

1 Faire des requêtes avec XPATH

1 Faire des requêtes avec XPATH

- Modèle de donnée
- Expressions de chemin
- Prédicats
- Fonctions
- Syntaxe abrégée

Introduction

- Un document XML est un arbre avec des nœuds éléments, des nœuds textes, des nœuds attributs, ...
- XPath est un langage permettant de désigner des nœuds dans l'arbre document, en décrivant des chemins dans cet arbre.
- On peut aussi désigner un nœud, indépendamment de tout chemin, grâce aux identifiants (attributs de type ID).
- Les expressions XPath sont utilisées dans de nombreux langages pour XML (en particulier, historiquement c'est une partie de XSLT)
- XPath 1 : juste des expressions de chemin. Recommandation depuis 1999.
- XPath 2 : des boucles for, des conditions if, des variables, Recommandation depuis 2006 (à l'étude depuis 2001)

Introduction

- Un document XML est un arbre avec des nœuds éléments, des nœuds textes, des nœuds attributs, ...
- XPath est un langage permettant de désigner des nœuds dans l'arbre document, en décrivant des chemins dans cet arbre.
- On peut aussi désigner un nœud, indépendamment de tout chemin, grâce aux identifiants (attributs de type ID).
- Les expressions XPath sont utilisées dans de nombreux langages pour XML (en particulier, historiquement c'est une partie de XSLT)
- XPath 1 : juste des expressions de chemin. Recommandation depuis 1999.
- XPath 2 : des boucles for, des conditions if, des variables, Recommandation depuis 2006 (à l'étude depuis 2001)

Introduction

- Un document XML est un arbre avec des nœuds éléments, des nœuds textes, des nœuds attributs, ...
- XPath est un langage permettant de désigner des nœuds dans l'arbre document, en décrivant des chemins dans cet arbre.
- On peut aussi désigner un nœud, indépendamment de tout chemin, grâce aux identifiants (attributs de type ID).
- Les expressions XPath sont utilisées dans de nombreux langages pour XML (en particulier, historiquement c'est une partie de XSLT)
- XPath 1 : juste des expressions de chemin. Recommandation depuis 1999.
- XPath 2 : des boucles for, des conditions if, des variables, Recommandation depuis 2006 (à l'étude depuis 2001)

Introduction

- Un document XML est un arbre avec des nœuds éléments, des nœuds textes, des nœuds attributs, ...
- XPath est un langage permettant de désigner des nœuds dans l'arbre document, en décrivant des chemins dans cet arbre.
- On peut aussi désigner un nœud, indépendamment de tout chemin, grâce aux identifiants (attributs de type ID).
- Les expressions XPath sont utilisées dans de nombreux langages pour XML (en particulier, historiquement c'est une partie de XSLT)
- XPath 1 : juste des expressions de chemin. Recommandation depuis 1999.
- XPath 2 : des boucles for, des conditions if, des variables, Recommandation depuis 2006 (à l'étude depuis 2001)

Introduction

- Un document XML est un arbre avec des nœuds éléments, des nœuds textes, des nœuds attributs, ...
- XPath est un langage permettant de désigner des nœuds dans l'arbre document, en décrivant des chemins dans cet arbre.
- On peut aussi désigner un nœud, indépendamment de tout chemin, grâce aux identifiants (attributs de type ID).
- Les expressions XPath sont utilisées dans de nombreux langages pour XML (en particulier, historiquement c'est une partie de XSLT)
- XPath 1 : juste des expressions de chemin. Recommandation depuis 1999.
- XPath 2 : des boucles for, des conditions if, des variables, . . .
.. Recommandation depuis 2006 (à l'étude depuis 2001)

Introduction

- Un document XML est un arbre avec des nœuds éléments, des nœuds textes, des nœuds attributs, ...
- XPath est un langage permettant de désigner des nœuds dans l'arbre document, en décrivant des chemins dans cet arbre.
- On peut aussi désigner un nœud, indépendamment de tout chemin, grâce aux identifiants (attributs de type ID).
- Les expressions XPath sont utilisées dans de nombreux langages pour XML (en particulier, historiquement c'est une partie de XSLT)
- XPath 1 : juste des expressions de chemin. Recommandation depuis 1999.
- XPath 2 : des boucles for, des conditions if, des variables, . . .
.. Recommandation depuis 2006 (à l'étude depuis 2001)

Le modèle de données de XPath 1.0

Le résultat d'une requête XPath est un **ensemble de nœuds** qu'il faut voir comme un ensemble de références vers des nœuds de l'arbre XML. Qui dit ensemble dit

- non ordonné
- sans doublon

Le modèle de données de XPath 1.0

Le résultat d'une requête XPath est un **ensemble de nœuds** qu'il faut voir comme un ensemble de références vers des nœuds de l'arbre XML. Qui dit ensemble dit

- non ordonné
- sans doublon

Le modèle de données de XPath 1.0

Le résultat d'une requête XPath est un **ensemble de nœuds** qu'il faut voir comme un ensemble de références vers des nœuds de l'arbre XML. Qui dit ensemble dit

- non ordonné
- sans doublon

Le modèle de données de XPath 2.0

Le résultat d'une requête XPath est une **séquence de nœuds ou de valeurs**. Qui dit séquence dit

- ordonné
- doublons possibles

Le modèle de données de XPath 2.0

Le résultat d'une requête XPath est une **séquence de nœuds ou de valeurs**. Qui dit séquence dit

- ordonné
- doublons possibles

Le modèle de données de XPath 2.0

Le résultat d'une requête XPath est une **séquence de nœuds ou de valeurs**. Qui dit séquence dit

- ordonné
- doublons possibles

XPATH

Il y a 7 sortes de nœuds :

- 1 document
- 2 element
- 3 attribute
- 4 text
- 5 namespace
- 6 processing instruction
- 7 comment

XPATH

Il y a 7 sortes de nœuds :

1 document

2 element

3 attribute

4 text

5 namespace

6 processing instruction

7 comment

XPATH

Il y a 7 sortes de nœuds :

1 document

2 element

3 attribute

4 text

5 namespace

6 processing instruction

7 comment

XPATH

Il y a 7 sortes de nœuds :

1 document

2 element

3 attribute

4 text

5 namespace

6 processing instruction

7 comment

XPATH

Il y a 7 sortes de nœuds :

1 document

2 element

3 attribute

4 text

5 namespace

6 processing instruction

7 comment

XPATH

Il y a 7 sortes de nœuds :

1 document

2 element

3 attribute

4 text

5 namespace

6 processing instruction

7 comment

XPATH

Il y a 7 sortes de nœuds :

- 1 document
- 2 element
- 3 attribute
- 4 text
- 5 namespace
- 6 processing instruction
- comment

XPATH

Il y a 7 sortes de nœuds :

- 1 document
- 2 element
- 3 attribute
- 4 text
- 5 namespace
- 6 processing instruction
- 7 comment

