EXEMPLE XML SCHEMA

```
<?xml version="1.0" encoding="iso-8859-1"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:annotation>
    <xsd:documentation xml:lang="fr">
      Schéma XML pour bibliography.xml
    </xsd:documentation>
  </xsd:annotation>
  <xsd:element name="bibliography" type="Bibliography"/>

  <xsd:complexType name="Bibliography">
    <xsd:sequence>
      <xsd:element name="book" minOccurs="1" maxOccurs="unbounded">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="title"     type="xsd:string"/>
            <xsd:element name="author"    type="xsd:string"/>
            <xsd:element name="year"      type="xsd:string"/>
            <xsd:element name="publisher" type="xsd:string"/>
            <xsd:element name="isbn"      type="xsd:string"/>
            <xsd:element name="url"       type="xsd:string" minOccurs="0"/>
          </xsd:sequence>
          <xsd:attribute name="key"  type="xsd:NMTOKEN" use="required"/>
          <xsd:attribute name="lang" type="xsd:NMTOKEN" use="required"/>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

Élément racine xsd:schema avec la déclaration de l'espace de noms des schémas associé au préfixe xsd.

EXEMPLE XML SCHEMA

```
<?xml version="1.0" encoding="iso-8859-1"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 <xsd:annotation>
  <xsd:documentation xml:lang="fr">
   Schéma XML pour bibliography.xml
  </xsd:documentation>
 </xsd:annotation>
 <xsd:element name="bibliography" type="Bibliography"/>

 <xsd:complexType name="Bibliography">
  <xsd:sequence>
   <xsd:element name="book" minOccurs="1" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="title"     type="xsd:string"/>
      <xsd:element name="author"    type="xsd:string"/>
      <xsd:element name="year"      type="xsd:string"/>
      <xsd:element name="publisher" type="xsd:string"/>
      <xsd:element name="isbn"      type="xsd:string"/>
      <xsd:element name="url"       type="xsd:string" minOccurs="0"/>
     </xsd:sequence>
     <xsd:attribute name="key"  type="xsd:NMTOKEN" use="required"/>
     <xsd:attribute name="lang" type="xsd:NMTOKEN" use="required"/>
    </xsd:complexType>
   </xsd:element>
  </xsd:sequence>
 </xsd:complexType>
</xsd:schema>
```

Documentation du schéma

EXEMPLE XML SCHEMA

```
<?xml version="1.0" encoding="iso-8859-1"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 <xsd:annotation>
