Techno du WEB

Langage XPath

Bernd Amann
(Lylia Abrouk)

Laboratoire Le2i
Université de Bourgogne

9 septembre 2019

Plan

- Arbre XML, ordre, valeur
- Interrogation d'un arbre : le langage XPath

Arbres XML

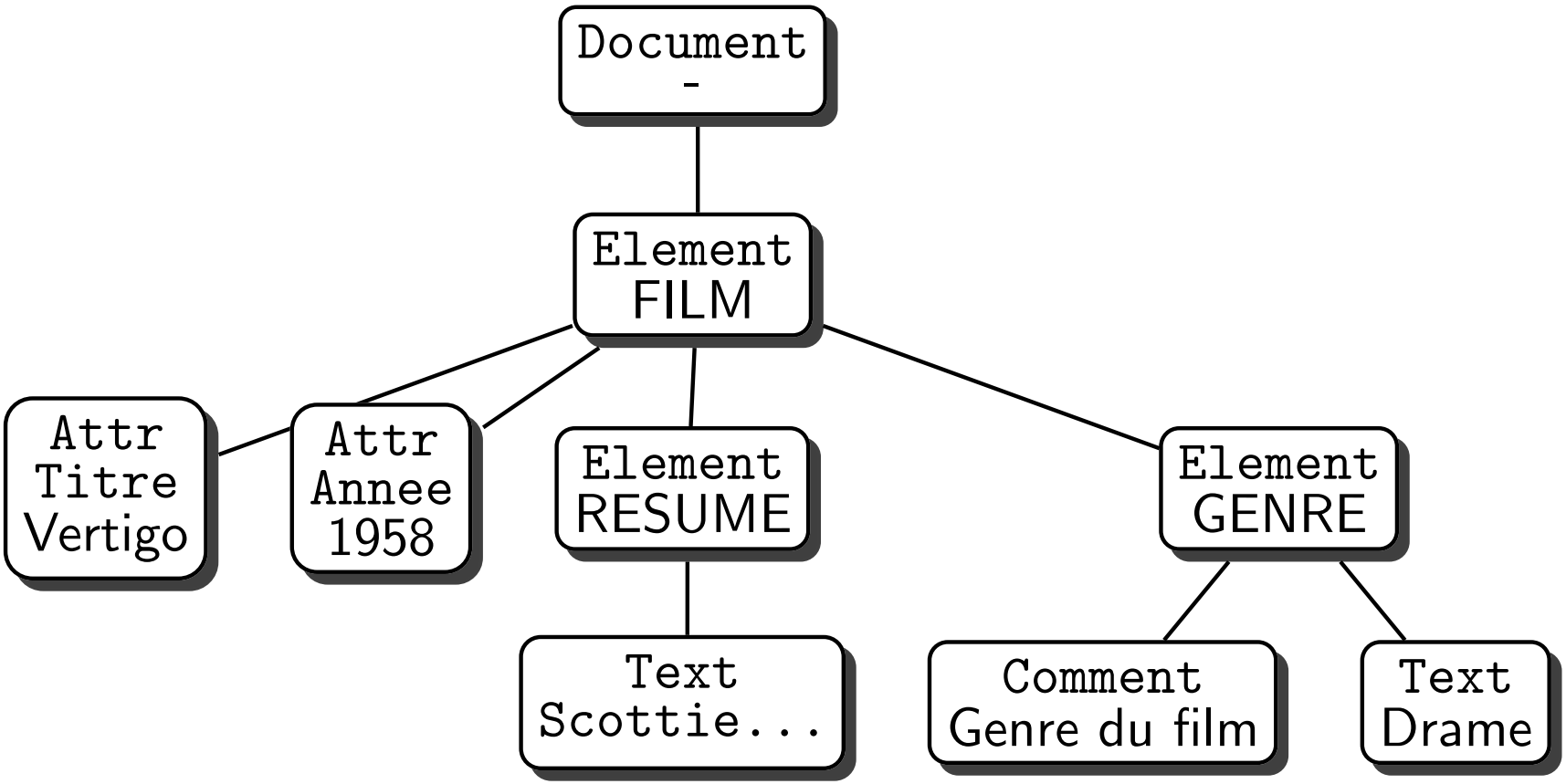
- Arbre d'un document : 7 types de noeuds
  - Document (document XML)
  - Element (élément XML)
  - Attr (attribut XML)
  - Text (valeurs textuelles)
  - Comment (commentaire)
  - ProcessingInstruction (instruction de traitement)
  - Instructions destinées à un processeur
    - ★ Commencent par <?
  - Namespace (espace de noms)
    - ★ Permet d'éviter les conflits de noms