XPATH

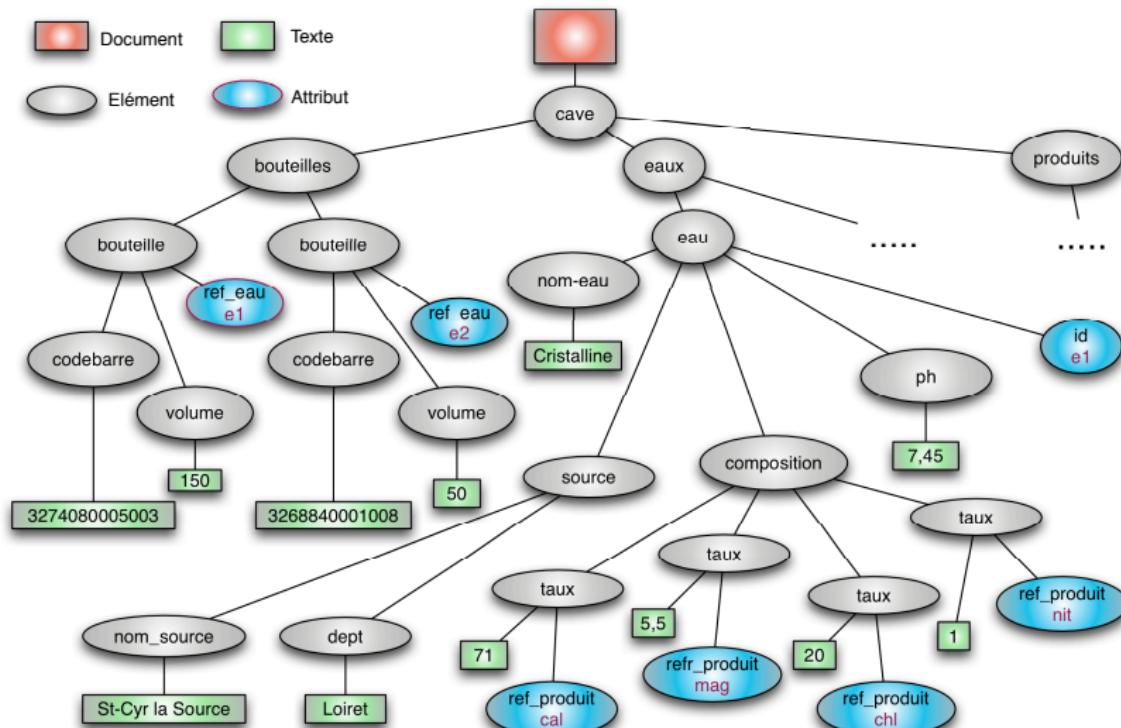
Il y a 7 sortes de nœuds :

- 1 document
- 2 element
- 3 attribute
- 4 text
- 5 namespace
- 6 processing instruction
- 7 comment

Exemple de document

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE cave SYSTEM "./eaux.dtd">
<cave>
  <bouteilles>
    <bouteille ref_eau="e1"> <codebarre>3274080005003</codebarre> <volume>150</volume></bouteille>
    <bouteille ref_eau="e2"> <codebarre>3268840001008</codebarre> <volume>50</volume></bouteille>
    <bouteille ref_eau="e2"> <codebarre>3546543211234</codebarre> <volume>150</volume></bouteille>
  </bouteilles>
  <eaux>
    <eau id="e1">
      <nom_eau>Cristalline </nom_eau>
      <source><nom_source>St-Cyr la Source</nom_source> <dept>Loiret</dept></source>
      <composition>
        <taux ref_produit="cal">71</taux> <taux ref_produit="mag">5,5</taux>
        <taux ref_produit="chl">20</taux> <taux ref_produit="nit">1</taux>
      </composition>
      <ph>7,45</ph>
    </eau>
    <eau id="e2"> ..... </eau>
  </eaux>
  <produits>
    <produit id="cal"><nom_produit>Calcium</nom_produit> <formule>Ca+</formule></produit>
    .....
  </produits>
</cave>
```

Arbre correspondant



Ordre du document

Les requêtes XPath utilisent un ordre total sur les nœuds, appelé ordre du document.

- C'est l'ordre en profondeur d'abord et de gauche à droite lorsque le document est représenté par un arbre.
- Pour les éléments, c'est donc l'ordre dans lequel on rencontre les balises ouvrantes lorsque l'on considère la représentation textuelle du document.
- Les nœuds namespace sont immédiatement successeurs du nœud élément qui leur est associé.
- Les nœuds attributs viennent immédiatement après ces nœuds namespace.
- Attention, l'ordre des nœuds attributs d'un même élément dépend de l'implémentation.

Ordre du document

Les requêtes XPath utilisent un ordre total sur les nœuds, appelé ordre du document.

- C'est l'ordre en profondeur d'abord et de gauche à droite lorsque le document est représenté par un arbre.
- Pour les éléments, c'est donc l'ordre dans lequel on rencontre les balises ouvrantes lorsque l'on considère la représentation textuelle du document.
- Les nœuds namespace sont immédiatement successeurs du nœud élément qui leur est associé.
- Les nœuds attributs viennent immédiatement après ces nœuds namespace.
- Attention, l'ordre des nœuds attributs d'un même élément dépend de l'implémentation.

Ordre du document

Les requêtes XPath utilisent un ordre total sur les nœuds, appelé ordre du document.

- C'est l'ordre en profondeur d'abord et de gauche à droite lorsque le document est représenté par un arbre.
- Pour les éléments, c'est donc l'ordre dans lequel on rencontre les balises ouvrantes lorsque l'on considère la représentation textuelle du document.
- Les nœuds namespace sont immédiatement successeurs du nœud élément qui leur est associé.
- Les nœuds attributs viennent immédiatement après ces nœuds namespace.
- Attention, l'ordre des nœuds attributs d'un même élément dépend de l'implémentation.

Ordre du document

Les requêtes XPath utilisent un ordre total sur les nœuds, appelé ordre du document.

- C'est l'ordre en profondeur d'abord et de gauche à droite lorsque le document est représenté par un arbre.
- Pour les éléments, c'est donc l'ordre dans lequel on rencontre les balises ouvrantes lorsque l'on considère la représentation textuelle du document.
- Les nœuds `namespace` sont immédiatement successeurs du nœud élément qui leur est associé.
- Les nœuds attributs viennent immédiatement après ces nœuds `namespace`.
- Attention, l'ordre des nœuds attributs d'un même élément dépend de l'implémentation.

Ordre du document

Les requêtes XPath utilisent un ordre total sur les nœuds, appelé ordre du document.

- C'est l'ordre en profondeur d'abord et de gauche à droite lorsque le document est représenté par un arbre.
- Pour les éléments, c'est donc l'ordre dans lequel on rencontre les balises ouvrantes lorsque l'on considère la représentation textuelle du document.
- Les nœuds namespace sont immédiatement successeurs du nœud élément qui leur est associé.
- Les nœuds attributs viennent immédiatement après ces nœuds namespace.
- Attention, l'ordre des nœuds attributs d'un même élément dépend de l'implémentation.

Ordre du document

Les requêtes XPath utilisent un ordre total sur les nœuds, appelé ordre du document.

- C'est l'ordre en profondeur d'abord et de gauche à droite lorsque le document est représenté par un arbre.
- Pour les éléments, c'est donc l'ordre dans lequel on rencontre les balises ouvrantes lorsque l'on considère la représentation textuelle du document.
- Les nœuds namespace sont immédiatement successeurs du nœud élément qui leur est associé.
- Les nœuds attributs viennent immédiatement après ces nœuds namespace.
- Attention, l'ordre des nœuds attributs d'un même élément dépend de l'implémentation.

Les expressions de chemin

- Une expression de chemin XPath décrit une séquence de nœuds extrémités du chemin, en s'appuyant sur la structure arborescente du document.
- Une expression consiste en une séquence de pas séparés par / ou //.
- On évalue un pas sur tous les nœuds de la séquence calculée au pas précédent et on concatène les séquences résultats.

Etape

axe ::= filtre[prédicat]

Etape

axe ::= filtre[prédicat]

axe navigationnel

Etape

axe ::= filtre[prédicat]

axe navigationnel

choix de type de noeud

Etape

axe ::= filtre[prédicat]

axe navigationnel

sélection supplémentaire optionnelle

choix de type de noeud

Etape

axe ::= filtre[prédicat]

axe navigationnel
child,
next-sibling,
...

sélection supplémentaire optionnelle
choix de type de noeud

Etape

axe ::= filtre[prédicat]

axe navigationnel
child,
next-sibling,
...

choix de type de noeud

*

,

node(),

text()

...

sélection supplémentaire optionnelle

Etape

axe ::= filtre[prédicat]

axe navigationnel
child,
next-sibling,
...

choix de type de noeud
*,
node(),
text()

...

sélection supplémentaire optionnelle
position()=2,
attribute::id="e3",
...

Les expressions de chemin

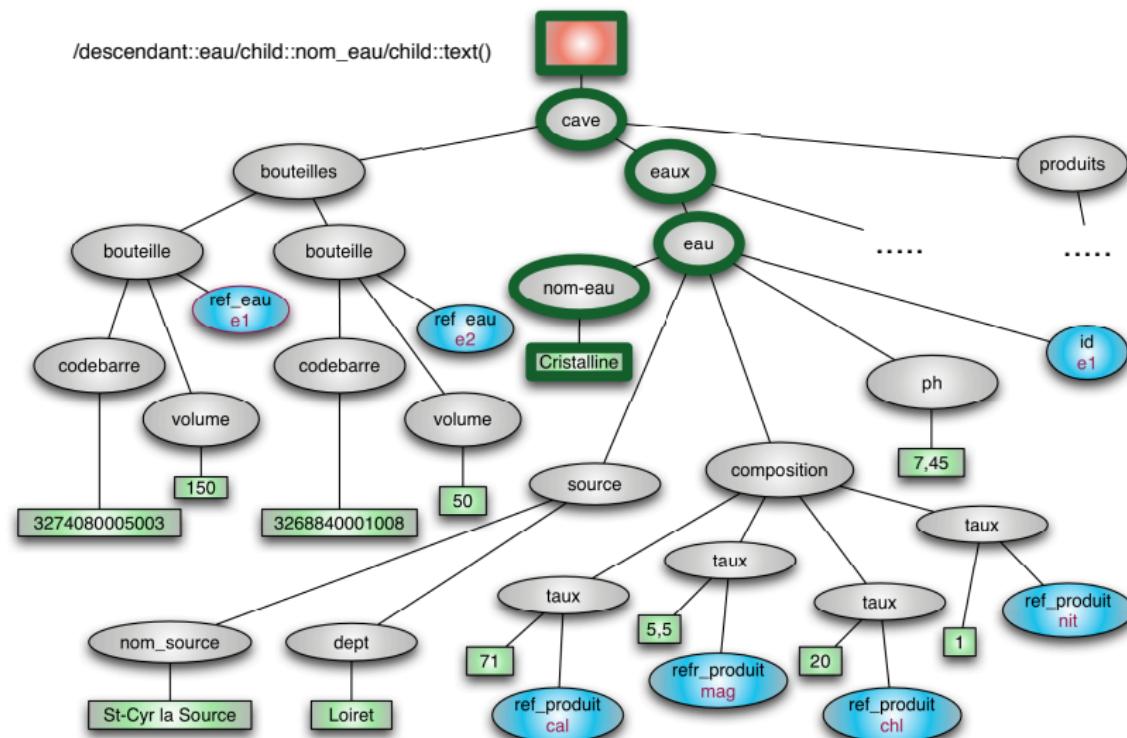
- Une expression de chemin XPath décrit une séquence de nœuds extrémités du chemin, en s'appuyant sur la structure arborescente du document.
- Une expression consiste en une séquence de pas séparés par / ou //.
- On évalue un pas sur tous les nœuds de la séquence calculée au pas précédent et on concatène les séquences résultats.