  <xsd:documentation xml:lang="fr">
   Schéma XML pour bibliography.xml
  </xsd:documentation>
 </xsd:annotation>
 <xsd:element name="bibliography" type="Bibliography"/>

 <xsd:complexType name="Bibliography">
  <xsd:sequence>
   <xsd:element name="book" minOccurs="1" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="title"     type="xsd:string"/>
      <xsd:element name="author"    type="xsd:string"/>
      <xsd:element name="year"      type="xsd:string"/>
      <xsd:element name="publisher" type="xsd:string"/>
      <xsd:element name="isbn"      type="xsd:string"/>
      <xsd:element name="url"       type="xsd:string" minOccurs="0"/>
     </xsd:sequence>
     <xsd:attribute name="key"  type="xsd:NMTOKEN" use="required"/>
     <xsd:attribute name="lang" type="xsd:NMTOKEN" use="required"/>
    </xsd:complexType>
   </xsd:element>
  </xsd:sequence>
 </xsd:complexType>
</xsd:schema>
```

Déclaration de l'élément bibliography avec le type Bibliography

EXEMPLE XML SCHEMA

```
<?xml version="1.0" encoding="iso-8859-1"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 <xsd:annotation>
  <xsd:documentation xml:lang="fr">
   Schéma XML pour bibliography.xml
  </xsd:documentation>
 </xsd:annotation>
 <xsd:element name="bibliography" type="Bibliography"/>

 <xsd:complexType name="Bibliography">
  <xsd:sequence>
   <xsd:element name="book" minOccurs="1" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="title"     type="xsd:string"/>
      <xsd:element name="author"    type="xsd:string"/>
      <xsd:element name="year"      type="xsd:string"/>
      <xsd:element name="publisher" type="xsd:string"/>
      <xsd:element name="isbn"      type="xsd:string"/>
      <xsd:element name="url"       type="xsd:string" minOccurs="0"/>
     </xsd:sequence>
     <xsd:attribute name="key"  type="xsd:NMTOKEN" use="required"/>
     <xsd:attribute name="lang" type="xsd:NMTOKEN" use="required"/>
    </xsd:complexType>
   </xsd:element>
  </xsd:sequence>
 </xsd:complexType>
</xsd:schema>
```

Début de la définition du type Bibliography

EXEMPLE XML SCHEMA

```
<?xml version="1.0" encoding="iso-8859-1"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 <xsd:annotation>
  <xsd:documentation xml:lang="fr">
   Schéma XML pour bibliography.xml
  </xsd:documentation>
 </xsd:annotation>
 <xsd:element name="bibliography" type="Bibliography"/>

