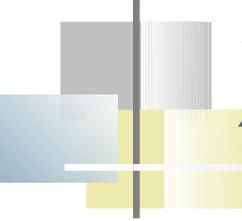


# Documents XML

## Description - Manipulation

---

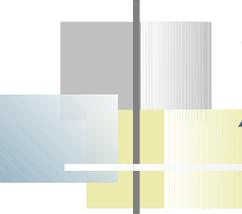
Elisabeth Muriasco  
muriasco@univ-tln.fr



# XML

---

- Format universel
  - Représentation la plus simple possible (chaîne de caractères) d'un contenu indépendant de toute application
- Publier l'information
  - outils de transformation simples pour convertir un contenu XML
- Échanger et intégrer l'information
  - assembler des contenus XML, ou au contraire en extraire des informations



# XML le langage du Web

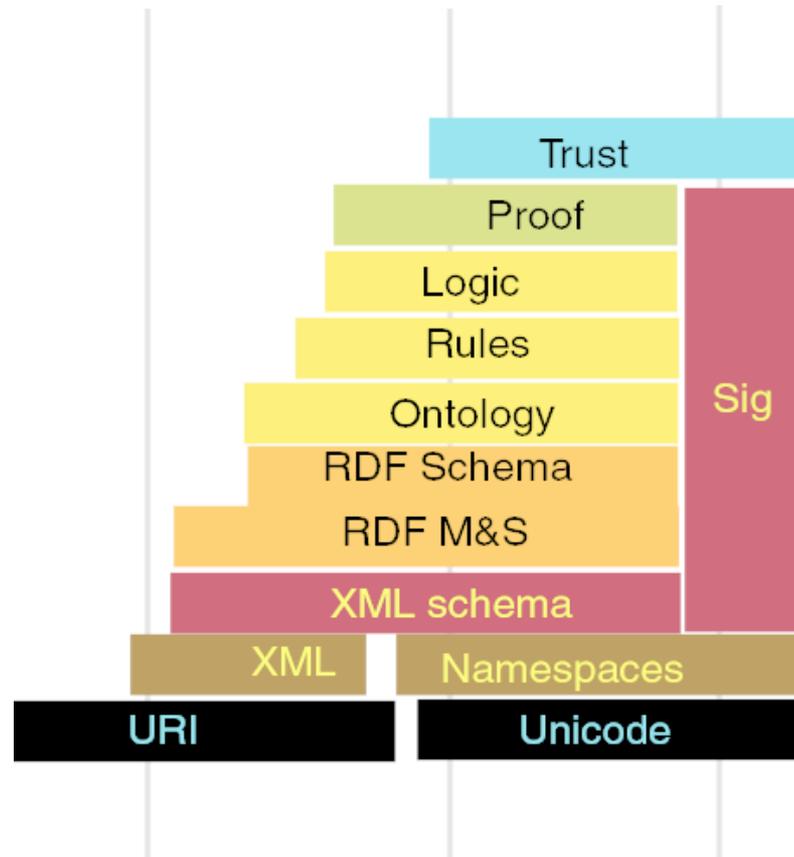
---

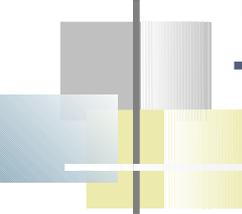
- XML est un langage pour représenter les données du Web
- XML standardise la manière dont l'information est :
  - Représentée
  - échangée
  - présentée
  - archivée
  - retrouvée
  - transformée
  - cryptée
  - ...

# La pyramide du W3C

## Les langages du W3C

Tim Berners-Lee:  
"Axioms, Architecture and Aspirations"  
W3C all-working group plenary  
Meeting  
28 February 2001

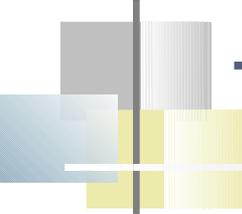




# Programme du cours (7 CM, 5 TD)

---

- Introduction
  - Documents structurés, XML, Bases de données
- Documents XML : description et manipulation
  - Description de documents
  - Typage - *Xschema*
  - Modélisation -- *XDM*
  - Localisation de composants XML -- *XPath*
  - Transformation de documents XML -- *XQuery*
  - Interrogation de documents XML -- *XSLT*