```
/descendant::eau/child::nom_eau/child::text()
```

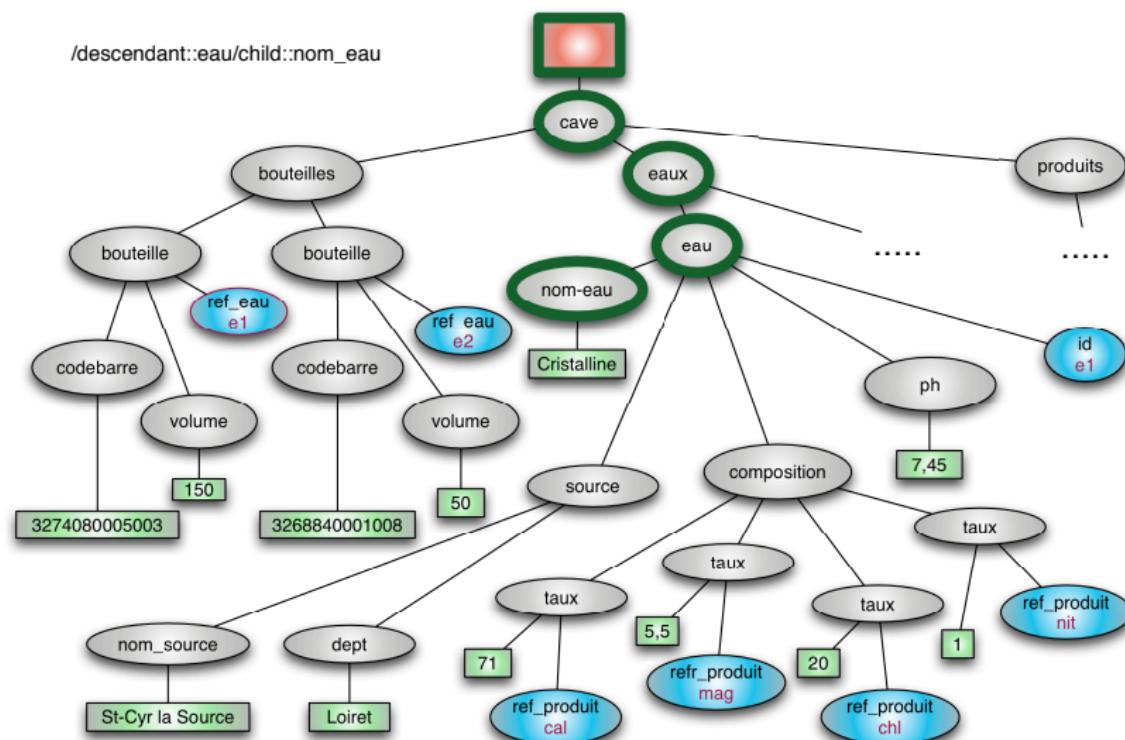
```
/descendant::eau[attribute::id="e3" or attribute::id="e2"]
```