 <xsd:complexType name="Bibliography">
  <xsd:sequence>
   <xsd:element name="book" minOccurs="1" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="title"     type="xsd:string"/>
      <xsd:element name="author"    type="xsd:string"/>
      <xsd:element name="year"      type="xsd:string"/>
      <xsd:element name="publisher" type="xsd:string"/>
      <xsd:element name="isbn"      type="xsd:string"/>
      <xsd:element name="url"       type="xsd:string" minOccurs="0"/>
     </xsd:sequence>
     <xsd:attribute name="key"  type="xsd:NMTOKEN" use="required"/>
     <xsd:attribute name="lang" type="xsd:NMTOKEN" use="required"/>
    </xsd:complexType>
   </xsd:element>
  </xsd:sequence>
 </xsd:complexType>
</xsd:schema>
```

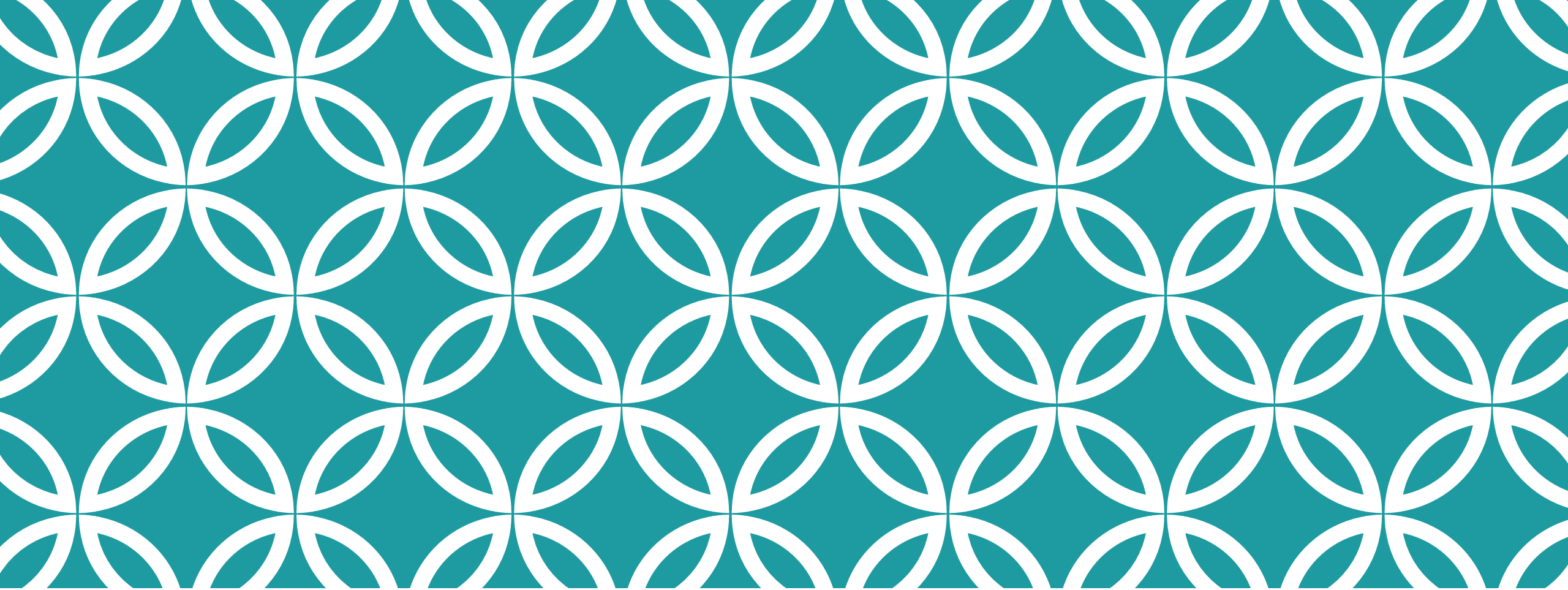
Déclaration du l'élément book dans le contenu du type Bibliography

EXEMPLE XML SCHEMA

```
<?xml version="1.0" encoding="iso-8859-1"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 <xsd:annotation>
  <xsd:documentation xml:lang="fr">
   Schéma XML pour bibliography.xml
  </xsd:documentation>
 </xsd:annotation>
 <xsd:element name="bibliography" type="Bibliography"/>

 <xsd:complexType name="Bibliography">
  <xsd:sequence>
   <xsd:element name="book" minOccurs="1" maxOccurs="unbounded">
    <xsd:complexType>
     <xsd:sequence>
      <xsd:element name="title"     type="xsd:string"/>
      <xsd:element name="author"    type="xsd:string"/>
      <xsd:element name="year"      type="xsd:string"/>
      <xsd:element name="publisher" type="xsd:string"/>
      <xsd:element name="isbn"      type="xsd:string"/>
      <xsd:element name="url"       type="xsd:string" minOccurs="0"/>
     </xsd:sequence>
     <xsd:attribute name="key"  type="xsd:NMTOKEN" use="required"/>
     <xsd:attribute name="lang" type="xsd:NMTOKEN" use="required"/>
    </xsd:complexType>
   </xsd:element>
  </xsd:sequence>
 </xsd:complexType>
</xsd:schema>
```

Déclaration des attributs key et lang de l'élément book avec le type xsd:NMTOKEN



LANGAGES DE REQUÊTES

- XPath
- XSLT
- XQuery

XPATH

- XPath: l'adressage XML
  - ✓ permet de définir la manière pour adresser un ou plusieurs nœuds d'un document XML (expression de chemins d'accès)
    - un chemin d'accès ressemble un peu à celui des noms de fichiers (chemins), d'où le nom "XPath"