# Mais pas dans cet ordre... et CM-TD

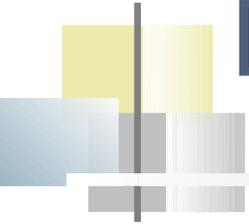
---

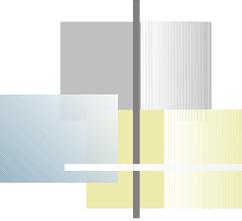
- Introduction
  - Documents structurés, XML, Bases de données
- Documents XML : description et manipulation
  - Description de documents
  - Modélisation - *XDM*
  - Localisation de composants XML -- *XPath*
  - Transformation de documents XML - *XSLT*
- Plus tard...
  - Typage -- *XSchema*
  - Interrogation de documents XML -- *XQuery*

# Introduction

## Documents - XML - BD

---



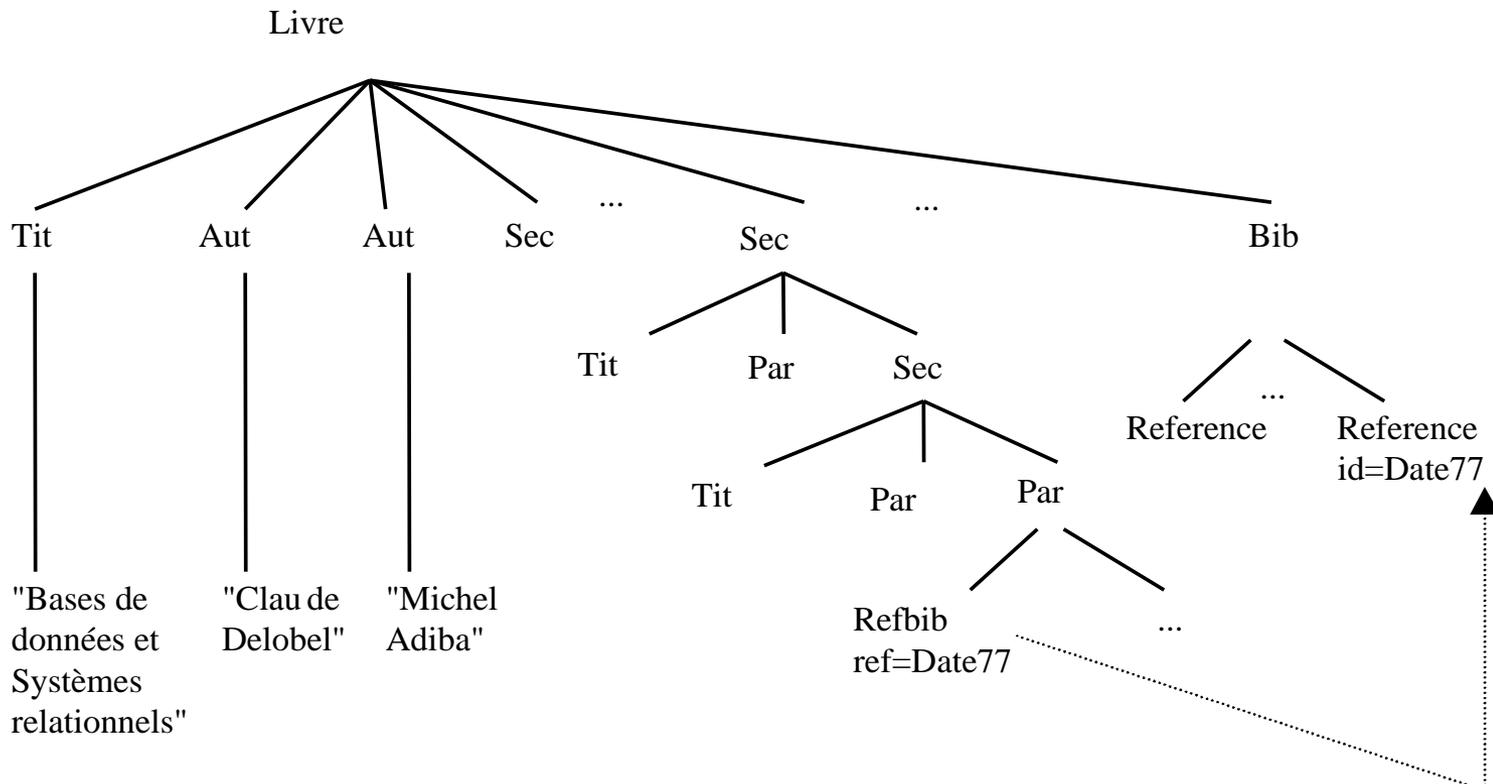


# Exemple de document XML (1)

---

```
<?xml version="1.0"?>
<Livre>
  <Titre>Bases de données et systèmes relationnels</Titre>
  <Aut>Claude Delobel</Aut>
