UE Ingénierie documentaire
Master 2ème Année

Fabrice Lefèvre

2021

XML
Partie 1 - Langage de structuration et annotation

Fabrice Lefèvre

fabrice.lefevre@univ-avignon.fr

2021

Développement du web

- **Moteur de recherche** : nécessite l'interprétation des informations visées pour pouvoir les indexer efficacement
  - accès à de grandes collections de données : journaux, livres, documents divers…
- **Commerce électronique** : les entreprises veulent pouvoir échanger des informations et pas uniquement pour les afficher
  - accès aux catalogues, aux commandes, fichiers clients…
- **Services en ligne** : pouvoir envoyer des données en ligne pour leur faire appliquer un traitement automatique particulier (par exemple la publication, la vérification d'information...)
  - SaaS, Software as a Service

Nouvelles nécessités

Echange et publication de données/informations

- *Réseau hétérogène* : représentation des données indépendante de la machine utilisée et du type de réseau utilisé pour le transport
- *Applications variées* : représentation des données indépendante de l'application source

Pourtant la représentation des données dépend très fortement de l'application visée :

➔Nécessité d'un format pivot

Approche de structuration et annotation

Approche retenue consiste :

- à conserver la forme brute (ou initiale) des données
  - ➔ `TEXTE`
- à superposer les informations permettant de structurer et d'interpréter les données contenue dans le document
  - ➔ `BALISAGE`

Balisage (1)

- Sert à repérer une partie du document présentant une caractéristique particulière afin de la différencier du reste du document
- Par exemple, application d'un style de présentation (police grasse ou italique...) comme dans les traitements de texte
  - à ne pas confondre avec les éditeurs de texte
- Balises **ouvrantes** et **fermantes** pour expliciter dans le flux le début et la fin de la partie concernée
- Une partie balisée devient un **élément** du document

Balisage (2)

Deux types de balisages :

- **procédural** : définit le traitement à appliquer à la zone balisée

  `HTML, PS, PDF, RTF...`

- **descriptif** : identifie les types d'information présents dans le document (appelée aussi *Structure Logique Generique*, SLG)

  `SGML, XML, RSS, RDF, OWL...`

Exemple de document

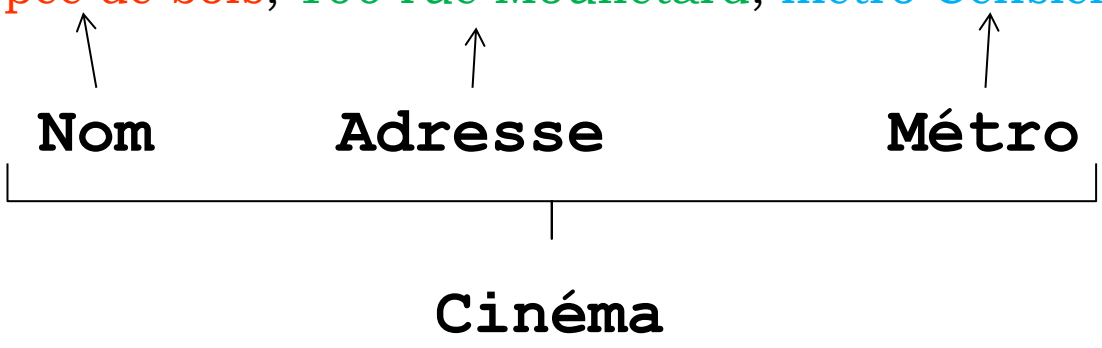
Le contenu :

L'Epée de bois, 100 rue Mouffetard, métro Censier-Daubenton

Structuration de document

La structure :

L'Epée de bois, 100 rue Mouffetard, métro Censier-Daubenton



Exemple de fichier balisé (1)

```
<CINEMA>L'Epée de Bois,  100,rue Mouffetard, métro Censier-
   Daubenton</CINEMA>
```

Intégrer la **structure** dans/avec
les données elles-mêmes.

