvoir l'introduire dans une expression.

```
xpointer(string-range("^^une valeur"))
```

Les caractères *&* et *<* doivent être remplacés par leur référence d'entité respective : & et <.

```
xpointer(//element position() &lt; 3)

xpointer(string-range("XML &amp;amp; Compagnie"))
```

Les caractères guillemets simples (') ou doubles (") doivent être remplacés par leur valeur hexadécimale ou leur référence d'entité.

```
<!-- Guillemets doubles " -->
xpointer(string-range(//*, 'un "essai fini"'))

<-- Valeur hexadécimale de " -->
xpointer(string-range(//*, 'un &#34;essai fini&#34;'))

<!-- Référence d'entité de " -->
xpointer(string-range(//*, 'un &quot;essai fini&quot;'))

<!-- Guillemets simples ' -->
xpointer(string-range(//*, 'un 'essai fini"))

<!-- Valeur hexadécimale de ' -->
xpointer(string-range(//*, 'un &#39;essai fini&#39;'))

<!-- Référence d'entité de ' -->
xpointer(string-range(//*, 'un &apos;essai fini&apos;'))
```

Les caractères accentués du jeu d'encodage ISO-8859 doivent être échangé par leur valeur hexadécimale ou leur référence d'entité.

```
<!-- Valeur hexadécimale de é -->
xpointer(id("Pr&#233;ambule"))

<!-- Référence d'entité de é -->
xpointer(id("Pr&eacute;ambule"))
```

3.2 / La formulation complète

La formulation complète des expressions peuvent combiner plusieurs parties spécifiques des XPointers.

Un X-Pointeur commence toujours par le mot *xpointer*, hormis pour les formes abrégées, et **renferme l'expression à proprement parler au sein de paranthèses**.

xpointer(expression)

xpointer(/descendant::element[6])

L'exemple ci-dessus permet d'atteindre le sixième élément qui descend directement ou indirectement du noeud racine.

Une combinaison de plusieurs X-Pointeurs se suivant, séparés par un espace blanc, permet **une évaluation de la gauche vers la droite en cas d'échec de la partie précédente** pour une quelconque raison.

- Le XPointer est inconnu.
- Le XPointer n'est pas applicable au type de média de la ressource.
- Le XPointer ne parvient pas à localiser une sous-ressource dans la ressource.
- La fonction *string-range* du XPointer, possèdant un argument chaîne de caractères, ne trouve pas la chaîne dans la valeur du noeud localisé.
- Le troisième et quatrième argument de la fonction *string-range* indique une chaîne qui se trouve au delà du commencement ou de la fin d'un document.
- Le noeud retourné par la fonction *start-point* est du type attribut ou espace de noms.

xpointer(expression) xpointer(expression)

xpointer(/elt_racine/element) xpointer(//element)

Ce pointeur sélectionne les éléments *<element>* présents sous l'élément racine *<elt_racine>*. Si la première expression échoue alors la seconde prend le relais et tente de trouver l'élément *<element>* dans la descendance du noeud racine.

3.2.1 / Les espaces de noms

Le langage XPointer accepte une instruction spéciale afin de spécifier l'espace de noms de l'élément à atteindre.

Si un pointeur tente de cibler un élément contenant un préfixe d'espace de noms, sans l'avoir auparavant déclarer, une erreur sera retournée.

C'est pourquoi, un XPointer peut combiner **une partie déclarant l'URI (Uniform Resource Identifier) de l'espace de noms cible *xmlns()*** et le pointeur *xpointer()* afin de lever toute ambiguité quant à l'espace de nom à accèder.

xmlns(prefixe=URI) xpointer(expression)

xmlns(art=http://www.art.com/xmlns/) xpointer(//art:element)

Comme pour les combinaisons de pointeurs, **il est possible de déclarer plusieurs espaces de noms** si l'expression du XPointer fait appel à plusieurs préfixes *namespace* différents.

<xmlns(pfx1=URI1) xmlns(pfx2=URI2) xpointer(//**pfx1:**elt/**pfx2:**elt)

Exemple

```
<!-- http://www.site.com/archive.xml -->
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE archive SYSTEM "../dtd/archive.dtd">
<archive xmlns="http://www.site.com/xmlns/xln/"
         xmlns="http://www.site.com/xmlns/cbn/"
         xmlns="http://www.site.com/xmlns/arc/"
         xmlns="http://www.site.com/xmlns/lbr/">
 <xln:document>
 <arc:partie id="KL001">
    <arc:date>10/05/61</arc:date>
    <arc:auteur>Henri Archit</arc:auteur>
    <lbr:auteur>Jaccques Labrosse</lbr:auteur>
    <arc:texte>
    ...
    </arc:texte>
 </arc:partie>
 <lbr:partie id="KL002">
    <lbr:date>15/06/61</lbr:date>
    <lbr:auteur>Jaccques Labrosse</lbr:auteur>
    <lbr:texte>
    ...
    </lbr:texte>
 </lbr:partie>
 ...
 </xln:document>
 <cbn:document>
 ...
 </cbn:document>
 ...
</archive>
```

Sélection du premier noeud *<arc:partie>*.

xpointer(id('KL001'))

Sélection de *<arc:auteur>* et de *<lbr:auteur>*.

xpointer((id('KL001')/*[2])range-to(id('KL001')/*[3]))

Sélection du noeud dont l'identifiant est *KL002* ou le cas échéant du second noeud *<lbr:partie>* du premier document *<xln:document>* dans l'élément racine *<archive>*.

xpointer(id('KL002')) xpointer(/*[1]/*[1]/*[2])

Sélection de l'élément *<arc:partie>* avec l'espace de noms approprié.

```
xmlns(arc=http://www.site.com/xmlns/arc/) xpointer(//arc:partie)
```

Sélection de l'élément *<lbr:auteur>* dans le noeud *<arc:partie>* avec les espaces de noms appropriés.

```
<!-- Cette expression doit être écrite sur une seule ligne ->
xmlns(arc=http://www.site.com/xmlns/arc/)
    xmlns(lbr=http://www.site.com/xmlns/lbr/)
        xpointer(//arc:partie/lbr:auteur)
```

Sélection du second noeud dans *<arc:partie>* avec l'espace de nom approprié.

```
xmlns(arc=http://www.site.com/xmlns/arc/)
    xmlns(lbr=http://www.site.com/xmlns/lbr/)
        xpointer(//arc:partie[position()=2])
```

3.2.2 / Les formes abrégées

La formulation d'un X-Pointeur peut être élaborée selon deux types : la forme standard avec sa construction *xpointer(expression)* ou une forme abrégée dénommée *child sequence* ou *Bare name*.

Cette dernière fonctionne par rapport au **nombre d'éléments dans chaque étage de l'arborescence** ou par rapport à la **valeur de l'attribut d'identification *id***.