```
//eau/.../.../bouteilles/bouteille/@ref_eau
```

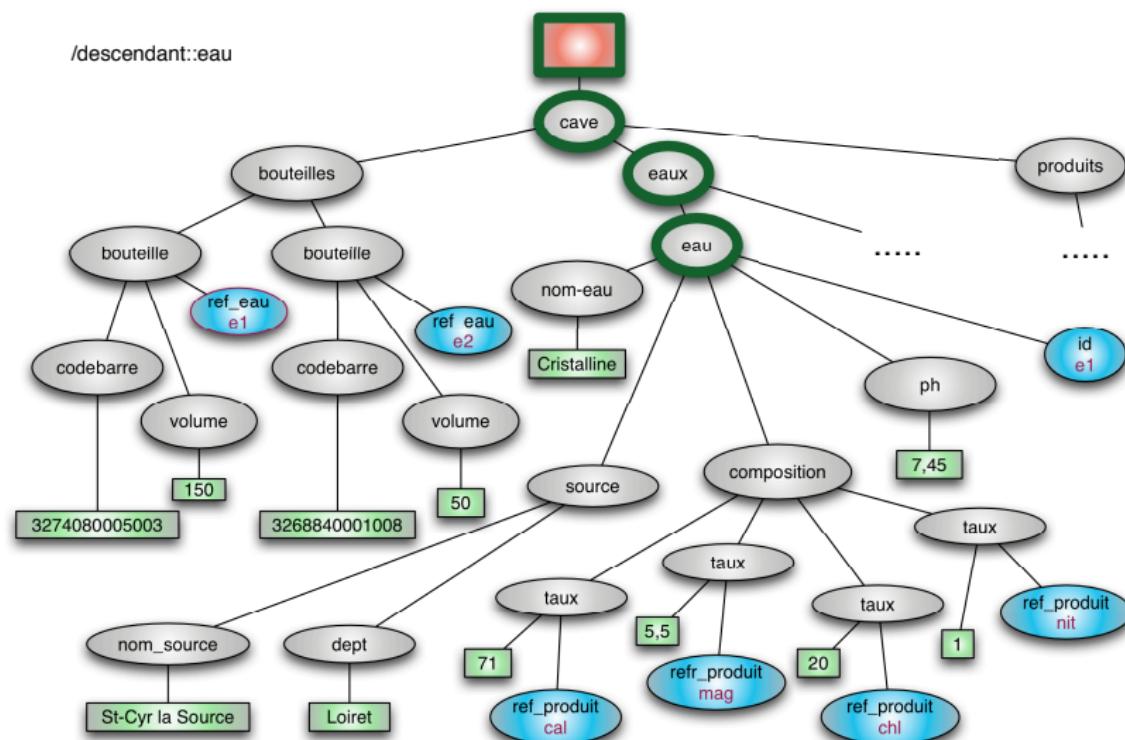
Résultats



Résultats

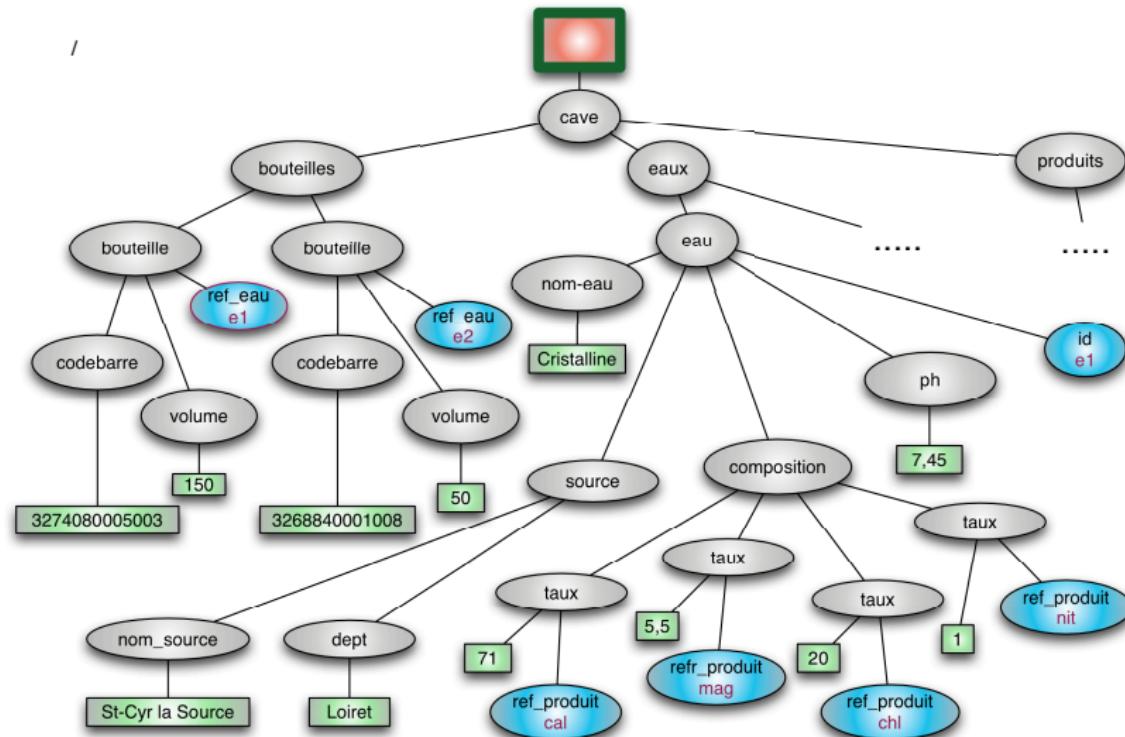


Résultats

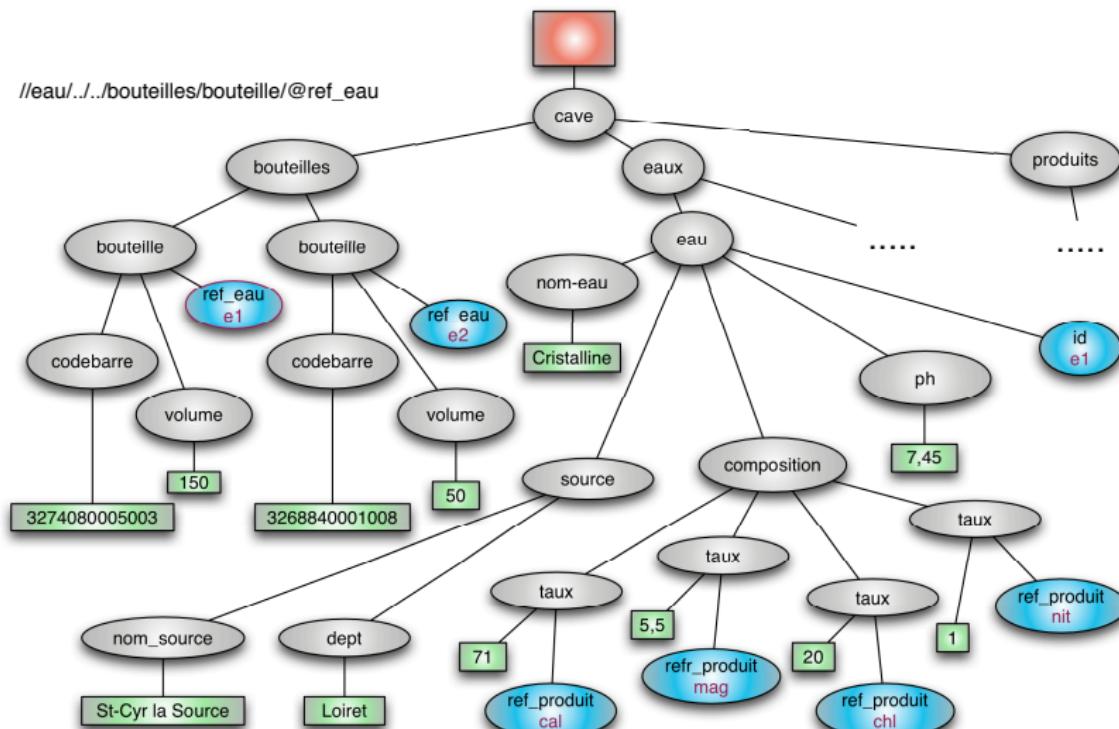


Résultats