Exemple

```
<? xml version= ‘‘1.0’’ ?>

<FILM titre=‘Vertigo’ annee=‘1958’>
<RESUME> Scottie...</RESUME>
<GENRE>
<!-- genre du film -->
Drame
</GENRE>
</FILM>
```

Exemple


Document
-
Element
FILM
Attr
Titre
Vertigo
Attr
Annee
1958
Element
RESUME
Element
GENRE
Text
Scottie...
Comment
Genre du film
Text
Drame

Ordre du document

- L'ensemble des noeuds de l'arbre d'un document est muni d'un ordre : l'ordre du document qui est l'ordre de lecture, dans le document XML, des constituants représentés par chaque noeud

Elément, contenu, valeur

- Un document est un arbre d'éléments
- Ne pas confondre le contenu de l'élément
  - Ce qui est délimité par ses balises, càd un fragment XML (ou un sous-arbre)
- Avec sa valeur

Valeur textuelle

- Chaque noeud a une valeur textuelle :
  - la valeur textuelle du noeud racine/d'un noeud élément est la concaténation des valeurs textuelles de ses descendants de type texte, dans l'ordre du document,
  - la valeur textuelle d'un noeud texte est la chaîne de caractères constituant ce texte,
  - la valeur textuelle d'un noeud attribut est la chaîne de caractères constituant la valeur de cet attribut

XPath

`http://www.w3.org/TR/xpath.html`

- Spécification W3C
- Expression de chemins XML standards
- Prédicats pour spécifier les valeurs d'éléments ou d'attributs
- Brique de base pour d'autres standards XML
  - XLink : spécification des hyperliens
  - XPointer : pointer des éléments de documents avec des expressions XPath dans les URL
  - XSLT : langage de transformation
  - XQuery : langage de requêtes ($\approx$ SQL pour XML)

XPath, langage de navigation

- XPath permet de naviguer
  - Dans l'arbre du document
  - A partir d'un noeud origine
  - Vers un ou plusieurs noeuds destinations
  - En sélectionnant des chemins dans l'arbre

Navigation

- Modèle de données XML : graphe dont les noeuds sont des éléments
- Expression de chemins
  - séquence de noeuds $T_1, T_2, \ldots, T_n$
  - retourne un ou plusieurs noeuds T_n tels qu'il existe des arcs $T_1 \rightarrow T_2, \ldots T_{n-1} \rightarrow T_n$
- Une expression de chemins XPath permet de sélectionner des chemins à parcourir dans l'arbre du document en partant d'un noeud origine jusqu'à un ou plusieurs noeuds destination
- Expression de chemin simple : nom de racine suivi par une séquence de balises
- Expression de chemin généralisée : utilisation de motifs (*patterns*, sous-chaîne de caractère, disjonction, optionalité, nombre quelconque de balises)