La première séquence, ci-dessous, est une succession de *slash* (/) avec un entier *n* indiquant le niveau de l'élément à atteindre.

```
<!-- Child sequence -->
/n/n/n...

<!-- Correspond à -->
xpointer(/*[n]/*[n]/*[n]...)
```

La seconde formule, citée plus bas, est **équivalente à *xpointer(id(valeur_attribut_id))***, elle permet de **rechercher la valeur de l'attribut *id* dans l'arborescence d'un document XML** et cela quelque soit son niveau de profondeur.

```
<!-- Bare name -->
valeur_attribut_id

<!-- Correspond à -->
xpointer(id('valeur_attribut_id')S)
```

Considérons le document XML ci-dessous, afin de pointer des éléments précisément à l'aide des XPointers sous formes abrégées.

```
<!-- http://www.site.com/bibliotheque.xml -->
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE bibliotheque SYSTEM "../dtd/bibliotheque.dtd">
<biblitotheque>
  <categorie nom="roman">
  <livre isbn="2744110558">
     <titre>Napoléon - Le chant du départ</titre>
     <auteur>Max Gallo</auteur>
     <chapitres>
       <ouverture>
       C'était le 4 avril 1805, peu après l'aube...
       </ouverture>
       <chapitre id="chap1">
       Il n'avait pas encore dix ans,...
       </chapitre>
       <chapitre id="chap2">
       L'enfant est seul. Il a dû se contraindre...
       </chapitre>
       ...
       <chapitre id="chap39">
       C'est aujourd'hui, le 19 brumaire an VIII...
       </chapitre>
     </chapitres>
  </livre>
  ...
  </categorie>
  ...
</biblitotheque>
```

En premier lieu, nous allons tenter d'atteindre le premier livre, de la première catégorie de la bibliothèque. Pour cela, il suffit d'écrire ceci :

```
<index xlink:type="locator"
        xlink:href="bibliotheque.xml#/1/1/1">
   Napoléon - Tome I
</index>
```

Il est également possible d'utiliser l'identificateur *isbn* de l'élément *<livre>* de la manière suivante.

```
<index xlink:type="locator"
        xlink:href="bibliotheque.xml#2744110558">
  Napoléon - Tome I
</index>
```

Une autre expression permet encore une fois d'atteindre le premier élément *<livre>*.

```
<index xlink:type="locator"
        xlink:href="bibliotheque.xml#roman/1">
  Napoléon - Tome I
</index>
```

L'avant-dernière méthode utilise un XPointeur particulier appelé *bare name*. L'exemple ci-dessus montre que l'**on peut combiner sans problèmes les deux formes abrégées**.

Néanmoins, nous pouvons remarquer que les deux identificateurs utilisés précédemment sont *isbn* et *nom*. Si le document n'indique pas au processeur XML que ces deux attributs identificateurs ne sont pas du type *id*, alors ces exemples ne pourront guère fonctionner.

Il est impératif, de correctement déclarer les attributs d'identification sous peine de dysfonctionnement.

```
<!ATTLIST livre isbn ID #REQUIRED>
<!ATTLIST categorie nom ID #REQUIRED>
<!ATTLIST chapitre id ID #REQUIRED>
```

Maintenant, nous allons cibler le trente neuvième élément *<chapitre>* en utilisant en premier lieu la formulation *bare name*,

```
<index xlink:type="locator"
        xlink:href="bibliotheque.xml#chap39">
  Napoléon - Tome I - Capitre 39
</index>
```

ou par la forme abrégée dite *child sequence* :

```
<index xlink:type="locator"
        xlink:href="bibliotheque.xml#/1/1/1/3/40">
  Napoléon - Tome I - Capitre 39
</index>
```

ou encore en utilisant une combinaison des deux pour finir.

```
<index xlink:type="locator"
        xlink:href="bibliotheque.xml#2744110558/3/40">
  Napoléon - Tome I - Capitre 39
</index>
```

Il ne faut pas oublier que les nombres entiers dans la forme *child sequence* doivent comptabiliser le nombre d'éléments à chaque niveau de l'arborescence. Ainsi, il existe trois éléments enfants sous *<livre>* et <chapitres> est le troisième, de même pour le trente-neuvième chapître, en réalité il se situe à la position quarante, puisqu'un élément *<ouverture>* se situe à la première position.

3.3 / La localisation d'une ressource

Les Xpointers sont capables de cibler n'importe quel noeud, une position dans la valeur d'un noeud ou un intervalle de noeuds dans l'arborescence d'un document XML.

Les pointeurs ont la possiblité de **cibler une ressource par un chemin soit absolu, soit relatif**.

Lorsqu'une ressource est pointée directement par une expression, alors il s'agit d'un accès absolu.

En fait, un XPointer peut atteindre de façon directe, un élément quelconque par l'intermédiaire de son **identificateur unique de type ID**.

```
xpointer(id(identificateur))
```

De même que le noeud racine du document peut également être atteint directement par **la fonction *root()*** ou par son abréviation /.

```
xpointer(root())
```

```
xpointer(/)
```

L'autre manière d'accèder à une ressource, s'effectue par un cheminement relatif, car le pointage s'accomplit selon **des étapes successives**, c'est-à-dire qu'une ressource est atteinte par rapport à un noeud de départ, éventuellement d'autres noeuds intermédiaires et un noeud d'arrivée en l'occurrence, la cible.

Dans ce cas, **les axes nodaux, les tests de noeuds, les fonctions XPath, les prédicats, ainsi que les fonctions XPointer peuvent servir dans l'élaboration de telles expressions**.

```
<!-- Axe nodal, Fonction XPath, Prédicat-->
xpointer(/descendant::element[position() = 1])

<!-- Formulation XPointer abrégée -->
/1/2/2/6/9

<!-- Axe nodal, Test de noeud, Fonction XPath, Prédicat -->
xpointer(//element/child::text()[last()]

<!-- Axe nodal, Fonction XPath, Fonction XPointer -->
xpointer(id('identificateur')range-to(id('identificateur')))
```

3.4 / Les types de données

Les X-Pointeurs permettent de retourner non-seulement n'importe quels noeuds d'un document XML, mais également des points, des intervalles (ranges) ou encore des positions (location).

Le type de donnée *position*

(location) est une généralisation du concept de noeuds, points ou intervalles.

En fait, **la position dans le langage XPointer peut être de n'importe lequel de ces types de données**, soit un noeud XML quelconque (élément, attribut, textuel, espace de noms, commentaire ou instruction de traitement) mais aussi, un point de caractères ou de noeuds, ou encore un intervalle de texte ou d'éléments XML.

Voir le document XML qui sert de support aux exemples de cette section.

```
xpointer(//logiciel[position()=2])

xpointer(string-range(//logiciel[@code="13404148"]/nom,'2000',0,4))

xpointer(//editeur[2]/@lien)

xpointer(start-point(//logiciel[nom='Cooktop 2.200']))
```

Le premier XPointer retourne le noeud élément *<logiciel code="13413363">...</logiciel>*.

La seconde expression renvoie un intervalle de caractères *2000* trouvé au sein de l'élément *<nom>Web Expert 2000<nom>*.

Le troisième pointeur atteint un noeud d'attribut *lien="http://www.macromedia.com/software/"*.

Enfin, le dernier localise un point de départ situé entre le marqueur d'ouverture *<logiciel>* et l'élément *<nom>Cooktop 2.200</nom>*.

3.4.1 / Le type de données point

Le type de données *point* correspond à une position dans un élément contenant.

Voir le document XML qui sert de support aux exemples de cette section.

Cette position peut être définie par un noeud, appelé le noeud contenant, et un entier non-négatif appelé l'index.

```
xpointer(//noeud_contenant[index))
```

Lorsque le point atteint, se situe au sein d'un élément contenant d'autres éléments, le type de point se dénomme, alors, *point de type noeud*.

```
xpointer(start-point(//categorie))

xpointer(start-point(range(//logiciel[1])))
```

Les deux exemples ci-dessus localisent le même point dans l'arborescence du document XML. Le point se situe entre les marqueurs d'ouverture *<categorie nom="Editeur">* et *<logiciel>* dont le nom est *BBEdit 6.1*.

```
...
  <categorie nom="Editeur">[Point]
  <logiciel>
     <nom>BBEdit 6.1</nom>
...
```

Dans le premier exemple, l'index n'est pas spécifié, néanmoins par défaut il est égal à *0*, alors que le second indique *1*, permettant d'accéder au premier élément *<logiciel>* dans le document XML.