  <Aut>Michel Adiba</Aut>
  ...
  <Sect>
    <Titre>Les problèmes d'intégrité</Titre>
    <Par>Ce chapitre aborde ...</Par>...
    <Sect>
      <Titre>Notes bibliographiques</Titre>
      <Par> Le problème ...</Par>
      <Par><Refbib ref="Date77">Date 1977</Refbib>dresse ...</Par>...
    </Sect>
  </Sect>...
  <Biblio>...
    <Reference id="Date77">DATE, C. J., [1977], An introduction ...
    </Reference>...
  </Biblio>
</Livre>
```

# Vision arborescente (partielle) du document XML



## Sémantique des balises :

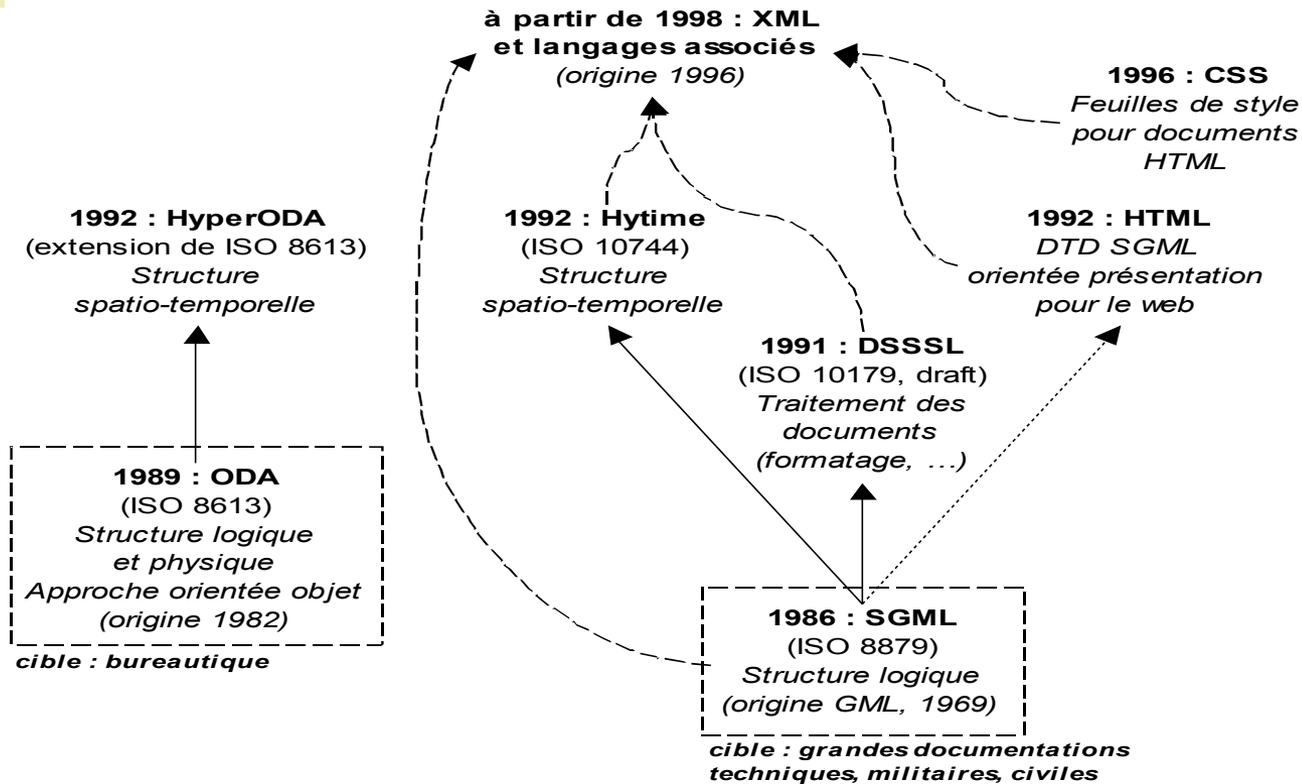
- Des indications structurelles : h1, ..., h6, title, ...
- Des indications de mise en page  
center, hr, b, i, big, small, ...