  - ✓ traiter des chaînes de caractères, des nombres et des booléens
- XML Path Language
  - ✓ recommandation W3C
  - ✓ version 2 disponible
- Expressions de chemins communes à :
  - ✓ XSLT, XLink/XPointer (liens), XQuery (requêtes), XML Schema (clés/références)
- XPath permet donc
  - ✓ de naviguer dans un arbre XML
  - ✓ de rechercher un ou plusieurs éléments dans un document
  - ✓ de référencer tout fragment d'un document

XPATH (2)

- XPath opère sur une représentation arborescente du document
- Chaque élément de l'arbre est appelé un nœud (*node*)
- Une expression de chemin spécifie une traversée de l'arbre du document :
  - ✓ depuis un nœud de départ (appelé également nœud contexte)
  - ✓ vers un ensemble de nœuds cibles (triés dans l'ordre du document)
  - ✓ les cibles constituent la valeur du cheminement
- Notion de contexte
  - ✓ pour comprendre ce que fait une expression XPath, il faut toujours faire attention à son contexte d'utilisation
- Un chemin est une séquence d'étapes et peut être :
  - ✓ absolu (le nœud contexte est la racine)
    - commence à la racine: /étape1/…/étapeN
  - ✓ relatif (le nœud contexte est un nœud du document)
    - commence à un nœud courant: étape1/…/étapeN

SYNTAXE ET SÉMANTIQUE

- Une étape est de la forme: [axe::]filtre[prédicat]*
  - ✓ l'axe définit la relation entre les nœuds et le sens de parcours – optionnel
  - ✓ le filtre sélectionne un type de nœud (élément ou @attribut)
  - ✓ les prédicats doivent être satisfaits par les nœuds retenus – optionnel
- Exemples:
  - ✓ child::para
  - ✓ child::figure[attribute::id="fr vesca"]
  - ✓ child::*[position()=last()]
- Syntaxe simplifiée
  - ✓ @name équivalent à attribute::name
  - ✓ para[1] équivalent à child::para[position()=1]
  - ✓ .//para équivalent à self::node()/descendant-or-self::node()/child::para
  - ✓ . équivalent à self::node()
  - ✓ ../para équivalent à parent::node()/child::para

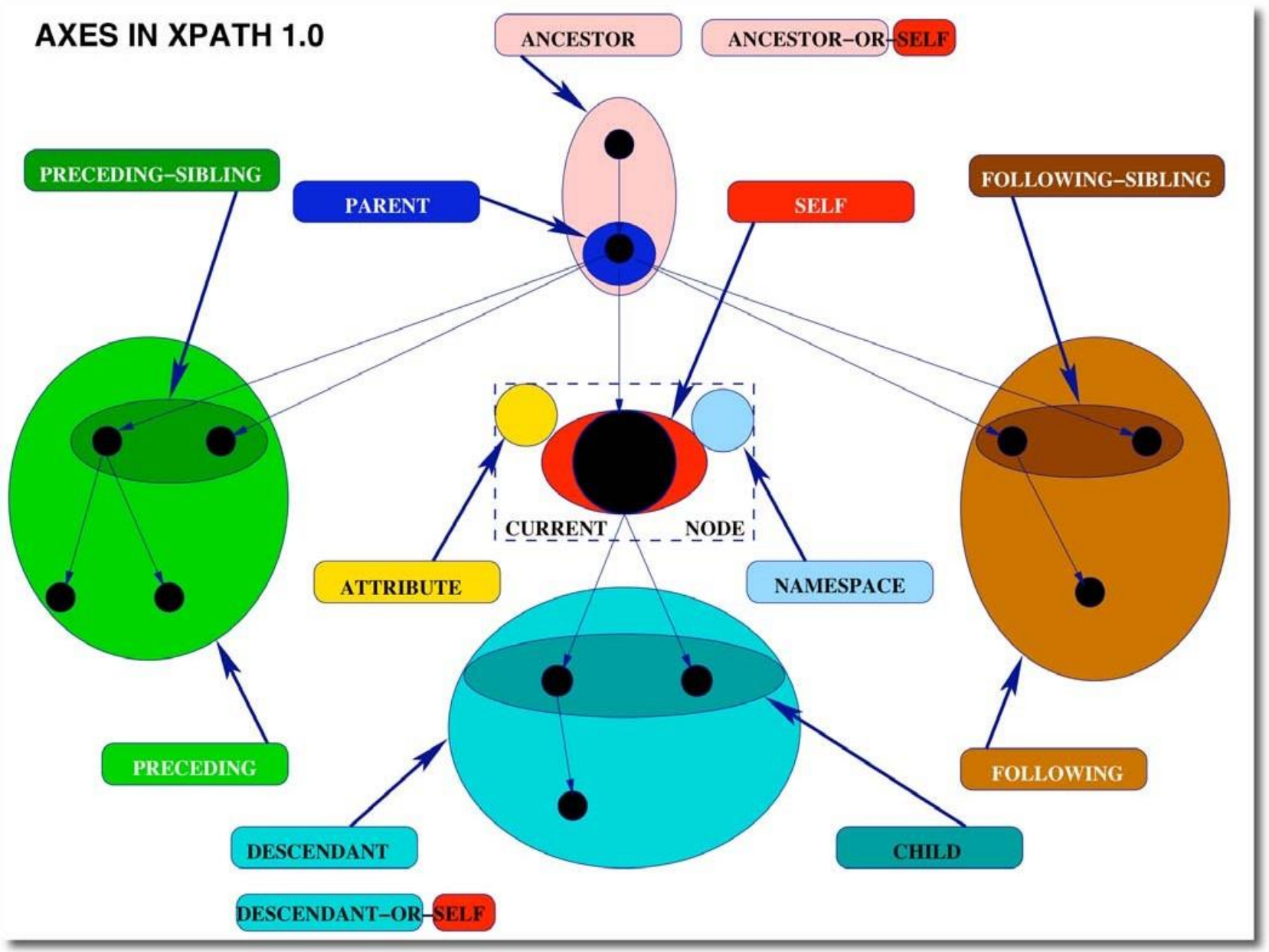
LE CONTEXTE

- Le contexte d'une évaluation XPath consiste
  - ✓ un nœud contexte (un nœud de l'arbre)
  - ✓ une position et une taille
  - ✓ un ensemble de variables associées
  - ✓ une librairie de fonctions
  - ✓ la déclaration d'un ensemble de namespace
- L'application détermine le contexte initial
- Si le chemin commence à la racine ("/")
  - ✓ le contexte initial est le nœud racine
  - ✓ la position est 1, et la taille est 1