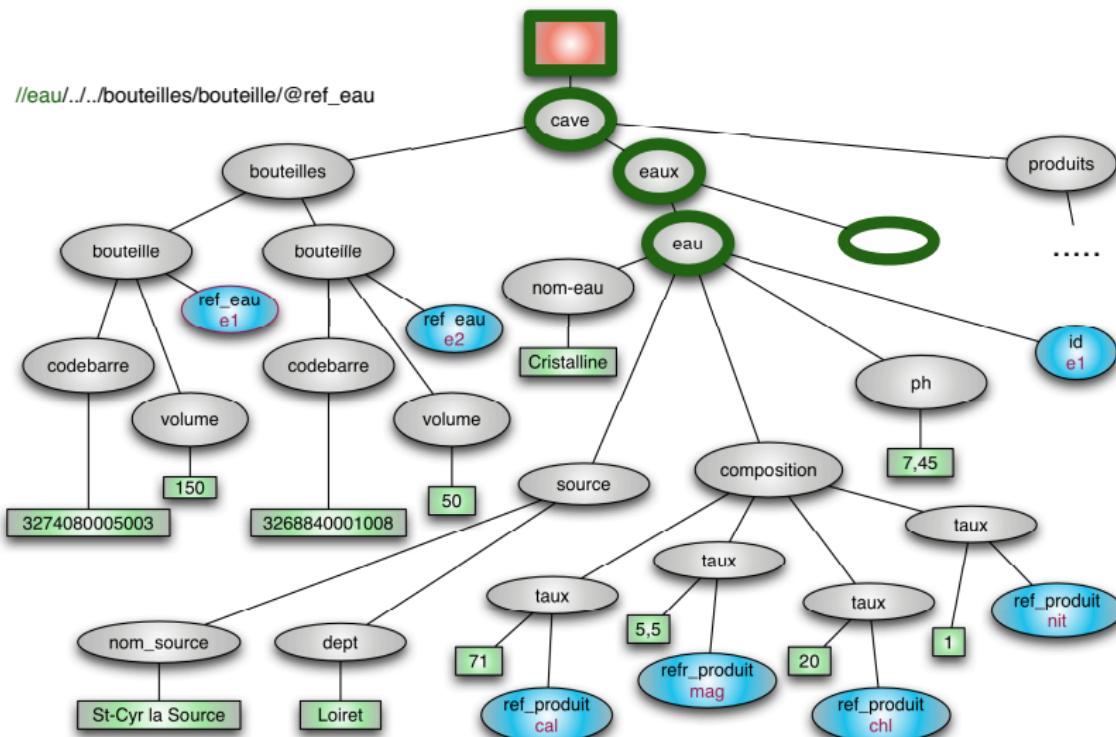
/



Résultats

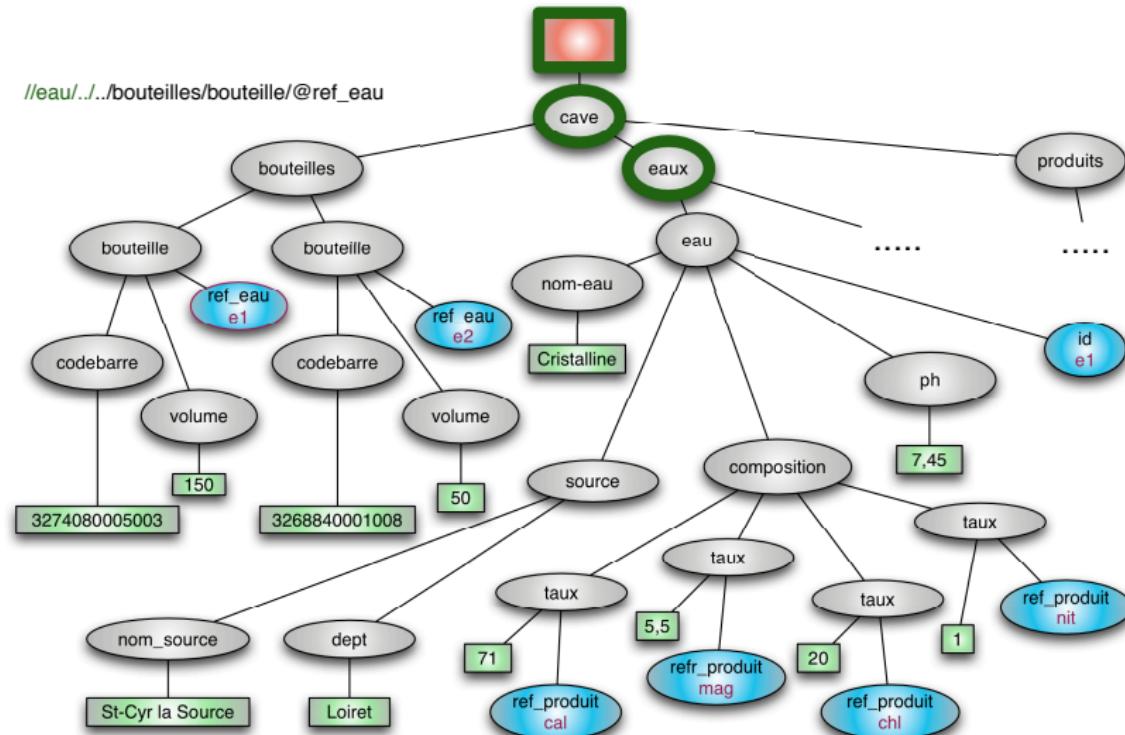


Résultats

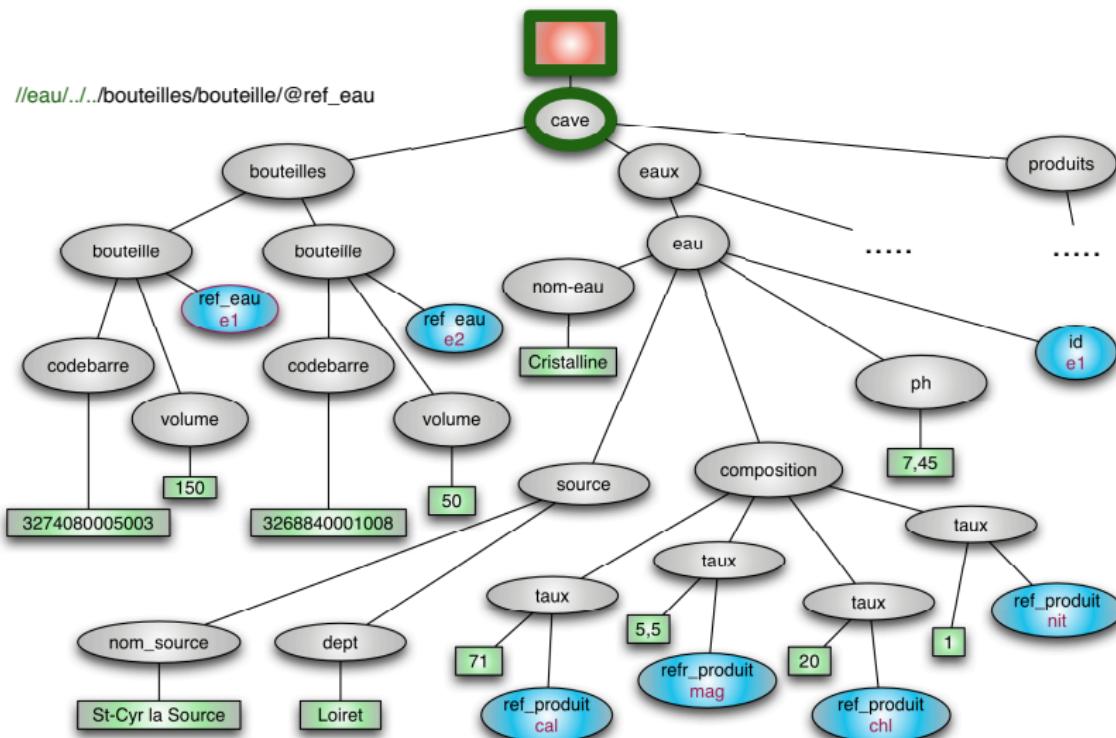


Résultats

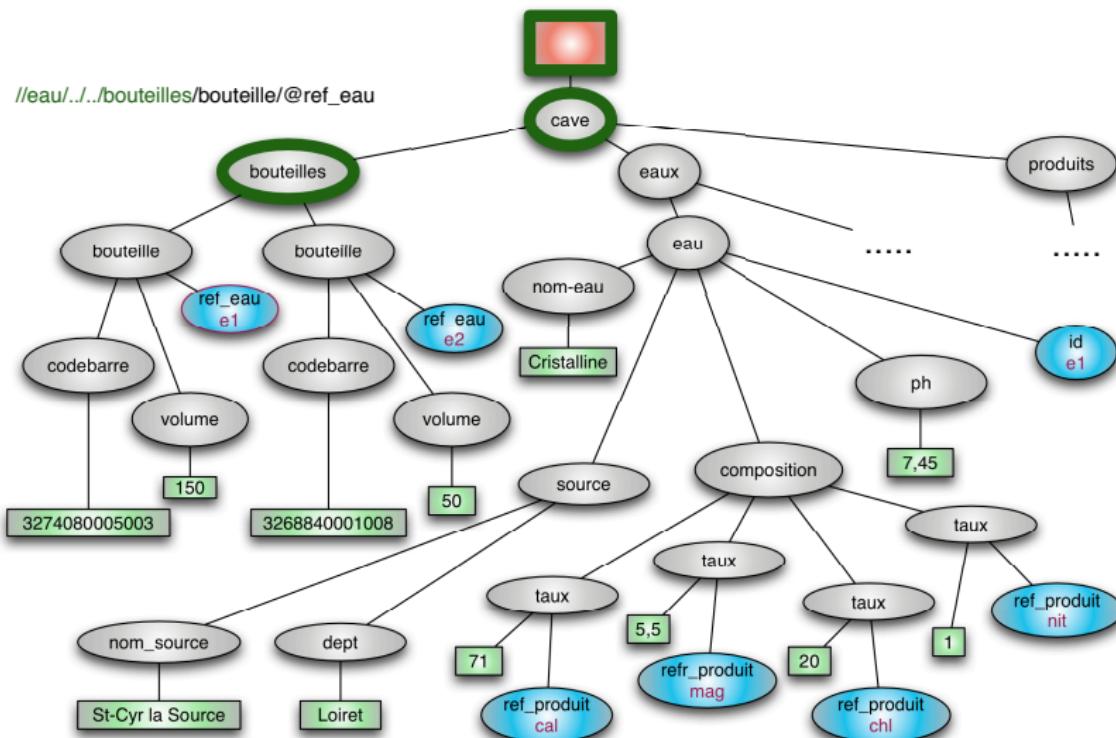