Exemple de fichier balisé (2)

```
<CINEMA><NOM>L'Epée de Bois</NOM>, <ADRESSE>100,rue
   Mouffetard</ADRESSE>, <METRO>métro Censier-Daubenton</METRO>
   </CINEMA>
```

Cette représentation facilite l'échange **informatique** de documents.

Apparition de la structure

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<CINEMA>
<NOM>
   L'Epée de Bois
</NOM>
<ADRESSE>
   100, rue Mouffetard
</ADRESSE>
<METRO>
   métro Censier-Daubenton
</METRO>
</CINEMA>
```

Cette représentation facilite l'échange **informatique et "humain"** de documents.

Pas toujours aussi simple…

Le contenu :

La rue Mouffetard est une rue très vivante du Quartier Latin. On y trouve de nombreux restaurants et quelques salles de cinéma. L'Epée de bois, situé au 100, proche du restaurant le Mouff Village, est accessible par le métro Censier-Daubenton.

…de faire apparaître la structure

Le contenu :

La rue Mouffetard est une rue très vivante du Quartier Latin. On y trouve de nombreux restaurants et quelques salles de cinéma. L'Epée de bois, situé au 100, proche du restaurant le Mouff Village, est accessible par le métro Censier-Daubenton.

…complète

Le contenu :

La rue Mouffetard est une rue très vivante du Quartier Latin. On y trouve de nombreux restaurants et quelques salles de cinéma. L'Epée de bois, situé au 100, proche du restaurant le Mouff Village, est accessible par le métro Censier-Daubenton.

Structure documentaire

Structure arborescente vs. structure relationnelle (ou dynamique)

- Structure arborescente : document divisible en sous-parties disjointes jusqu'aux éléments balisés
- Implication 1 : le chevauchement des balises est interdit
- Implication 2 : le document est représentable sous forme d'un arbre