# Exemple de document HTML

```
<html>
  <head><title>Programme des journées</title></head>
  <body>
    <h2>Jeudi 16 Janvier 2006</h2>
    <ul>
      <li><b>8h30 :</b>Accueil</li>
      <li><b>8h45 :</b>Ouverture des Journées</li>
      ...
    </ul>
    ...
    <li>Validation des Acquis,
      <a href="mailto:carrez@cnam.fr">C. Carrez </a>
    </li>
  </body>
</html>
```

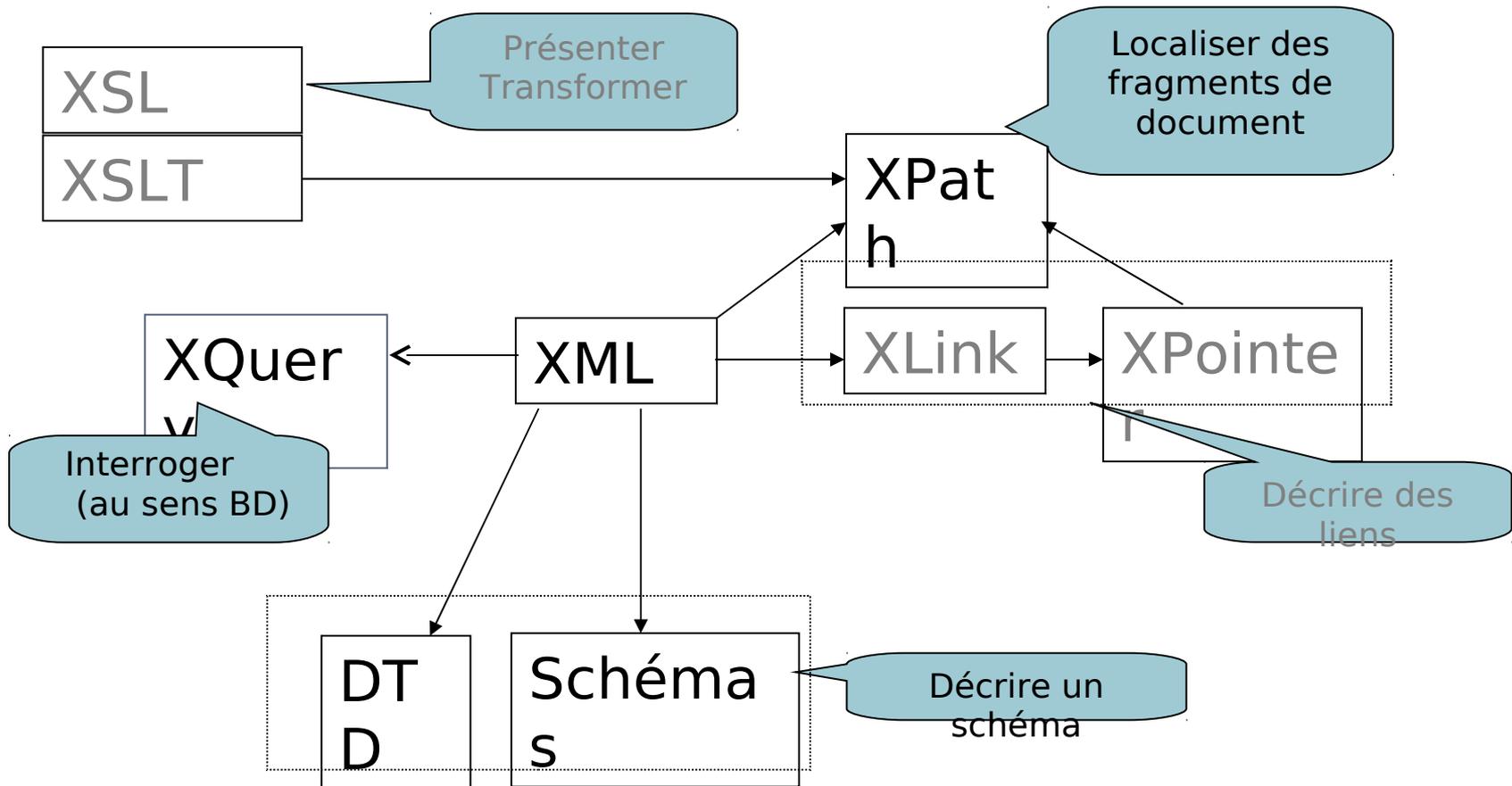
HTML est un langage pour **présenter des** informations à **l'écran**  
il ne permet pas d'échanger des données  
il ne permet pas un traitement des données autre que l'affichage