//eau/..../bouteilles/bouteille/@ref_eau



Résultats

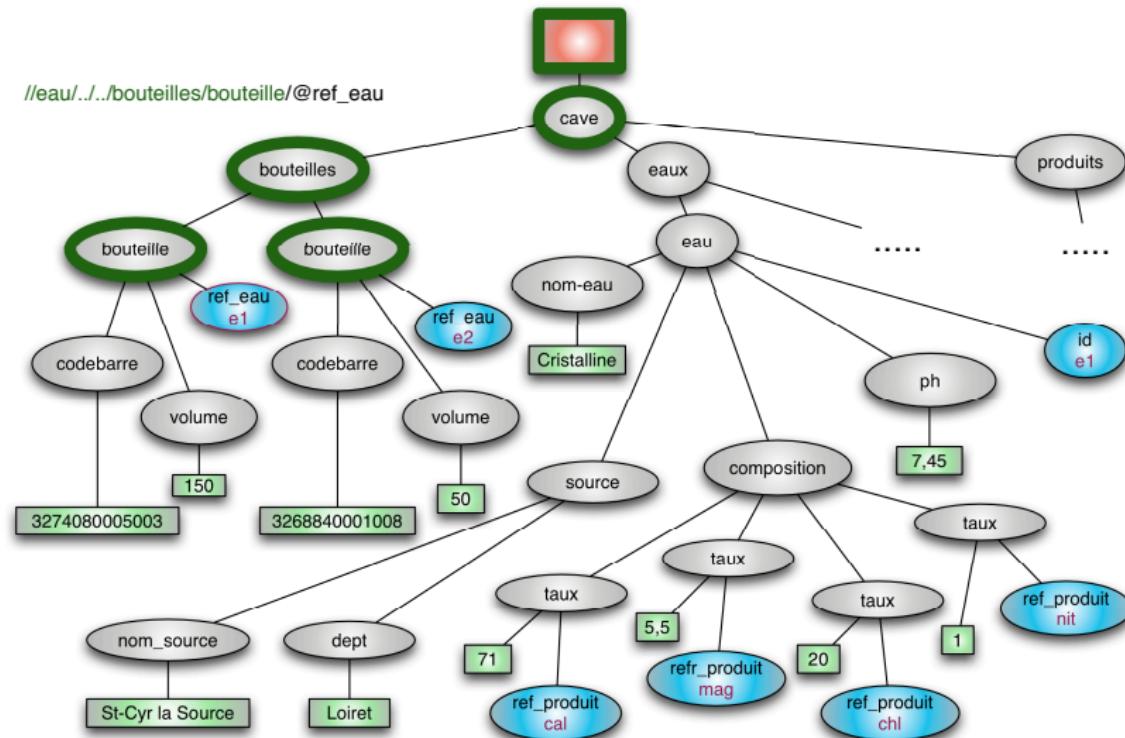


Résultats

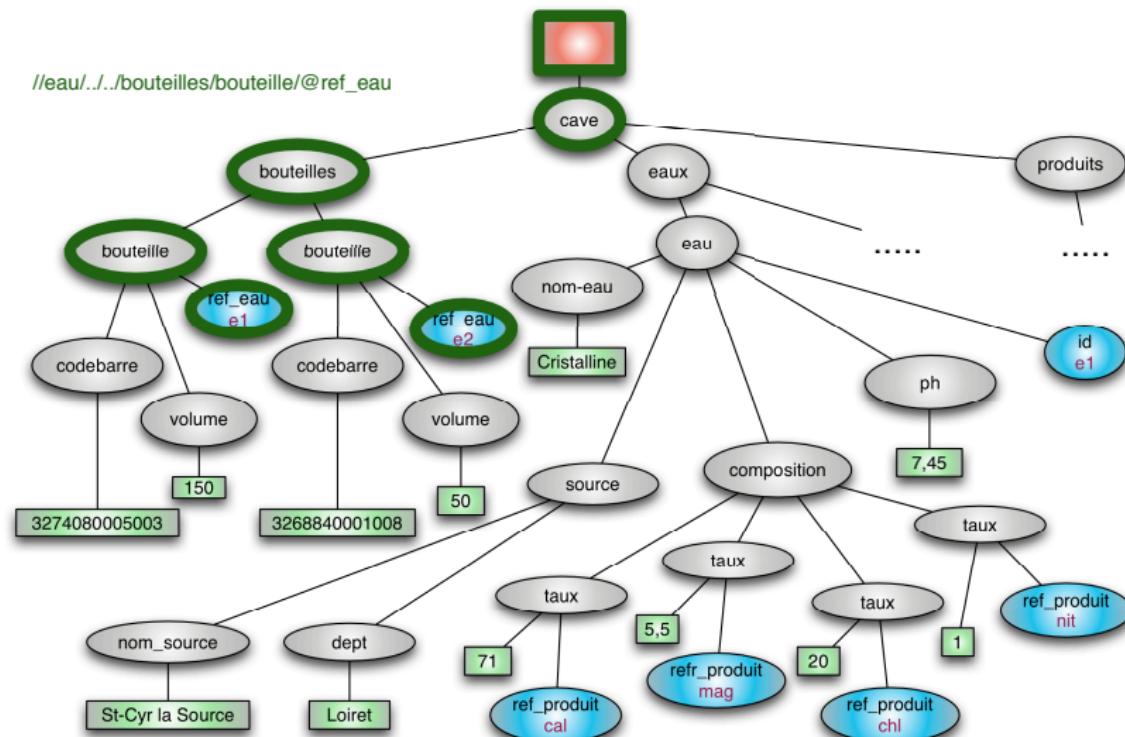


Résultats

//eau/../../bouteilles/bouteille/@ref_eau



Résultats



Syntaxe des expressions