Exemple d'ambiguïté : <gras>ce texte est d'abord en gras <italique>puis en italique et gras</gras>, finalement en italique seulement</italique>

➔peut être surmontée malgré tout (si besoin) !

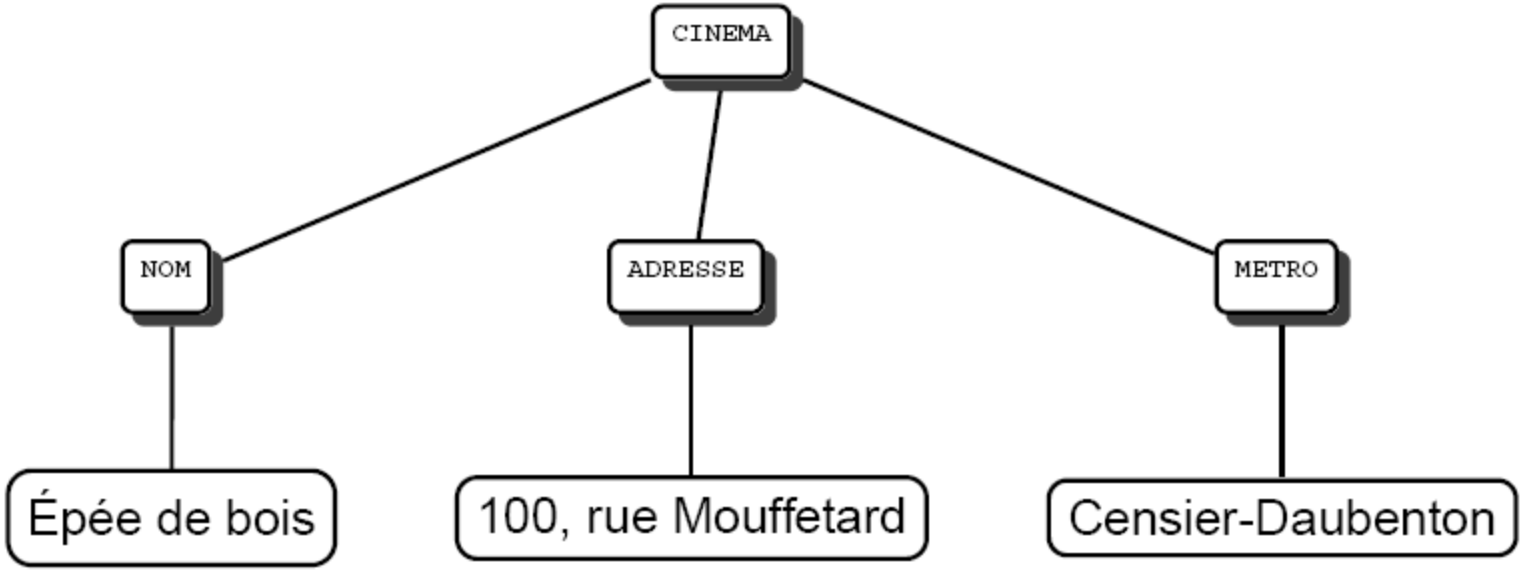
Forme arborescente

Représentation arborescente créée à partir de l'analyse du document sérialisé :

- Nœud racine de type **Document**
  - tellement évident qu'on oubliera souvent de le représenter !
- Catégories syntaxiques principales (texte, balises, commentaires) se traduisent par différents types de nœuds (**Text, Element, Comment**)
  - propriétés et représentation différentes
- Ensemble des nœuds forment un arbre qui reflète l'imbrication des éléments dans la forme sérialisée

Forme arborescente (2)

- Un arbre, constitué de noeuds typés (éléments, texte…)



- La structure des arbres est définie par un *modèle*
  - par exemple DOM (*Document Object Model*).
- **Il est plus facile de raisonner sur la forme arborescente pour concevoir des traitements**

Equivalence des representations

représentation sérialisée : **syntaxe** du langage de balise

représentation arborescente : **modèle** de document

Composants :

- les éléments (et leurs attributs)
- les sections de texte
- les sections littérales
- les commentaires
- les instructions de traitement

Ainsi que quelques règles sur la structure d'un document.

→ Objectif : équivalence

XML, langage de balisage et structuration

- XML (et HTML, XHTML…) est une instance définie à partir de SGML
- *SGML, Standart Generalized Markup Language*
- Développé dés les années 80
- Méta-langage permettant de décrire les autres langages à balises par spécialisation
- S'appuie sur une DTD (*Definition Type Document*) pour définir la structure de ses éléments