LES AXES

- XPath supporte 12 axes
- Les axes en avant
  - ✓ child, self, descendant, descendant-or-self
- Les axes en arrière
  - ✓ parent, ancestor, ancestor-or-self
- Les axes à gauche et à droite
  - ✓ following-sibling, preceding-sibling
  - ✓ Les axes avant et après
  - ✓ following, preceding
- Les axes pour les attributs
  - ✓ attribute

LES AXES (2


AXES IN XPATH 1.0
ANCESTOR
ANCESTOR–OR–SELF
PRECEDING–SIBLING
PARENT
SELF
FOLLOWING–SIBLING
CURRENT
NODE
ATTRIBUTE
NAMESPACE
PRECEDING
FOLLOWING
DESCENDANT
CHILD
DESCENDANT–OR–SELF

LES FILTRES

- Filtrer les nœuds
  - ✓ **nom**: les nœuds de l'axe qui portent ce nom
  - ✓ ***: les nœuds de type Element ou Attribute de l'axe
- Filtrer les nœuds textuels
  - ✓ **text()**: tous les nœuds de type Text de l'axe
- Filtrer les commentaires
  - ✓ **comment()**: tous les nœuds de type Comment de l'axe
- Filtrer les instructions de traitement
  - ✓ **processing-instruction()**: tous les nœuds de type instruction de traitement de l'axe
- Filtrer les nœuds
  - ✓ **node()**: tous les types de nœud (sauf les attributs) de l'axe sauf la racine
  - ✓ **id(label)**: le nœud repéré par une étiquette

LES PRÉDICATS

- Expression logique vraie ou fausse qui affine le résultat obtenu avec le chemin de recherche
- Condition d'existence
  - ✓ expression XPath: vraie si l'expression retourne un ensemble non vide de nœuds
  - ✓ chercher un élément qui a un attribut:
    - nom_element_XML [@nom_attribut]
  - ✓ chercher un élément qui a un attribut avec une certaine valeur:
    - nom_element_XML [@nom_attribut = 'valeur']
- Condition de position
  - ✓ numéro (par ex. [1]): vrai si le nœud courant est à cette position dans le contexte courant
- Expressions booléennes: and, or, not(), true(), false()

LES PRÉDICATS (2)

- Comparaisons
  - ✓ valeurs générales: <=, <, >=, >, =, !=
    - vrai si une paire de valeurs satisfait la comparaison (8 = 4+4, (1,2) = (2,4))
    - $livre/auteur = "Kennedy" est vraie si $livre possède un ou plusieurs auteurs et qu'au moins un est Kennedy
  - ✓ valeurs atomiques: eq, ne, lt, le, gt, ge
    - comparaison stricte des valeurs atomiques (8 eq 4+4)
    - $livre/auteur eq "Kennedy" est vraie ssi $livre possède exactement un auteur et qu'il s'agit de Kennedy
  - ✓ nœuds: is, <<, >>
    - l'opérateur is compare deux nœuds (s'agit-il du même nœud ?)