```
PathExpr      ::=  '/' RelativePathExpr? |  
                   '//' ? RelativePathExpr  
RelativePathExpr ::= StepExpr(( '/' | '//' ) StepExpr)*  
StepExpr       ::= AxisStep | FilterExpr  
AxisStep        ::= Axis '::' NodeTest(Predicate)*  
FilterExpr      ::= PrimaryExpr (Predicate)*  
Predicate        ::= '[' Expr ']'
```

Cette syntaxe est la plus "verbeuse", on verra par la suite qu'on peut utiliser des abréviations.

Les Axes (Axis)

child sélectionne les nœuds éléments fils du nœud courant

descendant les nœuds éléments descendants du nœud courant

parent le nœud père du nœud courant

ancestor les éléments ancêtres

following-sibling (preceding-sibling) les éléments frères droits (les éléments frères gauches)

following (preceding) les éléments dont la balise ouvrante (fermante) apparaît après (avant) dans le document

self le nœud courant lui-même

ancestor-or-self, descendant-or-self

attribute les attributs de l'élément courant

namespace les espaces de noms

Les Axes (Axis)

child sélectionne les nœuds éléments fils du nœud courant

descendant les nœuds éléments descendants du nœud courant

parent le nœud père du nœud courant

ancestor les éléments ancêtres

following-sibling (**preceding-sibling**) les éléments frères droits (les éléments frères gauches)

following (**preceding**) les éléments dont la balise ouvrante (fermante) apparaît après (avant) dans le document

self le nœud courant lui-même

ancestor-or-self, **descendant-or-self**

attribute les attributs de l'élément courant

namespace les espaces de noms

Les Axes (Axis)

child sélectionne les nœuds éléments fils du nœud courant

descendant les nœuds éléments descendants du nœud courant

parent le nœud père du nœud courant

ancestor les éléments ancêtres

following-sibling (preceding-sibling) les éléments frères droits (les éléments frères gauches)

following (preceding) les éléments dont la balise ouvrante (fermante) apparaît après (avant) dans le document

self le nœud courant lui-même

ancestor-or-self, descendant-or-self

attribute les attributs de l'élément courant

namespace les espaces de noms

Les Axes (Axis)

child sélectionne les nœuds éléments fils du nœud courant

descendant les nœuds éléments descendants du nœud courant

parent le nœud père du nœud courant

ancestor les éléments ancêtres

following-sibling (preceding-sibling) les éléments frères droits (les éléments frères gauches)

following (preceding) les éléments dont la balise ouvrante (fermante) apparaît après (avant) dans le document

self le nœud courant lui-même

ancestor-or-self, descendant-or-self

attribute les attributs de l'élément courant

namespace les espaces de noms

Les Axes (Axis)

child sélectionne les nœuds éléments fils du nœud courant

descendant les nœuds éléments descendants du nœud courant

parent le nœud père du nœud courant

ancestor les éléments ancêtres

following-sibling (preceding-sibling) les éléments frères droits (les éléments frères gauches)

following (preceding) les éléments dont la balise ouvrante (fermante) apparaît après (avant) dans le document

self le nœud courant lui-même

ancestor-or-self, descendant-or-self

attribute les attributs de l'élément courant

namespace les espaces de noms

Les Axes (Axis)

child sélectionne les nœuds éléments fils du nœud courant

descendant les nœuds éléments descendants du nœud courant

parent le nœud père du nœud courant

ancestor les éléments ancêtres

following-sibling (preceding-sibling) les éléments frères droits (les éléments frères gauches)

following (preceding) les éléments dont la balise ouvrante (fermante) apparaît après (avant) dans le document

self le nœud courant lui-même

ancestor-or-self, descendant-or-self

attribute les attributs de l'élément courant

namespace les espaces de noms

Les Axes (Axis)

child sélectionne les nœuds éléments fils du nœud courant

descendant les nœuds éléments descendants du nœud courant

parent le nœud père du nœud courant

ancestor les éléments ancêtres

following-sibling (*preceding-sibling*) les éléments frères droits (les éléments frères gauches)

following (*preceding*) les éléments dont la balise ouvrante (fermante) apparaît après (avant) dans le document

self le nœud courant lui-même

ancestor-or-self, descendant-or-self

attribute les attributs de l'élément courant

namespace les espaces de noms

Les Axes (Axis)

child sélectionne les nœuds éléments fils du nœud courant

descendant les nœuds éléments descendants du nœud courant

parent le nœud père du nœud courant

ancestor les éléments ancêtres

following-sibling (preceding-sibling) les éléments frères droits (les éléments frères gauches)

following (preceding) les éléments dont la balise ouvrante (fermante) apparaît après (avant) dans le document

self le nœud courant lui-même

ancestor-or-self, descendant-or-self

attribute les attributs de l'élément courant

namespace les espaces de noms

Les Axes (Axis)

child sélectionne les nœuds éléments fils du nœud courant

descendant les nœuds éléments descendants du nœud courant

parent le nœud père du nœud courant

ancestor les éléments ancêtres

following-sibling (*preceding-sibling*) les éléments frères droits (les éléments frères gauches)

following (*preceding*) les éléments dont la balise ouvrante (fermante) apparaît après (avant) dans le document

self le nœud courant lui-même

ancestor-or-self, descendant-or-self

attribute les attributs de l'élément courant

namespace les espaces de noms

Les Axes (Axis)

child sélectionne les nœuds éléments fils du nœud courant

descendant les nœuds éléments descendants du nœud courant

parent le nœud père du nœud courant

ancestor les éléments ancêtres

following-sibling (*preceding-sibling*) les éléments frères droits (les éléments frères gauches)

following (*preceding*) les éléments dont la balise ouvrante (fermante) apparaît après (avant) dans le document

self le nœud courant lui-même

ancestor-or-self, descendant-or-self

attribute les attributs de l'élément courant

namespace les espaces de noms

Les Axes (Axis)

child sélectionne les nœuds éléments fils du nœud courant

descendant les nœuds éléments descendants du nœud courant

parent le nœud père du nœud courant

ancestor les éléments ancêtres

following-sibling (preceding-sibling) les éléments frères droits (les éléments frères gauches)

following (preceding) les éléments dont la balise ouvrante (fermante) apparaît après (avant) dans le document

self le nœud courant lui-même

ancestor-or-self, descendant-or-self

attribute les attributs de l'élément courant

namespace les espaces de noms

Les filtres (NodeTest)

Un filtre peut être :

- Un nom d'élément ou d'attribut
- Le caractère * qui sélectionne tous les objets de même nature (mais uniquement élément ou attribut). child::* sélectionne tous les éléments fils du nœud courant, attribute::* sélectionne tous les attributs du nœud courant.
- comment() qui sélectionne les nœuds de type commentaire
- text() qui sélectionne les nœuds de type texte
- node() qui sélectionne les nœuds de n'importe quel type.
- ...

Le résultat de /child::cave/child::eaux/child::eau