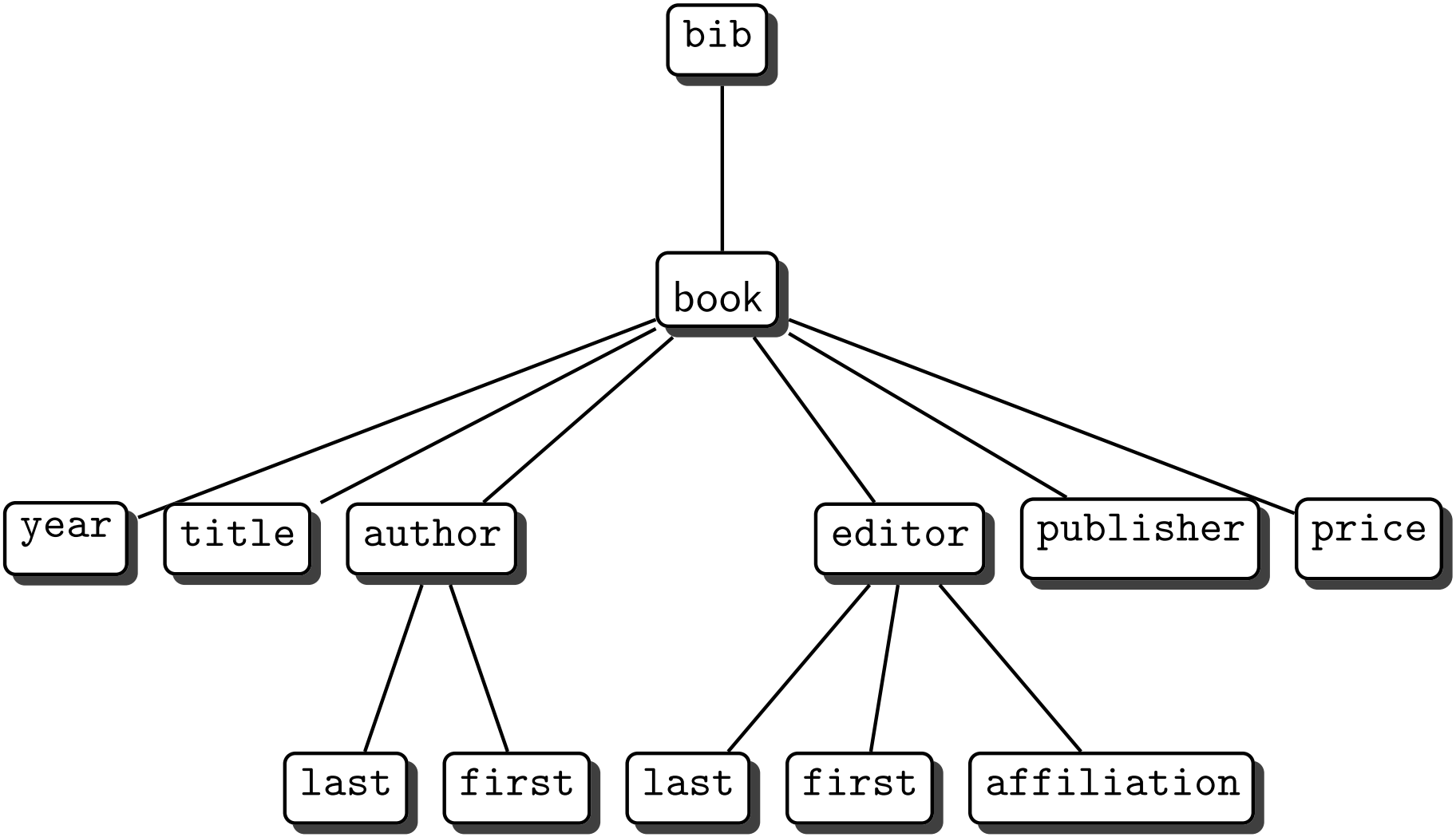
XPath (types)

- Types
  - Boolean, Number, String, Node-sets (ensemble de sous-arbres)
- Types de noeuds : types d'un arbre XML
  - processing instruction nodes (instructions)
  - comment nodes (commentaires)
  - root nodes (racine)
  - element nodes (élément)
  - attribute nodes (attribut)
  - namespace nodes (attributs d'un noeud)
  - text nodes (contenu)

DTD Exemple

```
<!ELEMENT bib (book*)>
<!ELEMENT book (title, (author+| editor+),
publisher, price)>
<!ATTLIST book year CDATA #REQUIRED>
<!ELEMENT author (last, first)>
<!ELEMENT editor (last, first, affiliation )>
<!ELEMENT title (#PCDATA )>
<!ELEMENT last (#PCDATA )>
<!ELEMENT first (#PCDATA )>
<!ELEMENT affiliation (#PCDATA )>
<!ELEMENT publisher (#PCDATA )>
<!ELEMENT price (#PCDATA )>
```

Exemple


bib
book
year
title
author
editor
publisher
price
last
first
last
first
affiliation

Syntaxe

- Le principe de la syntaxe XPath est semblable à celui de l'adressage du système de fichier. C'est une suite d'étapes, séparées par des '/' :

$$[/]\texttt{étape}_1/\texttt{étape}_2/.../\texttt{étape}_n$$

- Une étape a la forme suivante :

$$\texttt{Axe} :: \texttt{filtre}[\texttt{prédicat}_1][\texttt{prédicat}_2]...$$

  - L'axe définit le sens du parcours des noeuds
  - Le filtre indique le type des noeuds qui seront retenus dans l'évaluation de l'expression
  - Le (ou les) prédicat(s) expriment des propriétés que doivent satisfaire les noeuds retenus

Evaluation d'une expression

- L'évaluation d'une étape peut être une valeur, ou un ensemble de noeuds (node-set)
- Les étapes sont évaluées les unes après les autres :
  - À partir du noeud contexte, on évalue l'étape 1, qui renvoie un ensemble de noeuds
  - Chaque noeud de cet ensemble est considéré comme noeud contexte pour l'évaluation de l'étape 2
  - On procède de la même manière pour les autres étapes
- Remarques
  - Les noeuds ne sont pas extraits du document
  - Un noeud ne peut être référencé qu'une seule fois dans un même ensemble