Si le point se situe dans un élément contenant du texte sans éléments enfants, alors le type de point s'appelle *type de point de caractères*.

```
xpointer(start-point(string-range(//*,'Expert', 7, 0)))
```

Ce pointeur détermine un point de départ se trouvant dans l'élément *<nom>Web Expert 2000</nom>*. Ce point se situe à la septième place après le début de la chaîne de mise en correspondance.

```
...
<logiciel code="13404148">
     <nom>Web Expert[Point] 2000</nom>
     <commentaire>
...
```

3.4.2 / Le type de données intervalle

Le type de données *range* représente un intervalle entre un X-Pointeur de début et un autre de fin.

L'intervalle obtenu est contigu, puisque tout ce qui se trouve entre les deux points de début et de fin est sélectionné.

Voir le document XML qui sert de support aux exemples de cette section.

Un intervalle peut contenir une chaîne de caractères ou un ensemble de noeuds.

Des XPointers aidés par des fonctions telles que *string-range()* ou *range()* permettent de parvenir à **créer des intervalles de chaînes de caractères**.

```
xpointer(string-range(//*,'Cooktop 2.200Un éditeur XML'))
```

La fonction *string-range()* retourne un intervalle de caractères provenant de données textuelles de deux éléments différents et contigus.

```
<nom>[Cooktop  2.200[</nom>
<commentaire>
 ]Un éditeur XML], XSLT, XPath et DTD...
</commentaire>
```

Des intervalles de noeuds peuvent être également réalisés par des fonctions XPointers telles que *range()*, *range-to()* ou *range-inside()*.

```
xpointer(range(//logiciel[5]))
```

Le pointeur ci-dessus sélectionne une plage de noeuds dans l'arborescence du document, en l'occurrence, le noeud *<logiciel>* avec tous ses sous-éléments.

```
</logiciel>
<logiciel>
   <nom>XML Spy 3.5</nom>
   <commentaire>
    La version 4 bientôt disponible.
   </commentaire>
   <editeur lien="http://www.xmlspy.com/default.html">
    Altova Inc.
   </editeur>
   <langue>EN</langue>
   <plateforme>Win</plateforme>
   <prix monnaie="$US">199,00</prix>
</logiciel>
<logiciel code="13404148">
```

3.5 / Les fonctions XPointers

Les fonctions XPointers permettent de manipuler des points, intervalles ou positions dans l'arborescence d'un document XML.

XPointer ajoute plusieurs fonctions à celles de XPATH. L'ensemble de ces fonctions doivent être intégrées dans des expressions XPointers.

Voir le document XML qui sert de support aux exemples de cette section.

La fonction *range-to*

détermine un intervalle allant du *chemin* jusqu'à la position passée en argument.

```
chemin range-to(jeu_de_positions)

xpointer(//logiciel[position()=2]range-to(//logiciel[position()=3]))
```

Cette expression retourne l'intervalle comprenant le second et le troisième élément *<logiciel>* dans l'arborescence du document.

La fonction *string-range*

renvoie les positions des chaînes de caractères trouvées dans l'arborescence d'un document XML.

Les deux arguments *nombre* sont optionnels.

- Le premier spécifie **le premier caractère à inclure dans l'intervalle résultante**. La valeur par défaut est 1 soit le premier caractère de la chaîne.
- Le second indique **la longueur de l'intervalle**, dont la valeur par défaut recouvre toute la chaîne.

```
chemin string-range(jeu_de_positions,
                    "chaîne",
                    nombre?,
                    nombre?)

xpointer(string-range(//*,'XML Spy',1,3))
```

Ce pointeur retourne la position des trois premiers caractères de la chaîne de mise en correspondance, soit *XML*, dans l'élément *<nom>XML Spy 3.5</nom>*.

La fonction *range*

renvoie un intervalle d'un document XML par rapport aux positions fournies en argument.

```
chemin range(jeu_de_positions)

xpointer(range(//logiciel[1]))
```

L'ensemble du premier noeud *<logiciel>* dans l'arborescence du document est sélectionné.

La fonction *range-inside*

retourne un intervalle comprenant les positions passées en argument.

```
chemin range-inside(jeu_de_positions)

xpointer(range-inside(//logiciel[1]))
```

Dans ce cas, seul les éléments à l'intérieur du premier noeud *<logiciel>* sont sélectionnés. L'élément contenant ne fait pas parti de l'intervalle retourné.

La fonction *start-point*

donne le point de départ de chacune des positions passées en argument.

```
chemin start-point(jeu_de_positions)

xpointer(start-point(//categorie[@nom='Editeur']))
```

Ce pointeur retourne le point de début de l'élément *<categorie...>...</categorie>*, c'est-à-dire le point juste

après le marqueur de début et avant le premier élément contenu.

La fonction *end-point*

donne le point de fin de chacune des positions passées en argument.

```
chemin end-point(jeu_de_positions)

xpointer(end-point(//categorie[@nom='Editeur']))
```

Ce pointeur retourne le point de fin de l'élément *<categorie...>...</categorie>*, c'est-à-dire le point juste avant le marqueur de fin et après le dernier élément contenu.

La fonction *here*

retourne la position unique du noeud élément contenant la cible de l'expression XPointer.

```
chemin here()

xpointer(//logiciel/nom/string-range(self::*, 'gratuit')/here())
```

Tout d'abord, le pointeur cible l'élément *<nom>* dans lequel se trouve la chaîne de caractère *gratuit*, puis retourne la position du noeud contenant le texte précité.

La fonction *origin*

détermine, à partir d'une expression XPointer, une unique position qui localise l'élément duquel un utilisateur ou un programme a initié la traversée d'un lien.

```
chemin origin()

xpointer(//logiciel/origin())
```

3.6 / Les identificateurs uniques

Les identificateurs uniques permettent de repérer des éléments dans l'arborescence d'un document XML.

Les expressions XPointers sont capables de localiser n'importe quel élément XML possédant un identificateur unique.

Toutefois, **ces identificateurs doivent avoir été déclarés au préalable dans la Définition de Type de Document** (DTD).

```
<!ATTLIST element id ID #REQUIRED>

<!ATTLIST element nom ID #REQUIRED>
```

De plus, **le document XML doit être validé en fonction de sa DTD** par un processeur XML. Dans le cas contraire, **les identificateurs pourraient ne pas être correctement interprétés**, si leur nom ne correspond pas à *id* qui lui peut l'être par défaut.

```
<!-- Identificateur sûr -->
<element id="identifiant"/>

<!-- Risque de mauvaise interprétation -->
<element nom="identifiant"/>
```

Un X-Pointeur atteint un élément disposant d'un identificateur unique par l'intermédiaire d'une fonction XPath *id()*.

```
xlink:href="uri#xpointer(id('identifiant'))"
```

Le XPointer peut, également, utiliser directement la valeur identifiante comme ceci :

```
xlink:href="URI#identifiant"
```

Les identificateurs uniques constituent l'un des moyens les plus simple pour atteindre un élément XML, cependant, tous ne peuvent comporter ce genre d'attribut identifieur.