- Puissant, évolutif mais extrêmement complexe (1/2 millier de pages de spécification)

Caractéristiques

- Séparation de la **structuration des données** de leur représentation "visuelle » ou de leur manipulation (traitements)
  - `on dit ce que c'est, pas ce qu'on doit en faire`
  - `pas procédural`
- Format de représentation des données **orienté objet**
- Technologies associées pour la **validation** et la mise en forme
- **Portabilité** entre systèmes d'exploitation
- Utilisation conjointe possible avec **tous les langages** de programmation (moyennant une API)

➔**Format pivot** : candidat de choix pour la communication entre applications

Modèle de données

Le langage XML est le modèle de données pour le Web

Un document XML :

- peut être échangé facilement (format texte)
- permet la représentation de (presque) toute information structurée
- n'est pas associé à un contexte particulier
- manipulations (archivage, transformation, interrogation) sont facilitées par la disponibilité d'outils performants du fait de la standardisation

Modèle de données

W3C Recommendation

"The design goals for XML are:

- XML shall be straightforwardly usable over the Internet.
- XML shall support a wide variety of applications.
- XML shall be compatible with SGML.
- It shall be easy to write programs which process XML documents.
- The number of optional features in XML is to be kept to the absolute minimum, ideally zero.
- XML documents should be human-legible and reasonably clear.
- The XML design should be prepared quickly.
- The design of XML shall be formal and concise.
- XML documents shall be easy to create.
- Terseness in XML markup is of minimal importance.

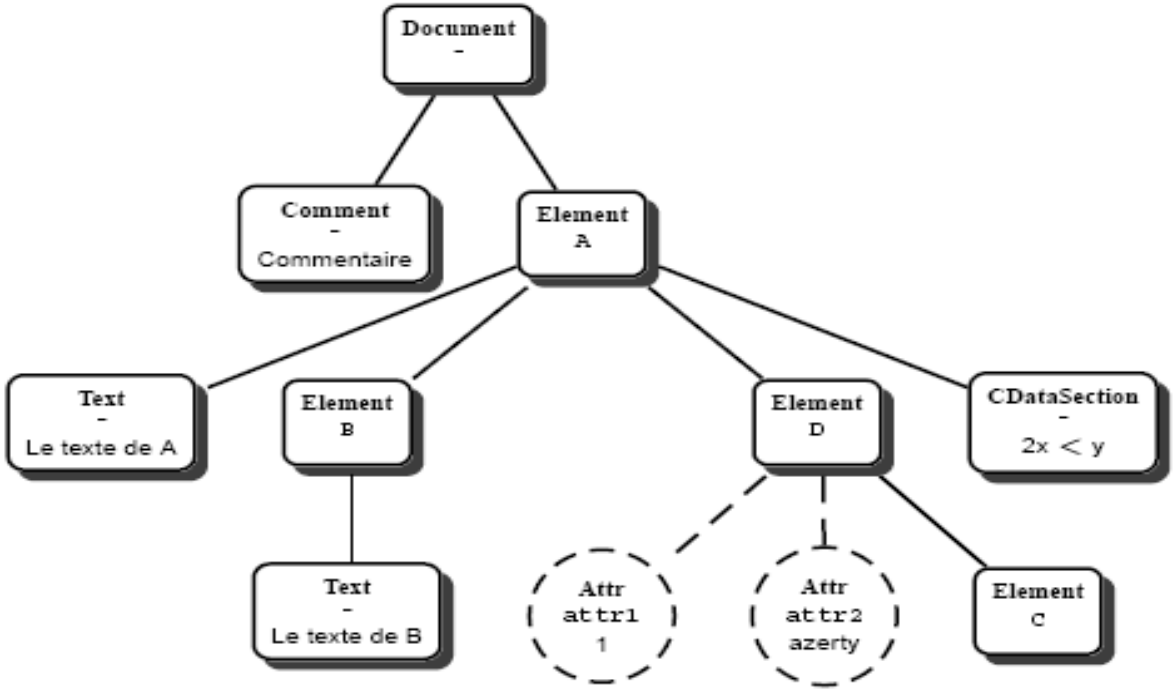
Exemple 2