Exemples

- Un chemin qui commence par / représente un chemin absolu (qui prend son origine à la racine du document) vers l'élément requis

Exemple

`/bib` sélectionne l'élément racine bib

- Un chemin qui commence par // sélectionne tous les éléments du document qui correspondent au critère qui suit

Exemple

`//editor` sélectionne tous les éléments editor rencontrés

- L'étoile * sélectionne tous les éléments localisés par ce qui la précède dans le chemin

Exemple

`//editor/*` sélectionne le contenu de tous les éléments editor rencontrés

```
<bib>
   <book>
        <title/>
        <editor>
                <last/>
                <first/>
                <affiliation/>
        </editor>
        <publisher/>
        <price/>
   </book>
<book>
        <title/>
        <editor>
                <last/>
                <first/>
                <affiliation/>
        </editor>
        <publisher/><price/></book>
</bib>
```

```
/bib
//editor
//editor/*
```

```
<bib>
   <book>
        <title/>
        <editor>
               <last/>
               <first/>
               <affiliation/>
        </editor>
        <publisher/>
        <price/>
   </book>
<book>
        <title/>
        <editor>
               <last/>
               <first/>
               <affiliation/>
        </editor>
        <publisher/><price/></book>
</bib>
```

```
/bib
//editor
//editor/*
```

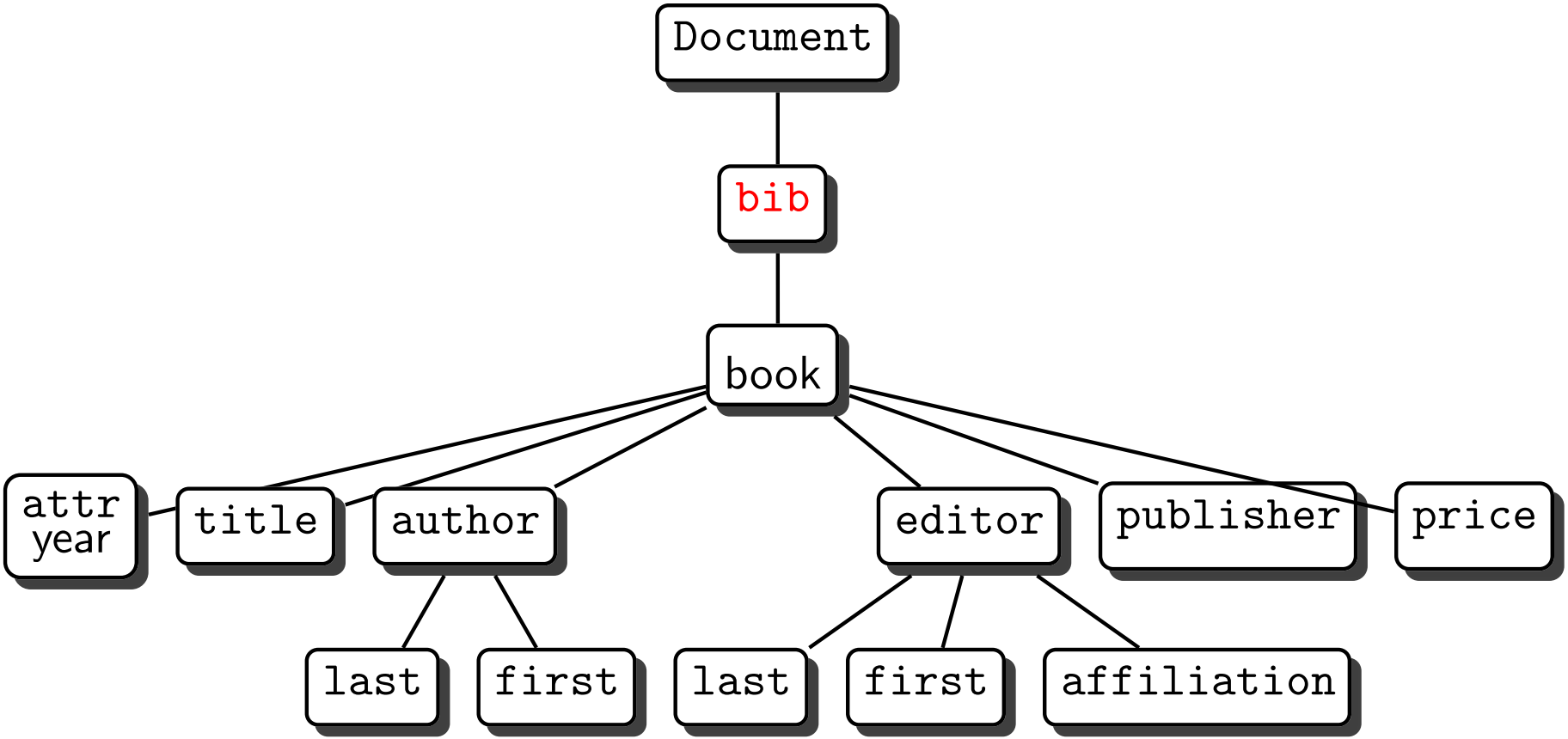
```
<bib>
   <book>
        <title/>
        <editor>
                <last/>
                <first/>
                <affiliation/>
        </editor>
        <publisher/>
        <price/>
   </book>
<book>
        <title/>
        <editor>
                <last/>
                <first/>
                <affiliation/>
        </editor>
        <publisher/><price/></book>
</bib>
```

```
/bib
//editor
//editor/*
```