```
<eau id="e1">
    <nom_eau> Cristalline</nom_eau>
    <source>
        <nom_source>St-Cyr la Source</nom_source>
        <dept>Loiret</dept>
    </source>
    <composition>
        <taux ref_produit="cal">71</taux>
        <taux ref_produit="mag">5,5</taux>
        <taux ref_produit="chl">20</taux>
        <taux ref_produit="nit">1</taux>
    </composition>
    <ph>7,45</ph>
</eau>
...
<eau id="e3">
    <nom_eau>Volvic</nom_eau>
...
</eau>
```

Autres exemples

- `/descendant::eau/child::nom_eau/child::text()`
retourne la séquence des noms d'eaux minérale (donc séquence de chaînes de caractères)
- `/descendant::bouteilles/following-sibling::*`
retourne les éléments frères à droite d'un élément bouteilles, donc la séquence formée des éléments eaux et produits.
- `/descendant::*` retourne la séquence de tous les éléments du document

Les prédictats

Un prédictat est une expression qui peut être évaluée à vrai ou faux. en fait, n'importe quelle expression XPath convient (variable Expr de la grammaire, encore plus générale que PathExpr) Voici comment convertir une expression en une valeur booléenne :

- Une séquence vide vaut faux
- Une séquence dont le premier item est un nœud vaut vrai
- Un singleton qui contient une valeur booléenne vaut cette valeur.
- Un singleton qui contient une chaîne de caractère vaut vraissi cette chaîne est non vide
- Un singleton qui contient un nombre vaut vrai pour un nombre différent de 0, faux sinon.

Les prédictats (suite)

- Dans une expression XPath, Axis::NodeTest donne comme résultat une séquence de nœuds. Le prédictat qui suit s'applique sur chacun des nœuds de la séquence. Pour chaque nœud, si le prédictat vaut vrai, le nœud est conservé, sinon le nœud est rejeté.
- On peut enchaîner des prédictats en les juxtaposant : [p1]...[pk].
- Un prédictat XPath peut être composé d'autres prédictats grâce aux opérateurs `and` et `or`.
- Un prédictat peut être une égalité ou une inégalité définie à l'aide des opérateurs = < > <= >=
- Les prédictats peuvent utiliser une fonction portant sur les nœuds.

Quelques fonctions

- `id("valeur")` : retourne l'élément dont l'identifiant vaut "valeur". On peut passer une séquence de valeurs en paramètre, le résultat est la séquence des éléments correspondants, sans doublons.
- `position()` : retourne la position du nœud dans la séquence. La première position vaut 1, la dernière vaut `last()`.
- `count(séquence-de-nœuds)` Retourne le nombre de nœuds présents dans la séquence
- `not(predicat)` renvoie vrai ssi predicat a la valeur faux.
- manipulation de chaînes de caractères : `contains`, `concat` ...
- fonctions numériques : somme, valeur arrondie, ...

Exemples

- `/descendant::*[child::taux]` la séquence des éléments qui ont un fils taux.
- `/descendant::eau[attribute::id="e3"]` l'élément eau dont l'attribut id vaut "e3"
- `/descendant::eau[attribute::id="e3" or attribute::id="e2"]` la séquence des éléments eau dont l'attribut id vaut "e3" ou "e2"
- `/descendant::taux` la séquence de tous les taux
- `/descendant::taux[position()=2]` le deuxième élément taux **de la séquence précédente.**
- `/descendant::* /child::taux[position()=2]` la séquence des taux qui sont deuxièmes fils (c'est la liste des taux de magnésium)

Syntaxe abrégée

syntaxe inspirée des systèmes de fichiers.

- child:: peut être omis
- attribute:: peut être remplacé par @
- /descendant-or-self::node() peut être remplacé par //
Attention, // est évalué à partir de la racine.
- self::node() peut être remplacé par un point.
- parent::node() peut être remplacé par ..
- [position()=x] peut s'écrire [x]

Syntaxe abrégée

syntaxe inspirée des systèmes de fichiers.

- child:: peut être omis
- attribute:: peut être remplacé par @
- /descendant-or-self::node() peut être remplacé par //
Attention, // est évalué à partir de la racine.
- self::node() peut être remplacé par un point.
- parent::node() peut être remplacé par ..
- [position()=x] peut s'écrire [x]

Exemples

- `//eau/..` sélectionne l'élément eaux.
- `//eau/@id` sélectionne tous les attributs id des éléments eau.
- `//eau/.../bouteilles/bouteille/@ref_eau`
sélectionne tous les attributs ref_eau des éléments bouteille.
- `//bouteilles/bouteille[2]` sélectionne le deuxième fils bouteille de l'élément bouteilles.
- `//bouteille[@ref_eau="e1"]` sélectionne les bouteilles dont l'attribut ref_eau vaut "e1".
- `//bouteille[../../@ref_eau="e1"]` même résultat que la requête précédente
- `//bouteille[//@ref_eau="e1"]` sélectionne tous les éléments bouteille car `//@ref_eau` est évalué à partir de la racine

Un dernier point sur les séquences

- 1 Construction d'une séquence : Un moyen de construire une séquence est d'utiliser la virgule comme opérateur de concaténation. L'opérateur **to** permet de définir un intervalle.
 - (10, 1, 2, 3, 4) séquence de 5 entiers
 - (10, (1, 2), (), (3, 4)) même séquence que précédemment
 - (10, 1 to 4) toujours la même séquence
- 2 Expressions de chemin
 - //bouteilles/bouteille/(codebarre,volume) séquence avec tous les fils codebarre et volume des éléments bouteille.
 - (21 to 29)[5] l'entier 25.
 - //bouteille[position() == (1 to 2)] le premier et le second élément bouteille du document.
- 3 Opérateurs sur les séquences : union, intersect, except.
- 4 Comparaison de deux séquences : quantification existentielle
 - (1,2) = (2,3) vaut vrai (Donc l'égalité n'est pas transitive !)
 - (2,3) = (3,4) vaut vrai, pourtant (1,2) = (3,4) vaut faux.)