```
<!-- Prologue -->
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<!-- Elément racine →
<biblio>
<!-- Premier fils -->
   <livre>
   <titre>Les Misérables</titre>
        <auteur>Victor Hugo</auteur>
        <nb_tomes>3</nb_tomes>
   </livre>
<livre>
   <titre>L'Assommoir</titre>
   <auteur>Emile Zola</auteur>
</livre>
<livre lang="en">
       <titre>David Copperfield</titre>
       <auteur>Charles Dickens</auteur>
     </livre>
   </biblio>
```

Exemple 3 (1)

```
<?xml version="1.0"encoding="ISO-8859-
    1"?>
<!– Commentaire -->
<A>
Le texte de A <B>Le texte de B</B>
<D attr1="1"attr2="azerty"><C /></D>
<![ CDATA [ 2x<y ]]>
</A>
```

Exemple 3, version arborescente (2)



AVIGNON UNIVERSITÉ

Passer à Syntaxe XML (jusqu'à « Commentaires »)

Composants de XML

- DTD ou XML schema
- Parser/processeur XML
  - DOM (Document Object Model) ou SAX (Simple API for XML)
- Espaces de noms

DTD

- Définissent la structure de l'information contenue dans les documents cibles
- Impliquent la notion de validité : accord entre les structures attendues et observées
- Parser : application permettant de d'analyser l'information contenue dans un document XML
  - `Peuvent être validateurs (ou non) s'ils vérifient la validité des documents traités`

Bien formé et valide

- Bien formé : respecte la syntaxe du XML

  Principalement :
  - `unique élément racine décomposable en éléments fils eux-mêmes décomposables en éléments`
  - `pas de chevauchement de balises`
  - `fermetures des balises`
- Valide : respecte les contraintes données par la grammaire définies dans la DTD ou le schéma XML

DOM/SAX

DOM : définit l'interface entre les parsers et les applications

- `interface objet = définit des propriétés et des méthodes pour tous les éléments`

- Permet d'accéder aux données d'un document XML par programmation
- Fait l'objet d'une normalisation par le W3C
  - `existe beaucoup de variations entre les implémentations`