Vue arborescente

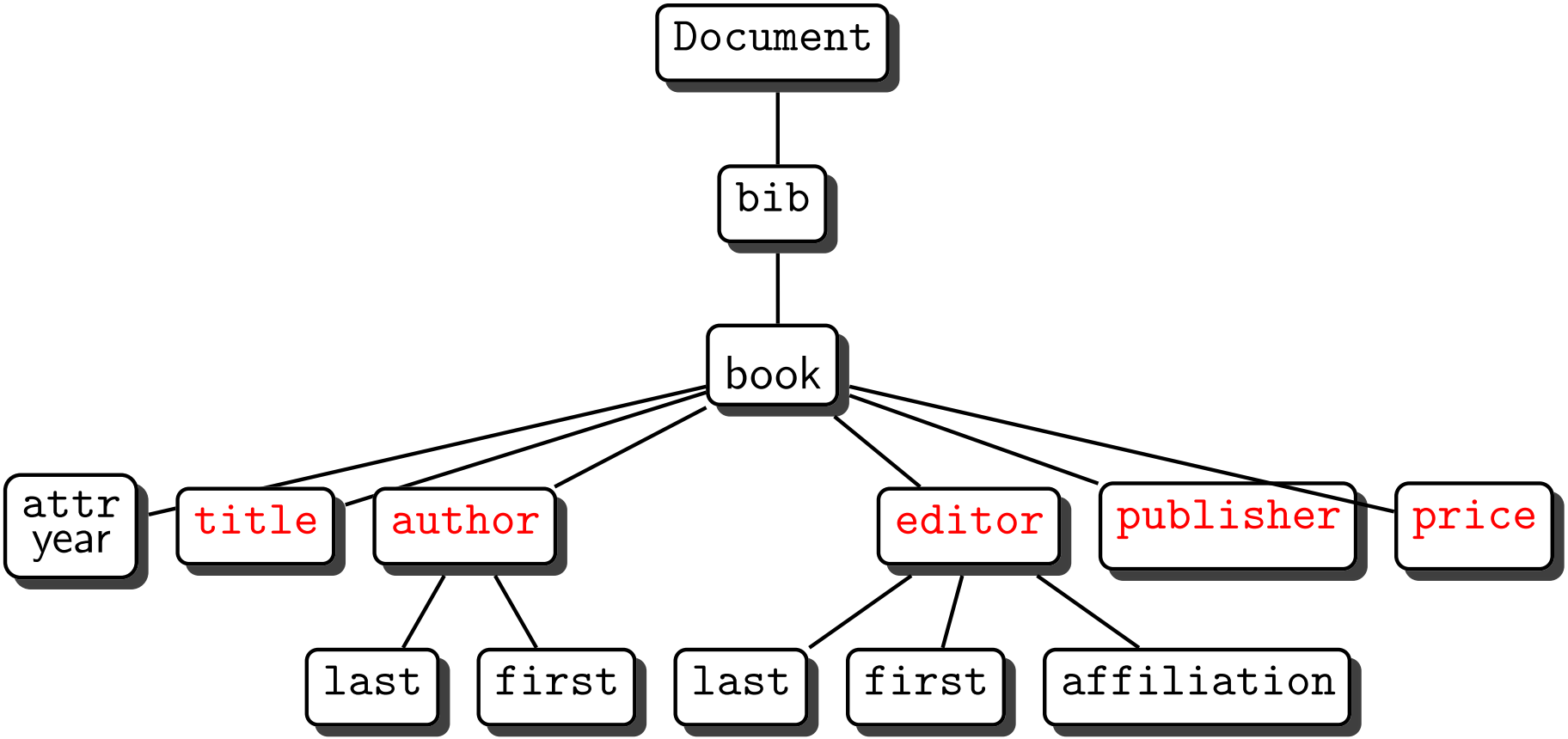
/bib

//book/*



Vue arborescente

```
/bib
//book/*
```



Exemples de prédicats

- Une expression entre crochets permet de spécifier plus précisément un élément
- Un nombre entre crochets donne la position d'un élément dans le jeu sélectionné