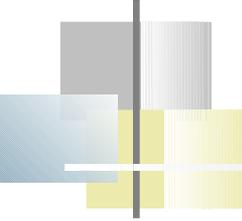
# Origine : le monde documentaire



- ▶ extension ou dérivation de la norme
- .....▶ application de la norme
- - -▶ est inspiré de

# XML et quelques-uns de ses langages associés



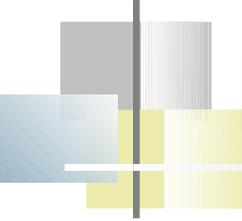


# Exemple de document XML (2)

---

- Données structurées

```
<?xml version="1.0"?>
<table>
  <personne>
    <nom>Bond</nom>
    <bureau>U1</bureau>
  </personne>
  <personne>
    <nom>Lupin</nom>
    <bureau>U2</bureau>
  </personne >
  <personne>
    <nom>Templar</nom>
    <bureau>U3</bureau>
  </personne>
</table>
```

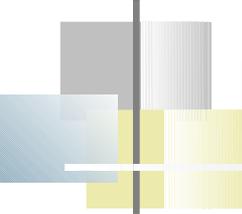


# Exemple de document XML (3)

---

- Données plus ou moins structurées

```
<?xml version="1.0"?>
<liste_personnel>
  <description>personnel 4ème étage</description>
  <personne bureau='U1'>
    <nom>Bond</nom>
    <mail>bond@service-secret.fr</mail>
    <age>35</age>
  </personne>
  <personne>
    <nom>Lupin</nom>
    <bureau>U2</bureau>
  </personne>
  <personne âge='32'>
    <nom>Templar</nom>
    <bureau>U3</bureau>
  </personne>
</liste_personnel>
```



# BD - Données semi-structurées

---

- Bases de données
  - Typage fort - Schéma imposé
  - Représentation - Manipulation

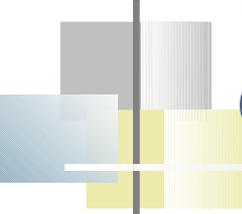
*BD structurées  
(relationnelles et objet)*

- Fédération de sources de données hétérogènes / Echange

- Vers un nouveau modèle de données semi-structurées

- Données auto-descriptives
  - Plus de description séparée de leur structure
- Données irrégulières
  - Typage fort relâché
    - Données manquantes, ou en plus
    - Variations de type: dollars, euros
    - Typage non obligatoire

*BD  
semistructurées*



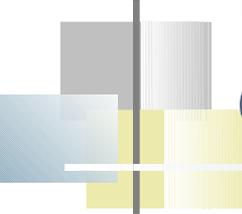
# Données semi-structurées : exemples

---

```
{nom : "toto", tel : "0491463728", email : toto@abc.com }
```

```
{nom : "toto", tel : "0491463728", tel : "06145637",  
email : toto@abc.com }
```

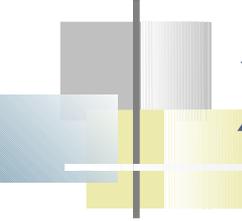
```
{personne :  
  {nom : "toto", tel : "0491463728",  
    email : toto@abc.com}  
  personne :  
    {nom : {prénom : "tutu", patronyme : « titi"},  
      tel : "0551461728", email : titi@abc.com}  
  personne :  
    {nom : "tata", tel : "0609046728", age : "35"}  
}
```



# Représentation en XML - convergence de modèles

---

```
<?xml version="1.0"?>
<personnes>
  <personne>
    <nom>toto</nom>
    <tel>04 91 46 37 28</tel>
    <email>toto@abc.com</email>
  </personne>
  <personne>
    <nom>
      <prenom>tutu</prenom>
      <patronyme>titi</patronyme>
    </nom>
    <tel>05 51 46 17 28</tel>
    <email>titi@abc.com</email>
  </personne>
  <personne>
    <nom>tata</nom>
    <tel>06 09 04 67 28</tel>
    <age>35</age>
  </personne>
</personnes>
```