SAX : alternative au DOM pour la manipulation de document XML volumineux

- Plus complexe mais plus efficace
- Interface avec Java

Espaces de noms

- Définissent des vocabulaires séparés sous XML
- Permettent la cohabitation de plusieurs domaines XML différents (définis par des DTDs séparées)
  - en évitant les conflits entre termes identiques lorsqu'ils doivent être utilisés dans un même document

Rôle du XML sur le web

La technologie XML (dans son acceptation large) est présente à tous les niveaux sur le web :

- **stockage des données** : entrepôts/bases de données XML
- **manipulation des données** : XPath/XQuery (interrogation), DOM/SAX (programmation), XSLT (transformation), ...
- **échanges entre applications** : services web (SOAP)
- **publication de données** : texte (XHTML, WML...), graphiques (SVG), multimedia (SMIL, VoiceXML...)
- **description des données** : web sémantique (RDF, OWL, DAML+OIL...)

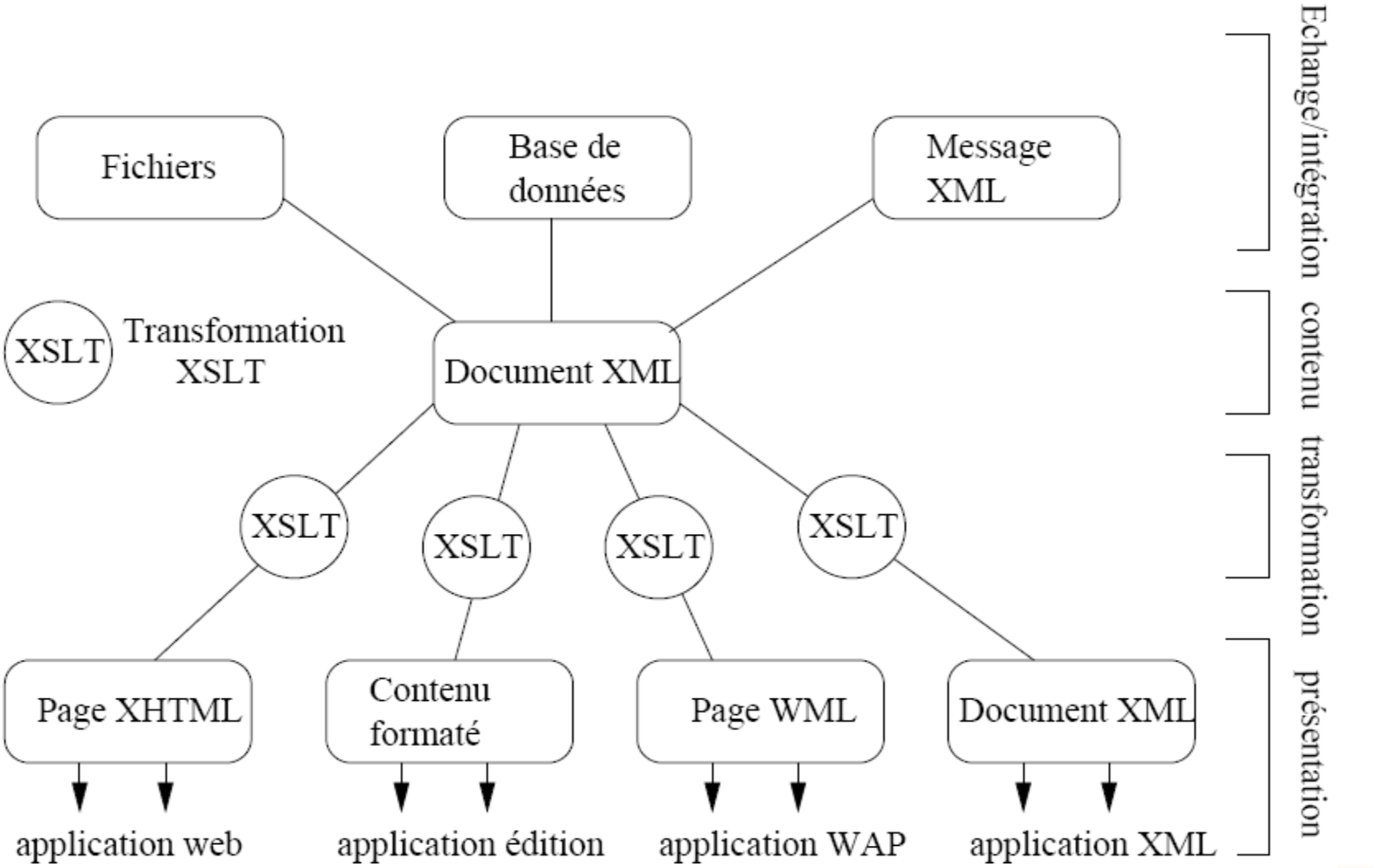
Publication des données avec XSLT

Objectif : séparer réellement la gestion du contenu de sa présentation

- gestion du contenu : décrire les informations avec un vocabulaire XML adaptée
- présentation du contenu : mettre en forme les informations dans le contexte d'une application particulière (publication Web, impression, projection...)

- XSLT permet d'écrire des programmes de conversions par réécriture des documents XML

Gestion de l'information avec XML



Rôle de XSLT

XSLT permet :

- de prendre en entrée un document XML source
- de produire en sortie un autre arbre XML
- d'introduire dans le document de sortie des fragments du document source

Permet la transformation vers tout format compatible XML :

- XHTML pour la présentation web
- WML pour la présentation WAP (Wireless Application Protocol) pour les mobiles
- SMIL pour la présentation multimédia
- XSL-FO (Formating Objects) pour la production de documents papier

De HTML à XML