C'est pourquoi, il est préférable d'**utiliser cette méthode d'accès que pour les principaux éléments** tels que de chapître, ou des produits, ou toutes rubriques quelconques...

3.7 / Les tests de noeuds

Les tests de noeuds permettent aux X-Pointeurs de cibler des noeuds en fonction de leur particularité.

Voir le document XML qui sert de support aux exemples de cette section.

Le test de noeud *

correspond à n'importe quel élément XML. Les autres noeuds, comme les commentaires, les instructions de traitement ne sont pas pris en compte par cette instruction.

```
xpointer(//logiciel[position() = 1]/following-sibling::*)
```

Cette expression sélectionne tous les éléments contenus dans le premier noeud *<logiciel>* y compris ce dernier.

Le test de noeud *node()*

représente l'ensemble des noeuds possibles dans un document XML.

```
xpointer(//node())
```

Ce pointeur sélectionne tous les noeuds du document XML.

Le test de noeud *text()*

ne traite que les données caractères analysées (Parsed Character DATA) des éléments XML.

```
xpointer(//logiciel/commentaire/self::text())
```

Cette expression fait référence à l'ensemble des textes présents dans les éléments *<commentaire>*.

Le test de noeud *comment()*

ne permet de sélectionner que les noeuds commentaires.

```
xpointer(/descendant::comment())
```

Ce XPointer atteint tous les commentaires présents dans tout le document.

Le test de noeud *processing-instruction()*

s'intéresse essentiellement aux instructions de traitement présentes dans l'arborescence d'un document XML.

```
xpointer(/descendant::processing-instruction())
```

Dans ce cas, tous les instructions de traitement seront localisées par cette expression.

Par ailleurs, le prologue du document *<?xml version="1.0"?>* et la déclaration *<!DOCTYPE>* ne font pas partie des noeuds d'un document XML.

3.8 / Les axes nodaux

Les axes nodaux permettent aux X-Pointeurs de localiser une ressource dans l'arborescence d'un document XML.

Voir le document XML qui sert de support aux exemples de cette section.

L'axe nodal *ancestor*

sélectionne tous les noeuds ancètres, c'est-à-dire le parent du noeud courant ainsi que tous les noeuds ascendants jusqu'au noeud racine du document XML.

```
xpointer(//logiciel[@code = '13413363'])/ancestor::*[position() = 1])
```

Le pointeur sélectionne dans ce cas l'élément parent *<categorie nom="Editeur">* placé en première position avant l'élément *<logiciel>* dont le code est égal à *13413363*.

L'axe *ancestor-or-self*

sélectionne comme l'axe précédent les noeuds ancètres y compris le noeud courant.

```
xpointer(/1/1/3/3/ancestor-or-self::*[position() = 1]/child::nom)
```

Ce pointeur sélectionne l'élément *<nom>CoffeeCup HTML Editor 8.9</nom>* enfant du premier parent *<logiciel>* de l'élément *<editeur lien="...">CoffeeCup Inc.</lien>* étant le troisième élément de son ancètre *<logiciel>* qui est également le troisième élément de *<categorie>* qui lui même correspond au premier élément de son unique père *<logitheque>* du noeud racine du document XML.

L'axe *attribute*

sélectionne les attributs du noeud courant.

```
xpointer(//logiciel[position() = 4]/editeur/attribute::lien)
```

Cette expression sélectionne *lien="http://xmleverywhere.com/cooktop/"*, l'attribut appartenant à l'élément *<editeur lien="...">XML Everywhere</editeur>* du quatrième élément dans l'arborescence du document XML.

L'axe *child*

sélectionne les éléments enfants du noeud courant.

```
xpointer(//logiciel[count(*) = 5]/child::nom)
```

L'expression atteint l'élément *<nom>Cooktop 2.200</nom>* du noeud *<logiciel>* dont le nombre d'éléments est égal à *5*.

L'axe *descendant*

sélectionne tous les éléments descendants du noeud courant, c'est-à-dire les fils, petit-fils, etc..

```
xpointer(/descendant::logiciel[position() = 5])
```

Cette expression atteint le cinquième élément *<logiciel>* petit-fils du noeud racine.

L'axe *descendant-or-self*

sélectionne tous les éléments descendants ainsi que le noeud courant lui-même.

```
xpointer(//categorie/descendant-or-self::logiciel[nom = 'BBEdit 6.1'])
```

Ce pointeur cible tous l'élément *<logiciel>* dont la valeur de *<nom>* est égale à *BBEdit 6.1* ainsi que tous ses noeuds descendants *<nom>*, *<commentaire>*, etc..

L'axe *following*

sélectionne tous les éléments suivant le noeud courant hormis les attributs et espaces de noms.

```
xpointer(//logiciel[@code = '13413363']
                    /prix/following::*[position() = 7])
```

Cette expression atteint le septième élément dans l'arborescence du document XML après l'élément *<prix>*

étant lui-même dans le noeud *<logiciel>* dont le code est *13413363*. Ce septième élément est *<prix...>49,00</prix>* du noeud *<logiciel>* qui suit. Pour parvenir à cet endroit, les éléments ont été comptés sans tenir compte de leur niveau hiérarchique dans l'ordre suivant :

1. *<logiciel>*
2. *<nom>CoffeeCup HTML Editor 8.9</nom>*
3. *<commentaire>Un logiciel...</commentaire>*
4. *<editeur lien="...">CoffeeCup Inc.</editeur>*
5. *<langue>US</langue>*
6. *<plateforme>Win</plateforme>*
7. *<prix monnaie="$US">49,00</prix>*

L'axe *following-sibling*

sélectionne les éléments suivants dont le parent est le même que celui du noeud courant.

```
xpointer(//logiciel[nom = 'XML Spy 3.5']
                /commentaire/following-sibling::*[position() = 4])
```

Le pointeur récupère l'élément *< prix>* positionné en quatrième position après le noeud *<commentaire>* de l'élément *<logiciel>* dont la valeur de son élément *<nom>* est égale à *XML Spy 3.5*. Une position au-delà de 4 provoquerait une erreur puisqu'il n'existe pas d'autres éléments frères après *<prix>*.

L'axe *parent*

sélectionne l'élément parent du noeud courant.

```
xpointer(//nom[position() = 6]/parent::*)
```

Cette expression cible le père *<logiciel code="13404148">...<logiciel>* du sixième élément *<nom>Web Expert 2000</nom>* dans l'arborescence du document.

L'axe *preceding*

sélectionne, sans tenir compte du niveau hiérarchique, tous les noeuds précédant le noeud courant hormis les attributs et les espaces de noms.

```
xpointer(//logiciel[last()]/preceding::*[position() = 1]
```

Le pointeur sélectionne l'élément *<prix monnaie="$US">199,00<prix>* précédant le dernier élément *<logiciel code="13404148">*.

L'axe *preceding-sibling*

sélectionne tous les noeuds précédents qui ont le même père que l'élément courant.

```
xpointer(//plateforme[self::* = Win][1]/preceding-sibling::nom)
```

Le XPointer localise l'élément frère *<nom>Dreamweaver 4</nom>* du premier noeud *<plateforme>* rencontré ayant la valeur *Win*.

Laxe *self*

sélectionne le noeud courant.

```
xpointer(/self::node())
```

L'expression pointe le noeud racine du document XML.