# XML : documents vs données

---

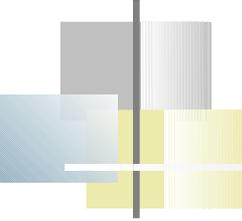
- Représentation standardisée d'informations
  - Plus ou moins structurées / Plus ou moins hétérogènes
  - Indépendante de toute plate forme, langue ou média
- Un format universel natif ou d'échange pour les documents et les données (semi) structurés sur le Web

## ■ Point de vue document

- Annoter la structure
- Stocker le document
- Diffuser : versions, formes, media différents
- Origine : auteur(s)

## ■ Point de vue donnée

- Le document : contenant
- Le document : stocké ou pas
- Origine : BD, programme ... ⇒ requête



# Bibliographie

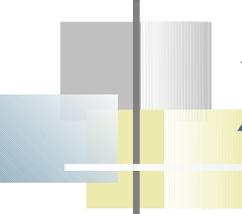
---

- W3 Consortium *<http://www.w3.org> (Note, Working draft, proposed recommandation, recommandation)*
  - Industriels: tous les poids lourds, comme Oracle, IBM, Compaq, Xerox, Microsoft, etc..
  - Laboratoires de recherche: MIT (représentant les US), INRIA (Europe), Japon (Asie)
  - ...
- Serge Abiteboul, Peter Buneman, Dan Suciu  
*Data on the Web : from relations to semistructured Data and XML*  
Morgan Kaufmann Publishers, 2000
- Jacques Le Maitre, *Description et manipulation de documents XML*, supports de cours  
<http://lemaitre.univ-tln.fr/cours.htm>
- Georges Gardarin, *XML : Des bases de données aux services Web*, Dunod, 2003  
et supports de cours XML <http://georges.gardarin.free.fr/>

XML :

# **Description** et manipulation

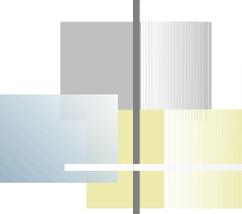
---



# XML

---

- **Présentation générale de XML**
- Documents bien formés
- Documents valides
- Espaces de noms

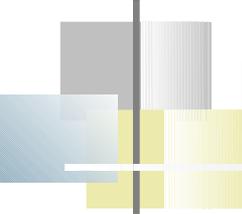


# Présentation générale de XML

---

## XML eXtensible Markup Language

- Première recommandation 10 février 1998
- 2 correctifs mineurs en octobre 2000 et en février 2004
- Une **syntaxe** simple
  - facile à mettre en oeuvre - adaptable à toute sorte d'application
- Texte UNICODE: pour n'importe quel alphabet
  - russe, hébreu, arabe, japonais, chinois...
- Séparation entre la description structurelle du document et tout traitement et en particulier de sa réalisation physique