```
<html>
    <head>
        <title>  ACM Conference  </title>
    </head>
    <body>
      This  conference focuses on...
    </body>
</html>
```

```
<Conference>
        <hasName>
            ACM Conference
        </hasNname>
  <hasDescription>
     This  conference focuses on…
  </hasDescription>
</Conference>
```

De XML à HTML

Si l'on veut produire un fichier HTML à partir de données mises sous format XML, il faut :

- créer éventuellement un fichier définissant les balises utilisables ;
- créer le fichier de données XML ;
- créer la feuille de style XSL permettant la création du fichier HTML ;
- créer éventuellement une feuille de style CSS.

XSL-FO

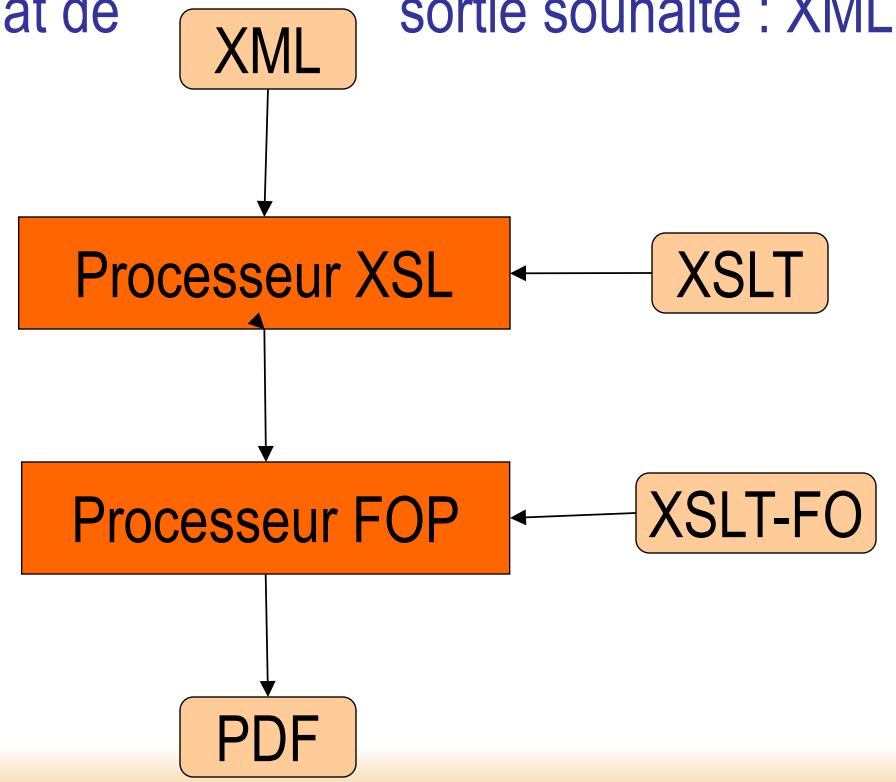
XSL-FO (extended Style Sheet Formatting Object) est un langage de description de formatage de documents avec XML

- On indique les paramètres de mise en page permet d'exprimer le rendu d'un document :
  - `pagination`
  - `notes de bas de page`
  - `marges`
  - `emplacement des différents objets sur la page`
  - `polices des caractères`
  - `affichage de tableaux…`
- Dans un document XSL-FO, le contenu est entre des balises de formatage → un processeur se charge de produire le document

Processeur XSL-FO

FOP (*Formatting Objects Processor*) une application java qui utilise XSL-FO

- FOP lit un arbre de formatage d'objet (FO) et renvoie une page suivant le format de sortie souhaité : XML, PDF, PS, SVG, TXT…



Applications basées sur XML

XML est la base de plusieurs applications qui utilisent ses capacités de description de documents au delà du simple texte :

- XHTML pour la réalisation de documents web
- SVG pour la réalisation de graphiques vectoriels
- SMIL pour la réalisation de documents multimédia
- VoiceXML permet la mise en oeuvre de navigateur web vocaux (*web browser*)

SVG

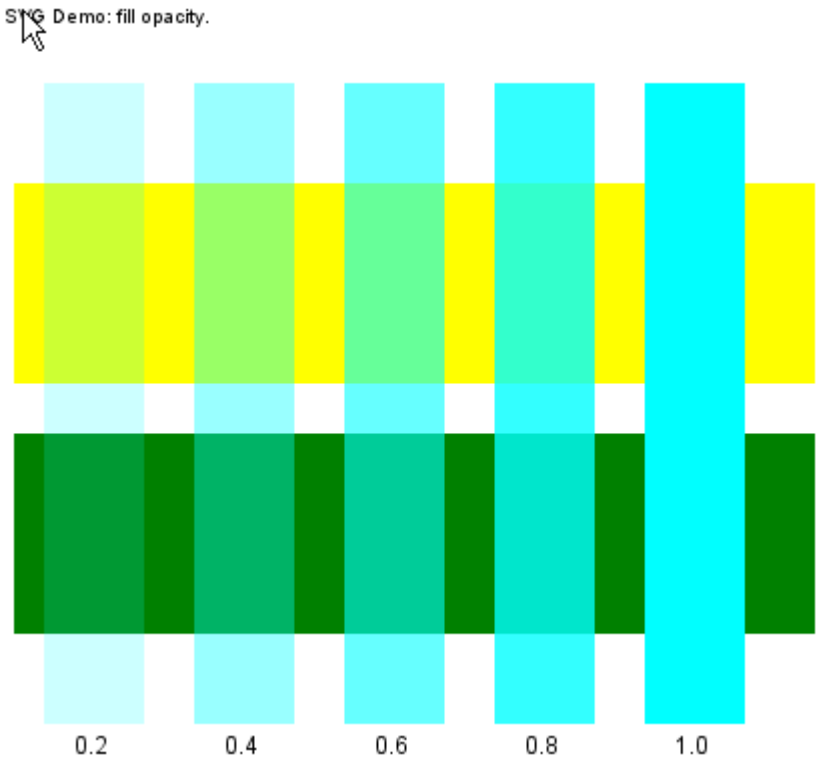
- *Scalable Vector Graphics*
- Langage permettant d'écrire des graphiques vectoriels 2D en XML
- Inventé en 1998 par un groupe de travail d'industriels pour répondre à un besoin de graphiques légers, dynamiques et interactifs
- Chaque élément graphique est représenté par un élément XML qui est paramétrable avec des attributs XML et qui hérite d'attributs de ses parents.

- Pas encore supporté en natif par tous les navigateurs web

SVG, exemple