4 / XML Base

La spécification XML Base propose d'affecter à un document XML, une adresse URI (Uniform resource Identifier) utilisée comme base pour l'ensemble des URI relatifs présents dans les atributs *xlink:href* des éléments liants.

La mise en oeuvre de XML Base s'effectue par l'intermédiaire de l'adjonction à l'élément racine d'un document XML, d'un attribut *xml:base*, dont la valeur est une adresse URI de base appropriée.

Le préfixe d'espace de noms *xml:* dépend d'un URI standardisé par le World Wide Web Consortium (W3C).

http://www.w3.org/XML/1998/namespace

L'adresse URI de base se substitue à l'URI du répertoire courant du document, si bien que ce -dernier n'est pas contraint à être placé dans un répertoire précis.

De cette manière, **aucune équivoque ne pourrait subsister quant à la localisation des ressources pointées par les liens d'un document**, à condition évidemment, que ces ressources soient effectivement accessibles à partir du chemin indiqué.

Le fonctionnement de XML Base peut être apparenté à celui de la balise *<base>* du langage HTML, définissant également une adresse URL (Uniform Resource Locator) de base pour la page HTML.

Exemple :

```
<?xml version="1.0" standalone="yes"?>
<!DOCTYPE magasin [
  <!ELEMENT magasin (service+)>
  <!ATTLIST magasin
          xmlns:xml CDATA #FIXED "http://www.w3.org/XML/1998/namespace"
          xmlns:xlink CDATA #FIXED "http://www.w3.org/1999/xlink"
          xml:base CDATA #REQUIRED>
  <!ELEMENT service (produit)>
  <!ATTLIST service code ID #REQUIRED>
  <!ELEMENT produit (#PCDATA)>
  <!ATTLIST produit
          code ID #REQUIRED
          xlink:type (simple) #FIXED "simple"
          xlink:href CDATA #REQUIRED
          xlink:show (new|replace|embed|other|none) #IMPLIED "replace"
          xlink:actuate (onLoad|OnRequest|other|none) #IMPLIED "onRequest">
]>
<magasin xml:base="http://www.site.com/produit/"
                xmlns:xlink="http://www.w3.org/1999/xlink">
  <service code="A001">
  <produit code="4DE205"
                xlink:href="prod4DE205.xml">
     Soupe
  </produit>
  <produit code="2TM206"
                xlink:href="prod2TM206.xml">
     Condiment
  </produit>
  <produit code="3KJ227"
                xlink:href="prod3KJ227.xml">
     Conserve
  </produit>
  </service>
  <service code="A003">
  <produit code="1OU152"
                xlink:href="prod1OU152.xml">
     Lessive
  </produit>
  <produit code="8AH070"
                xlink:href="prod8AH070.xml">
     Essui-tout
  </produit>
  </service>
</magasin>
```

5 / XML Inclusions

La spécification XML Inclusions propose un mécanisme d'inclusion de ressources distantes dans un document XML.

La mise en oeuvre de cette proposition s'effectue par l'intermédiaire de l'insertion d'un élément *<xi:include>* à la position désirée pour l'inclusion de la ressource distante.

```
<xi:include href="URI" parse="type" encoding="encodage"/>
```

L'élément *<xi:include>* possède un espace de noms standardisé par le World Wide Web Consortium (W3C)

http://www.w3.org/2001/XInclude

Les inclusions peuvent se réaliser sur tout type de ressources XML, à l'image des noeuds d'éléments, d'attributs, de textes, d'instructions de traitement, de commentaire, mais aussi grâce aux XPointers, d'intervalles de noeuds ou de texte.

Les attributs

Attribut	Valeur	Description
href	URI	spécifie l'adresse URI de la ressource distante.
parse	xml text	indique le type de format souhaité.
encoding	CDATA	indique l'encodage de la ressource distante.

L'adresse URI de l'attribut *href* peut être absolue ou relative et peut contenir des fragments de localisation, soit des expressions XPointer.

L'attribut *parse* accepte deux valeurs,

- ***xml***, indiquant que les ressources distantes doivent être analysées comme des données XML, puis sont fusionnées dans le document.
- ***text***, signifiant que les ressources distantes doivent être incluses dans le document comme le contenu d'un noeud textuel.

Enfin, **l'attribut d'encodage *encoding*** accepte n'importe quel jeu de caractères standardisé, afin de traduire les données textuelles dans le format adéquat. Cet attribut n'a aucun effet lorsque *parse* possède la valeur *xml*.

La déclaration dans la DTD

L'utilisation des éléments et attributs d'inclusion nécessite une déclaration préalable dans la Définition de Type de Document (DTD).

```
<!ELEMENT xi:include EMPTY>
<!ATTLIST xi:include
        xmlns:xi CDATA #FIXED "http://www.w3.org/2001/XInclude"
        href CDATA #REQUIRED
        parse (xml|text) "xml"
        encoding CDATA #IMPLIED>
```

Exemple :

```
<!-- Document des éléments à inclure : page.xml -->
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE page SYSTEM "def_page.dtd">
<page>
<partie id="entete">
  Librairie du Grand Colbert
  ...
</partie>
<partie id="baspage">
  Copyright 2001 Groupe Library Ent. SA 100 000 Francs
  ...
</partie>
</page>
<!-- Document accueillant les inclusions : biblio.xml-->
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE bibliothèque [
  <!ELEMENT bibliothèque ANY>
  <!ELEMENT xi:include EMPTY>
  <!ATTLIST xi:include
          xmlns:xi CDATA #FIXED "http://www.w3.org/2001/XInclude"
          href CDATA #REQUIRED
          parse (xml|text) "xml"
          encoding CDATA #IMPLIED>
  <!ELEMENT livre (titre,numero)>
  <!ELEMENT collection (titre,numero)>
  <!ELEMENT titre (#PCDATA)>
  <!ELEMENT numero (#PCDATA)>
]>
<bibliothèque>
  <xi:include parse="xml" href="page.xml#xpointer(id('entete'))"/>
  <livre>
  <titre>La bible XML</titre>
  <numero>XML0023</numero>
  </livre>
  <collection>
  <titre>XML, par la pratique</titre>
  <numero>XML1023</numero>
  </collection>
  <piedpage>
  <xi:include
       parse="text"
       encoding="ISO-8859-1"
       href="page.xml#xpointer(string-range(/*[1]/*[2],
                                    'Copyright 2001 Groupe Library Ent.'))"/>
  </piedpage>
</bibliothèque>
<!-- Document résultant -->
<?xml version="1.0" encoding="ISO-8859-1"?>
<bibliothèque>
  <partie id="entete">
  Librairie du Grand Colbert
  ...
  </partie>
  <livre>
  <titre>La bible XML</titre>
  <numero>XML0023</numero>
  </livre>
  <collection>
  <titre>XML, par la pratique</titre>
  <numero>XML1023</numero>
  </collection>
  <piedpage>
  Copyright 2001 Groupe Library Ent.
  </piedpage>
</bibliothèque>
```

6 / Introduction

Le RDF (Resource Description Framework) ou cadre de description des ressources a été conçu par le World Wide Web Corsortium (W3C) dans le but de créer un support de traitement automatique des ressources présentes sur Internet.

Le principe est d'affecter des données, les metadata (meta-données) à la description des ressources.

Ces données particulières possèdent un vaste champ d'application, notamment pour une indexation plus fiable dans les moteurs de recherche, ou encore dans le commerce électronique avec des signatures numériques, etc..