    - les opérateurs << et >> comparent l'ordre des nœuds dans le document
    - $livre/auteur is key('auteurs', 'kennedy') est vraie ssi $livre possède exactement un auteur et que cet élément auteur est le même que celui retourné par l'expression key

FONCTIONS XPATH

- XPath définit un certain nombre de fonctions (106 !)
- Chaque fonction retourne soit une valeur booléenne (vrai/faux), un nombre, une chaîne de caractères, ou encore une liste de noeuds

- Fonctions pour les ensembles de noeuds
  - ✓ number **last**(): retourne le nombre de noeuds qui se trouvent dans le contexte (qui ont le même parent)
  - ✓ number **position**(): retourne le nombre de la position contextuelle (context position) d'un élément par rapport à son parent
  - ✓ number: **count**(node-set): retourne le nombre de noeuds de l'ensemble de noeuds passés en arguments
  - ✓ node-set **id**(object): retourne les éléments à partir de leur ID unique
  - ✓ string **name**(node-set?), string **local-name**(node-set?), string **namespace-uri** (node-set?): retourne le nom/le nom local/l'espace de nom du nœud en paramètre

FONCTIONS XPATH (2)

- Fonctions sur les chaînes
  - ✓ boolean **starts-with**(string, string): retourne TRUE si le deuxième string se trouve au début du premier
  - ✓ boolean **contains**(string, string): retourne TRUE si le deuxième string se trouve dans le premier
  - ✓ number **string-length**(string?): retourne la longueur d'un string
  - ✓ string **string**(object ?): retourne la version chaîne de l'objet
  - ✓ string **concat**(string, string, string*): retourne la concaténation des deux chaînes
  - ✓ string **substring-before**(string, string): retourne la chaîne res tq ch1 = res + ch2
  - ✓ string **substring-after**(string, string): retourne la chaîne res tq ch1 = ch2 + res
  - ✓ string **substring**(string, number, number ?): retourne l'extraction de la sous- chaîne
  - ✓ string **normalize-space**(string ?): retourne une version normalisée (sans espace)
  - ✓ string **translate**(string, string, string): retourne une chaîne construite à partir de ch1 dans laquelle les caractères présents dans ch2 sont remplacés par les caractères de même position dans ch3

FONCTIONS XPATH (3)

- Fonctions booléennes
  - ✓ boolean boolean(object): teste si l'objet vaut True
  - ✓ boolean not(boolean): vraie si le paramètre est faux
  - ✓ boolean true()
  - ✓ boolean false()
- Fonctions numériques
  - ✓ number **floor**(number): arrondi à l'entier inférieur
  - ✓ number **ceiling**(number): arrondi à l'entier supérieur
  - ✓ number **number**(object?): transforme un objet en nombre
  - ✓ number **sum**(node-set): la somme de nombres trouvés dans un ensemble de noeuds. Effectue une conversion de strings si nécessaire, comme number()
  - ✓ number **round**(number): arrondit un nombre selon les conventions habituelles:
    1.4 devient 1 et 1.7 devient 2
- Expressions: calculs arithmétiques
  - ✓ on utilise: + - * div mod

QUELQUES CHEMINS SIMPLES: ÉLÉMENTS ENFANTS, PARENTS, COUSINS

- Noeud racine:

  / retourne le premier noeud trouvé dans un arbre
- Elément enfant direct:

  nom_element
- Elément enfant direct du noeud racine:

  /nom_element_enfant
- Enfant d'un enfant:

  nom_element_pere/nom_element_enfant
- Descendant arbitraire du noeud racine:

  //nom_element_descendant
- Descendant arbitraire d'un noeud:

  nom_element_ancetre//nom_element_descendant
- Un parent d'un noeud:

  ../
- Un cousin lointain d'un noeud:

  ../../nom_element_XML/nom_element_XML/nom_element_XML