```
<svg width="500" height="500">
<text x="5" y="20" style="font-size:10">SVG Demo: fill opacity.</text>
<rect x="10" y="100" width="400" height="100" style="fill:yellow"/>
<rect x="10" y="225" width="400" height="100" style="fill:green"/>
<g style="fill:cyan">
   <rect x="25" y="50" width="50" height="320" fill-opacity="0.2"/>
   <rect x="100" y="50" width="50" height="320" fill-opacity="0.4"/>
   <rect x="175" y="50" width="50" height="320" fill-opacity="0.6"/>
   <rect x="250" y="50" width="50" height="320" fill-opacity="0.8"/>
   <rect x="325" y="50" width="50" height="320"/>
</g>
<text x="40" y="385">0.2</text>
<text x="115" y="385">0.4</text>
<text x="190" y="385">0.6</text>
<text x="265" y="385">0.8</text>
<text x="340" y="385">1.0</text>
</svg>
```

SVG, exemple



SMIL

SMIL (sans E !) est un langage développé pour la création de documents multimédia avec XML

Principes :

- on indique une fenêtre d'affichage avec différentes régions pour l'affichage des composants
- on place les composants dans les différentes régions (positionnement spatiale)
- on synchronise l'affichage des composants (positionnement temporel)
- Différents types d'éléments peuvent être insérés dans un document SMIL :
  - `objets multimedia : texte, image, audio, video et flux de texte`
  - `éléments de synchronisation : séquence et en parallèle`

VoiceXML

- **Objectif** : extension du Web permettant aux utilisateurs d'accéder à des sites Web à l'aide de commandes vocales, ainsi que d'en recevoir le contenu sous la forme de voix pré-enregistrées ou synthétiques, et également sous forme de musiques. Soit mettre en place des sites Web accessibles à l'aide d'un simple téléphone.
- VoiceXML (*Voice eXtensible Markup Language*) est un langage de description de services téléphoniques interactifs. C'est un langage textuel normalisé par le W3C et construit sur la syntaxe XML.
- VoiceXML est aux services vocaux ce que HTML est aux services web

➔**Notion de navigateur web vocal**

Navigateur vocal et VoiceXML