La description des ressources est mis en oeuvre par l'intermédiaire d'éléments RDF décrivant des propriétés spécifiques comme les droits d'auteur et auxquelles est affectée à chacune une valeur, en l'occurrence dans ce cas le nom de l'auteur.

Le modèle de donnée élémentaire se repose sur trois types d'objets : les ressources, les propriétés et les déclarations.

Les ressources correspondent à tout objet présent sur le Web et accessible par le biais d'un URI (Uniform Resource identifier) comme une page web, des éléments composant cette-dernière (image, texte, intervalle de noeuds, etc.), mais aussi une collection de documents ou encore un site entier.

L'Altruiste --> http://www.laltruiste.com/
Cours XML --> http://wwww.laltruiste.com/coursxml/
Bandeau --> http://www.laltruiste.com/images/interface/bandeau.jpg
Livre --> http://www.laltruiste.com/librairie/librairie.xml#xpointer(id('livre'))

Les propriétés représentent des caractéristiques, des aspects, des attributs ou de relations particulières des ressources et en définissent les valeurs descriptives appropriées.

L'Altruiste --> création --> 1er septembre 2000
Cours XML --> création --> 1er juillet 2001
Bandeau --> format --> JPEG
Livre --> auteur --> Jean Edouard

Les déclarations sont composées d'un sujet (la ressource), d'un prédicat (la propriété) et d'un objet (la valeur) matérialisant les dépendances et les composants d'une relation RDF.

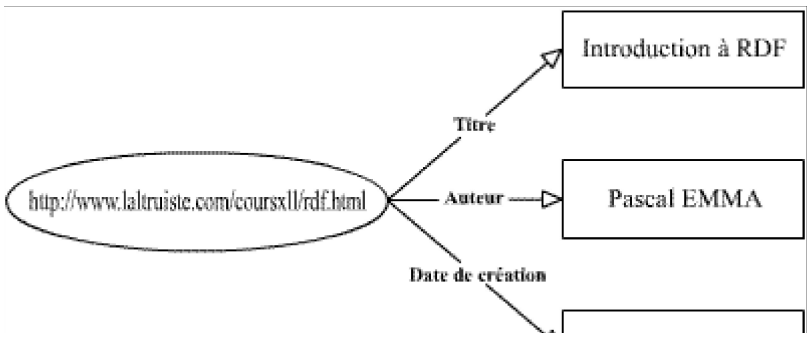
Par exemple, l'énoncé suivant :

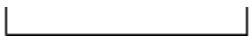
> Située à l'adresse *http://www.laltruiste.com/coursxll/rdf.html*, une page HTML posséde le titre *Introduction à RDF* a été créée le 21 août 2001 par Pascal EMMA.

met en exergue trois caractéristiques avec leur valeur associée, ainsi que l'URI identifiant la page HTML :

Page HTML (uri = "http://www.laltruiste.com/coursxll/rdf.html")
Titre --> Introduction à RDF
Création --> 21 août 2001
Auteur --> Pascal EMMA

A partir de cette constatation, il est possible de déterminer un diagramme symbolisant la déclaration RDF adéquate.





A partir de là, une syntaxe spécifique permet de formaliser un énoncé llittérale en une déclaration RDF compréhensible par des applications.

6.1 / Les éléments principaux

La syntaxe RDF s'appuie sur deux éléments principaux, il s'agit de l'élément racine *<rdf:RDF>* et de l'élément de description *<rdf:Description>*.

```
<rdf:RDF...>
  <rdf:Description...>
   ...
  </rdf:Description>
</rdf:RDF>
```

Tout d'abord, **les éléments RDF appartiennent à un espace de noms spécifique** standardisé par le World Wide Web Consortium (W3C), symbolisé par le préfixe *rdf:*.

http://www.w3.org/1999/02/22-rdf-syntax-ns#

La déclaration de cet espace de noms s'effectue par l'intermédiaire de l'attribut *xmlns:rdf*

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  ...
<rdf:RDF>
```

Il est possible également d'utiliser **un espace de noms implicite**, permettant d'éviter l'emploi systématique du préfixe *rdf:*.

```
<RDF xmlns="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  ...
<RDF>
```

L'élément racine

d'un document RDF se dénomme donc *<rdf:RDF>*, il possède un ou plusieurs attributs d'espaces de noms *xmlns:* et ne peut contenir que des éléments *<rdf:Decription>*.

Attribut	Description
xmlns:pfx="URI"	affecte un URI à l'espace de noms utilisé.
xmlns:lang="langue"	spécifie une langue pour le document RDF.

L'élément de description

***<rdf:Description>* comporte deux attributs *about* et *id*, et consitute un conteneur pour des éléments descriptifs**, tels que ceux fournis par le Dublin Core.

Attribut	Description
about="URI"	spécifie l'URI identificateur de la ressource à décrire.
id="Identificateur"	indique la création d'une nouvelle ressource.
aboutEach="URI"	spécifie l'URI se référant à chaque membre du conteneur.
aboutEachPrefix="Chaîne"	spécifie une chaîne de caractères se référant à chaque membre d'un conteneur *Bag* anonyme.
type="URI"	définit l'URI d'appartenance à la ressource.
bagid="Identificateur"	indique un identificateur vers un *Bag* (Emballage) de ressources.

Les éléments descriptifs peuvent être utilisés soit sous leur **forme habituelle d'élément** au sein des marqueurs *<rdf:Description>...</rdf:Description>*, soit sous une **forme abrégée**, c'est-à-dire comme des attributs de l'élément vide **<rdf:Description.../>**

Exemple :

L'exemple ci-dessous prend pour support, la déclaration citée dans l'introduction à RDF.

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:dc="http://purl.org/dc/elements/1.0/">
 <rdf:Description
        about="
coursxll/rdf.html">
 <dc:Title>Introdution à RDF</dc:Title>
 <dc:Creator>Pascal EMMA</dc:Creator>
 <dc:Date>21 août 2001</dc:Date>
 </rdf:Description>
</rdf:RDF>
```

En utilisant la syntaxe abrégée, le code devient :

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:dc="http://purl.org/dc/elements/1.0/">
 <rdf:Description
        about="
coursxll/rdf.html">
        dc:Title="Introdution à RDF"
        dc:Creator="Pascal EMMA"
        dc:Date="21 août 2001"/>
</rdf:RDF>
```

L'attribut *id* dans l'élément de description permet ici de faire référence à l'élément *<dc:Creator>* afin d'expliquer plus en détail sa signification.