Exemple

`/bib/book[1]` sélectionne le premier élément book de bib

- La fonction last() sélectionne le dernier élément

Exemple

`/bib/book[last()]` sélectionne le dernier élément book de bib

```
<bib>
<book>
<title/>
<editor>
<last/>
<first/>
<affiliation/>
</editor>
<publisher/
<price/>
</book>
<book>
        <title/>
         <editor>
                 <last/>
                 <first/>
                 <affiliation/>
         </editor>
       <publisher/><price/></book>
</bib>
```

/bib/book[1]

/bib/book[last()]

Attributs

- Les attributs sont spécifiés par le préfixe @

Exemple

- `//@year` sélectionne tous les attributs year du document
- `//book[@year]` sélectionne tous les éléments book qui ont un attribut year

- Les valeurs des attributs peuvent être utilisés comme critère de sélection

Exemple

`//book[@year="2004"]` sélectionne tous les éléments book ayant un attribut year dont la valeur est 2004

Fonctions

- Positions relatives et information locale
  - position() : position dans le contexte
  - name() : retourne le nom de l'élément
  - count() : cardinalité d'un node-set (compte le nb d'éléments sélectionnés)
  - last() : indicateur de dernière position
- Fonctions booléennes : and, or, not
- opérateurs : mod, $>$, $<=$, etc.
- fonctions de chaîne : contains, substringbefore, stringlength, ...
- fonctions d'environnement : normalizespace (supprime les espaces de début et de fin, remplace les séquences d'espaces blancs par un seul espace)
- Permettent des requêtes de base
  - analyse des contenus et noms de balises/attributs

Exemples

- `//*[name()= "book"]` éléments book
- `//*[count(book)="2"]` éléments ayant 2 enfants book
- `//*[count(*)="2"]` éléments ayant deux enfants
- `//*[stringlength(name())="3"]` retourne tous les éléments dont le nom a 3 caractères

Axes

- Position (noeud courant) : self (.)
- Navigation de l'arbre
  - descendant direct : child (/)
  - descendant indirect : descendant (//)
  - ancêtre direct : parent (..)
  - ancêtre indirect : ancestor
  - frères : following-sibling / preceding-sibling
- Navigation dans le document
  - noeuds suivants/précédent : following et preceding
- Combinaisons : ancestor-or-self, descendant-or-self
- namespace
- attribut (@)

Axes

- L'axe enfant (child) contient les enfants du noeud contextuel. L'axe enfant est l'axe par défaut, et il peut être omis
- L'axe self renvoie le noeud courant
- L'axe parent contient le parent du noeud contextuel, s'il en a un
- L'axe descendant contient tous les descendants du noeud contextuel (enfant, petit-enfant, etc.), à l'exception des noeuds attributs et espaces de nom
- L'axe ancêtre (ancestor) contient tous les éléments ancêtres du noeud contextuel (parent, parent du parent, etc.). Il contient forcément le noeud racine (sauf si le noeud contextuel est la racine)

Axes

- L'axe followingsibling contient tous les noeuds frères qui suivent le noeud contextuel
- L'axe precedingsibling contient tous les frères prédécesseurs du noeud contextuel
- L'axe suivant (following) contient tous les noeuds du même document que le noeud contextuel qui sont après le noeud contextuel dans l'ordre du document, à l'exclusion de tout descendant, des attributs et des espaces de noms
- L'axe *précédent* (*preceding*) contient tous les prédécesseurs du noeud contextuel ; si le noeud contextuel est un attribut ou un espace de noms, le précédent est vide