```
<rdf:Description id="Creator">
 <rdf:type
      rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"
      xmlns:rdfs="http://www.w3.org/TR/1999/PR-rdf-schema-19990303#"/>
 <rdfs:label>Auteur/Créateur</rdfs:label>
 <rdfs:comment>
    La personne ou l'organisation principalement responsable
    de la création du contenu des ressources.
 </rdfs:comment>
 <rdfs:isDefinedBy rdf:resource = ""/>
</rdf:Description>
```

En savoir plus :



6.2 / L'attribut rdf:resource

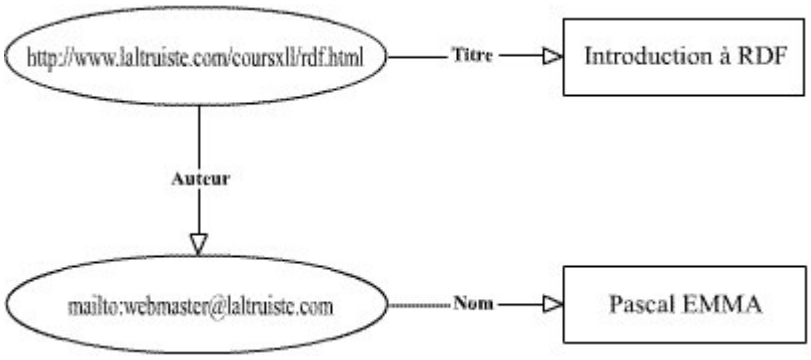
Dans le cas d'une déclaration RDF faisant intervenir plusieurs ressources en relation entre elles, la construction syntaxique devient, alors, plus complexe.

En prenant l'énoncé suivant en compte,

> Située à l'adresse *http://www.laltruiste.com/coursxll/rdf.html*, cette page HTML a été créée par Pascal EMMA, dont l'adresse email est *mailto:webmaster@laltruiste.com*.

on remarque que les deux objets *page HTML* et *Pascal EMMA* possèdent chacun un identifiant unique.

Page HTML (uri = "http://www.laltruiste.com/coursxll/rdf.html")
Titre --> Introduction à RDF
Auteur --> Pascal EMMA (uri = "mailto:webmaster@laltruiste.com")



La traduction en langage RDF peut devenir :

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:dc="http://purl.org/dc/elements/1.0/">
  <rdf:Description
          about="
coursxll/rdf.html">
  <dc:Title>Introdution à RDF</dc:Title>
  <dc:Creator
          rdf:resource="mailto:webmaster@laltruiste.com"/>
  </rdf:Description>

  <rdf:Description
          about="mailto:webmaster@laltruiste.com">
  <dc:Title>
     Pascal EMMA
  </dc:Title>
  </rdf:Description>
</rdf:RDF>
```

Cette écriture permet de **rendre disponible la seconde ressource au bénéfice de la première**.

Par ailleurs, il est également possible **qu'une ressource puisse être utilisée par plusieurs autres** comme le montre le code suivant :

```
<rdf:RDF
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:dc="http://purl.org/dc/elements/1.0/">
  <rdf:Description
              about="
coursxll/rdf.html">
  <dc:Title>Introdution à RDF</dc:Title>
  <dc:Creator
              rdf:resource="mailto:webmaster@laltruiste.com"/>
  </rdf:Description>

  <rdf:Description
              about="
coursxll/element_rdf.html">
  <dc:Creator
              rdf:resource="mailto:webmaster@laltruiste.com"/>
  </rdf:Description>

  <rdf:Description
              about="mailto:webmaster@laltruiste.com">
  <dc:Title>
     Pascal EMMA
  </dc:Title>
  </rdf:Description>
</rdf:RDF>
```

L'attribut *rdf:resource* permet, donc, d'établir une liaison entre deux ressources.

Il **se place dans un élément descriptif**, ici *<dc:Creator>* et **pointe sur la ressource sollicitée par l'intermédiaire d'un identifiant**, dans l'exemple précitée une adresse email.

6.3 / L'attribut parseType

Un élément descriptif (propriété), tel que *<dc:Title>* peut contenir des données textuelles analysables (PCDATA), d'autres ressources, mais aussi des marqueurs XML. Pour cela, l'élément doit accueillir un attribut spécifique, *parseType* avec la valeur adéquate.

```
<dc:Title parseType="valeur">
  Marqueurs XML...
</dc:Title>
```

Valeur	Description
Literal	accepte un contenu XML, marqueurs, attributs, PCDATA.
Resource	autorise de nouvelles ressources de descriptions.

L'attribut *parseType* possède deux valeurs :

- ***Literal*** indiquant un contenu XML, comme des marqueurs XML ou du PCDATA (Parsed Character DATA) comme le montre l'exemple ci-dessous.

```
<rdf:RDF
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:dc="http://purl.org/dc/elements/1.0/"
      xmlns:alt="
">
  <rdf:Description
              about="
coursxll/rdf.html">
  <dc:Title parseType="Literal">
     <[CDATA[Introduction à RDF]]>
     <alt:commentaire>
      (Resource Description Framework)
     </alt:commentaire>
  </dc:Title>
  </rdf:Description>
</rdf:RDF>
```

- ***Resource*** siginfiant que l'élément descriptif ne peut accepter que des éléments descriptifs avec un attribut *rdf:Resource*.

```
<rdf:RDF
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:dc="http://purl.org/dc/elements/1.0/"
      xmlns:dcq="http://purl.org/metadata/dublin_core_qualifiers#">
  <rdf:Description
             about="
coursxll/rdf.html">
  <dc:Creator parseType="Resource">
     <rdf:value>
      L'Altruiste
     </rdf:value>
     <dcq:AgentType
          rdf:resource="http://purl.org/metadata/dublin_core_qualifiers#Editor"/>
  </dc:Creator>
  </rdf:Description>
</rdf:RDF>
```

Dans l'exemple ci-dessus, l'élément *<dc:Title>* possèdant l'attribut *parseType="Resource"* contient un élément *<rdf:value>* comprenant la valeur de la déclaration RDF et un élément *<dcq:AgentType>* indiquant le role de l'agent par rapport à la ressource.

6.4 / Les éléments du schéma RDF

Classe principales
rdfs:Resource
rdf:Property
rdfs:Class
ropriétés principales
rdf:type
rdfs:subClassOf
rdfs:subPropertyOf
rdfs:seeAlso
rdfs:isDefinedBy
Contraintes principales
rdfs:ConstraintResource
rdfs:ConstraintProperty
rdfs:range
rdfs:domain

Propriétés de documentation
rdfs:label
rdfs:comment
Modèle et concepts de syntaxe
rdfs:Literal
rdf:Statement
rdf:Subject
rdf:Predicate
rdf:object
rdfs:Container
rdf:Bag
rdf:Seq
rdf:Alt
rdfs:ContainerMembershipProperty
rdf:value

rdfs:Resource

La classe *rdf:Resource* représente des expressions RDF, soit des ressources.

rdf:Property

La classe *rdf:Property* représente les propriétés des ressources RDF.

```
  ...
<rdf:Property
      about="http://www.w3.org/1999/02/22-rdf-syntax-ns#predicate">
  <rdfs:label xml:lang="en">predicate</rdfs:label>
  <rdfs:label xml:lang="fr">prédicat</rdfs:label>
  <rdf:type
      resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
  <rdfs:domain
      rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Statement"/>
  <rdfs:range
      rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
</rdf:Property>
...
```

rdfs:Class

La classe *rdfs:Class* représente n'importe quel *type* ou *catégorie* de ressources, comme des pages Web, des individus, des types de document, des bases de données ou des concepts abstraits.

```
  ...
<rdfs:Class
      rdf:about="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag">
  <rdfs:label xml:lang="en">Bag</rdfs:label>
  <rdfs:label xml:lang="fr">Ensemble</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Container"/>
</rdfs:Class>
...
```

rdf:type

La propriété *rdf:Type* définit l'appartenance d'une ressource à un type spécifique dont la valeur est une certaine classe *rdfs:Class*.

```
<rdf:RDF
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
  <rdf:Description id="Livre">
    <rdf:type
      resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
  </rdf:Description>
</rdf:RDF>
```

rdfs:subClassOf

La propriété spécifie une relation de sous-ensemble/sur-ensemble entre des classes.

```
<rdf:RDF
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
<rdf:Description id="Document">
  <rdf:type
      resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
  <rdfs:subClassOf
      rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>
</rdf:Description>
</rdf:RDF>
```

rdfs:subPropertyOf

La propriété *rdfs:subPropertyOf* est une instance de *rdf:Property*, utilisée afin de spécifier qu'une propriété est une spécialisation d'une autre.

```
<rdf:RDF
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
  <rdf:Description id="Livre">
    <rdf:type
      resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
  </rdf:Description>
  <rdf:Description id="Titre">
    <rdf:type
      resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
   <rdfs:subPropertyOf rdf:resource="#Livre"/>
  </rdf:Description>
</rdf:RDF>
```

rdfs:seeAlso

La propriété *rdfs:seeAlso* indique qu'une ressource contient une information à propos du sujet de la ressource.

```
  ...
<rdf:Description
      about="http://www.w3.org/2000/01/rdf-schema#">
  <rdfs:seeAlso
      resource="http://www.w3.org/2000/01/rdf-schema-more"/>
</rdf:Description>
...
```

rdfs:isDefinedBy

La propriété *rdfs:isDefinedBy* spécifie une ressource afin de définir le sujet d'une autre.

```
<rdf:RDF
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
  <rdf:Description id="Montant">
    <rdf:type
      resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#Commande"/>
    <rdfs:domain rdf:resource="#produit"/>
    <rdfs:range
      rdf:resource="http://www.w3.org/2000/03/example/classes#Number"/>
```

```
    </rdf:Description>
  </rdf:RDF>
```

rdfs:ConstraintResource

La classe *rdfs:ConstraintResource* définit des formes de contraintes pour des ressources *rdfs:Resource*.

rdfs:ConstraintProperty

La classe *rdfs:ConstraintProperty* définit des formes de contraintes pour des propriétés *rdf:Property*.

rdfs:range

La propriété *rdfs:range* spécifie une classe utilisée par *rdfs:ConstraintProperty*, pour créer des contraintes sur des valeurs de propriétés.

```
<rdf:RDF
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
   <rdf:Description id="Montant">
     <rdf:type
       resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
     <rdfs:domain rdf:resource="#Commande"/>
     <rdfs:domain rdf:resource="#produit"/>
     <rdfs:range
       rdf:resource="http://www.w3.org/2000/03/example/classes#Number"/>
   </rdf:Description>
</rdf:RDF>
```

rdfs:domain

La propriété *rdfs:domain* spécifie une classe utilisée par *rdfs:ConstraintProperty*, pour créer des contraintes sur le domaine d'action des propriétés.

```
<rdf:RDF
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
   <rdf:Description id="Facture">
     <rdf:type
       resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
     <rdfs:domain rdf:resource="#Commande"/>
     <rdfs:range rdf:resource="#Client"/>
   </rdf:Description>
</rdf:RDF>
```

rdfs:comment

La propriété *rdfs:comment* est untilisé pour fournir une description sur une ressource.

```
<rdf:RDF
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
<rdfs:Class rdf:id="Entreprise">
   <rdfs:comment>
     La classe des entreprises.
   </rdfs:comment>
<rdf:type
      resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
<rdfs:subClassOf
      rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>
</rdfs:Class>
</rdf:RDF>
```

rdfs:label

La propriété *rdfs:label* est utilisée pour fournir un label à un nom de ressource.

```
<rdf:RDF
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
<rdfs:Class rdf:id="pop_2001">
   <rdfs:label xml:lang="FR">
    Population_mondiale_en_2001
   </rdfs:label>
   <rdfs:label xml:lang="EN">
    World_population_in_2001
   </rdfs:label>
<rdf:type
      resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
<rdfs:subClassOf
      rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>
</rdfs:Class>
</rdf:RDF>
```

rdfs:Literal

La propriété *rdfs:Literal* correspond à un ensemble appelé les *Littéraux*, comme des chaînes de caractères, dans le modèle formel pour RDF.

rdf:Statement

La propriété *rdf:Statement* correspond à un ensemble appelé *Déclaration* dans le modèle formel pour RDF.

rdf:subject

La propriété *rdf:subject* correspond à une propriété appelée le *sujet*, soit une ressouce décrite par un délclaration, dans le modèle formel pour RDF.

rdf:predicate

La propriété *rdf:predicate* correspond à la propriété appelée le *Prédicat*, soit la propriété, dans le modèle formel pour RDF.

rdf:object

La propriété *rdf:object* correspond à la propriété appelée l'*Objet*, soit la valeur de la propriété, dans le modèle formel pour RDF.

rdfs:Container

La classe *rdfs:Container* est utilisée pour conserver les classes conteneurs.

rdf:Bag

La propriété *rdf:Bag* correspond à la classe appelée *Emballage* dans le modème formel pour RDF.

```
<rdf:RDF
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:dc="http://purl.org/dc/elements/1.0/">
   <rdf:Description
      about="
coursxll/rdf.html">
     <dc:Title>
     Introduction à RDF
     </dc:Title>
     <dc:Subject>
     <rdf:Bag id="liens">
       <rdf:li
          resource="http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/"/>
       <rdf:li
          resource="http://www.la-grange.net/w3c/REC-rdf-syntax/"/>
       <rdf:li
          resource="http://www.w3.org/TR/2000/CR-rdf-schema-20000327/"/>
```

```
      </rdf:Bag>
      </dc:Subect>
    </rdf:Description>
  </rdf:RDF>
```

rdf:Seq

La propriété *rdf:Seq* correspond à la classe appelée *Séquence* dans le modème formel pour RDF.

```
  <rdf:RDF
        xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:dc="http://purl.org/dc/elements/1.0/">
    <rdf:Description
        about="
coursxll/rdf.html">
      <dc:Title>
      Introduction à RDF
      </dc:Title>
      <dc:Subject>
      <rdf:Seq id="Complémentarité">
        <rdf:li>Le langage XML&lt:/rdf:li>
        <rdf:li>Le langage XSL&lt:/rdf:li>
        <rdf:li>Le langage XLL&lt:/rdf:li>
        <rdf:li>Le langage XHTML&lt:/rdf:li>
        <rdf:li>Le langage HTML&lt:/rdf:li>
      </rdf:Seq>
      </dc:Subject>
    </rdf:Description>
  </rdf:RDF>
```

rdf:Alt

La propriété *rdf:Alt* correspond à la classe appelé *Alternative* dans le modèle formel pour RDF.

```
  <rdf:RDF
        xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:dc="http://purl.org/dc/elements/1.0/">
    <rdf:Description id="Situation_Familiale">
      <rdf:Alt>
      <rdf:li>Célibataire&lt:/rdf:li>
        <rdf:li>Marié&lt:/rdf:li>
        <rdf:li>Pascé&lt:/rdf:li>
        <rdf:li>Concubin&lt:/rdf:li>
        <rdf:li>Divorcé&lt:/rdf:li>
        <rdf:li>Veuf&lt:/rdf:li>
      </rdf:Alt>
    </rdf:Description>
  </rdf:RDF>
```

rdfs:ContainerMembershipProperty

La classe *rdfs:ContainerMembershipProperty* a comme membres les propriétés _1, _2, _3, etc. utilisées pour indiquer l'effectif d'un conteneur.

rdf:value

La propriété *rdf:value* correspond à la valeur de la propriété.

```
  <rdf:RDF
        xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:dc="http://purl.org/dc/elements/1.0/">
    <rdf:Property id="Article">
      <rdf:value>150</rdf:value>
    </rdf:property>
  </rdf:RDF>
```

En savoir plus :