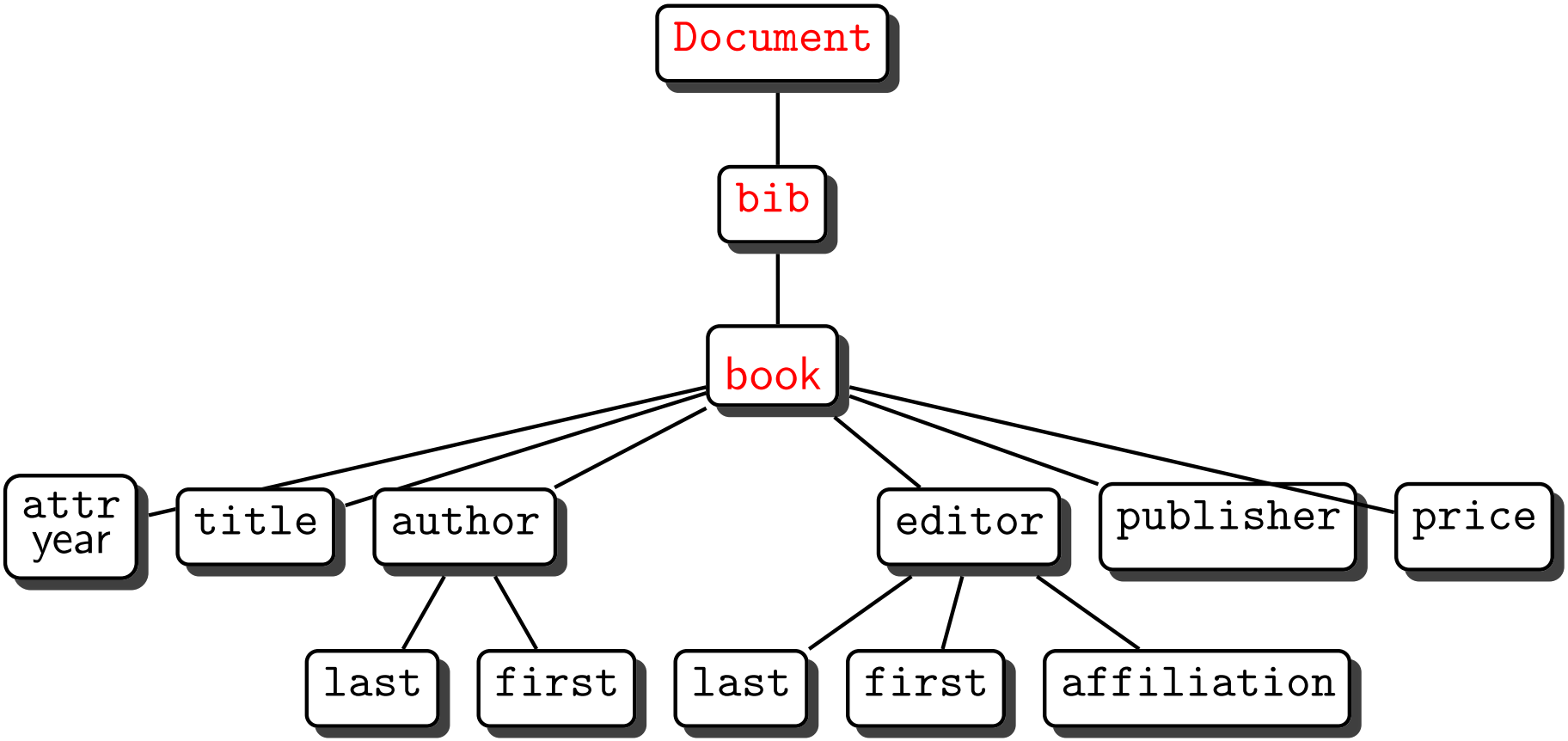
Axes

- L'axe descendant-or-self contient le noeud contextuel et ses descendants
- L'axe ancestor-or-self contient le noeud contextuel et ses ancêtres ; ainsi l'axe ancestor-or-self contient toujours le noeud racine
- Les axes ancestors, descendants, following, preceding et self partitionnent un document (ignorant les attributs et les noeuds d'espace de nom) : il ne se chevauchent pas et ensemble ils contiennent tous les noeuds d'un document
- Le symbole | permet de combiner des chemins

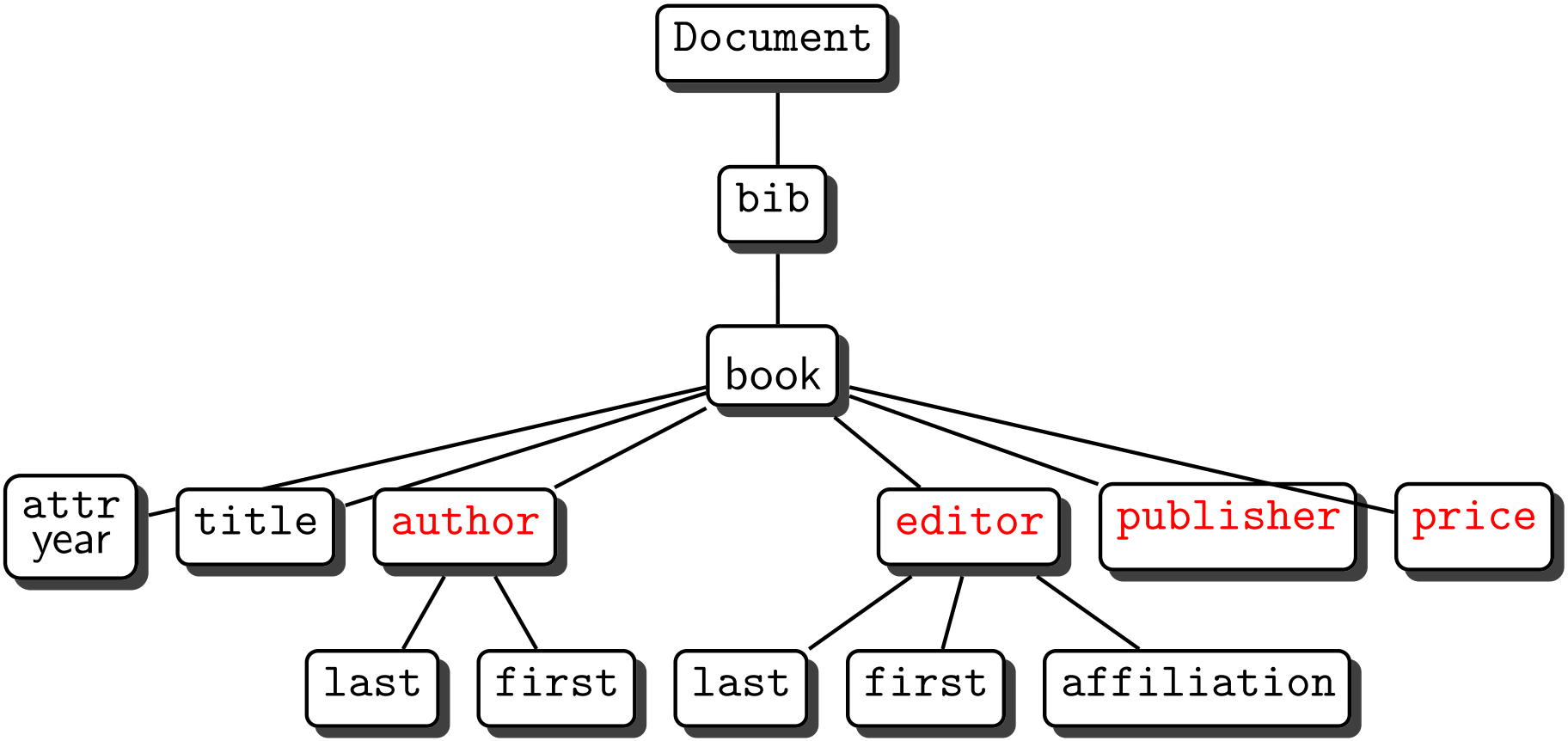
Exemples

- //editor/parent : :* renvoie les parents du noeud editor, càd book
- /book/editor/descendant : :* renvoie les descendants de editor, càd last, first, affiliation
- //last/ancestor : :* renvoie tous les ancêtres de last, càd author, editor, book, bib
- //editor/following-sibling : :* renvoie les noeuds publisher, price
- //editor/preceding-sibling : :* renvoie les noeuds author, title
- //author/following : :* renvoie les noeuds editor, last, first, affiliation, publisher, price
- //author/preceding : :* renvoie le noeud title

```
//publisher/ancestor::*
```



title/following-sibling::*



Abréviations

- child : : est l'axe par défaut, et peut être omis
- /child : :book est équivalent à /book
- child : :book/child : :title peut s'écrire book/title
- attribute : : peut être remplacé par @
- child : :book[attribute : :year= 2002] peut s'écrire book[@year= 2002]
- // est l'abréviation de /descendant-or-self : :node()/
- . est l'abréviation de self : :node()
- .. est l'abréviation de parent : :node()

XPath

- XPath 1.0
- XPath 2.0 :
  - type étendu (XML Schema)
  - langage plus naturel
  - brique de base pour XQuery
  - compatibilité descendante avec XPath 1.0

Pour jouer ce soir

- Sous Linux, navigateur XML en ligne de commande : xmllint
- Sous MS Windows : XML Cooktop, XML BluePrint, StylusStudio, ...