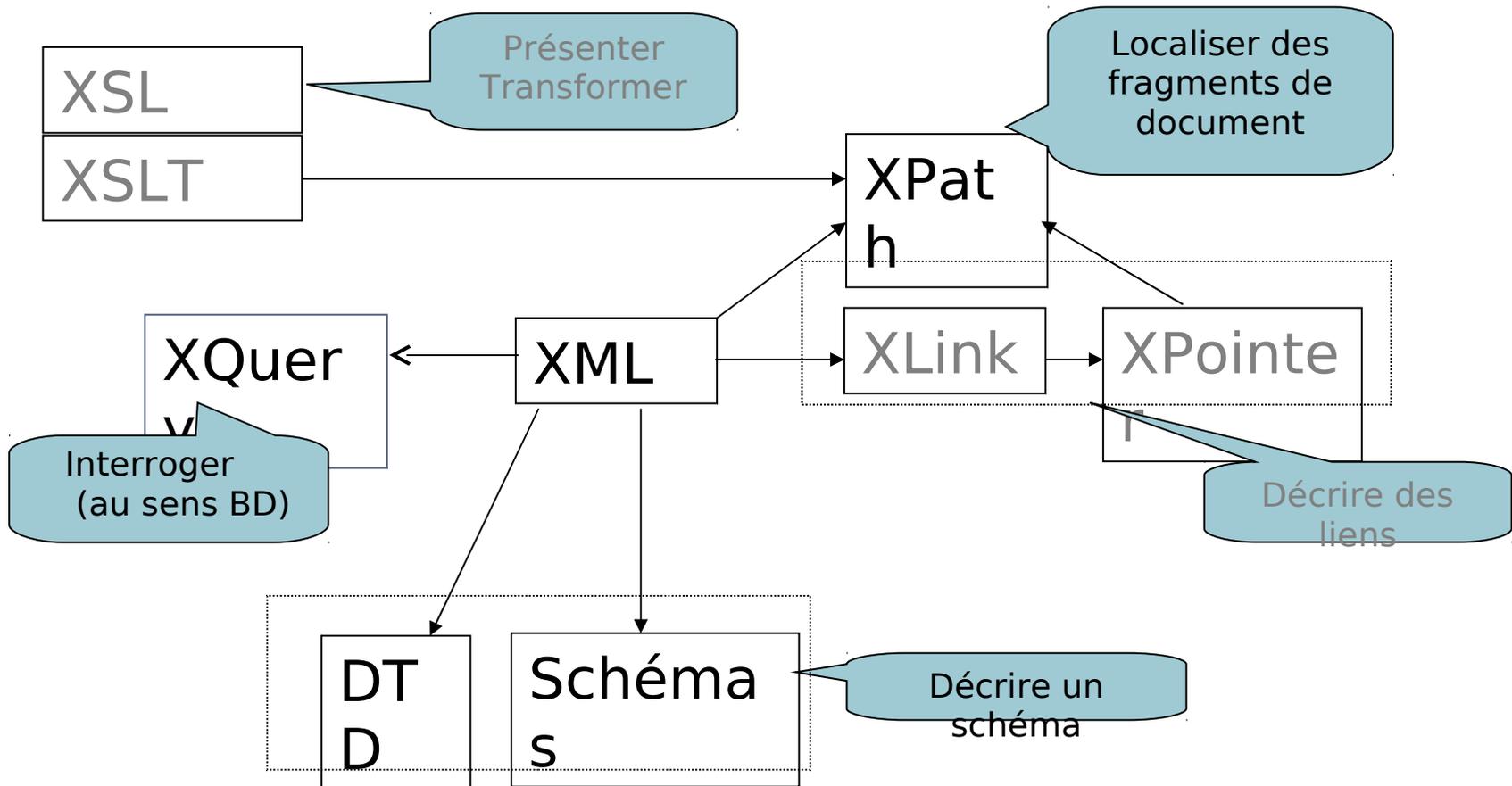


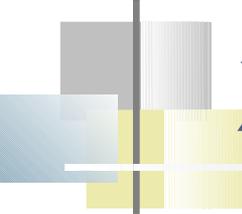
# Présentation générale de XML

---

- Baliser le document (contenu alpha numérique) pour mettre en évidence
  - sa structure (en général) **hiérarchique** : les éléments qui composent le document et leur organisation
- Le balisage est
  - libre, les contraintes sont d'ordre syntaxique et structurel
  - au sein même du texte...  
on parle de **structure auto-descriptive**
- La recommandation : base pour la description du document
- Mécanismes complémentaires : recommandations séparées

# Présentation générale de XML

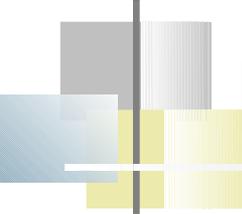




# XML

---

- Présentation générale de XML
- **Documents bien formés**
- Documents valides
- Espaces de noms



# Documents bien formés

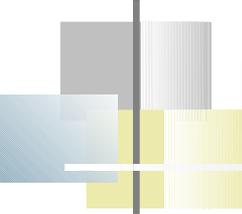
---

- La description du document respecte des règles syntaxiques
- Structure générale d'un document
  - Un prologue (optionnel mais conseillé)
    - version de XML utilisée pour décrire le document
    - le jeu/codage de caractères utilisé

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
```

- Optionnels : des commentaires, des instructions de traitement destinées aux applications traitant le document
- Un élément racine (obligatoire) appelé **élément du document**
  - Cet élément (comme tout élément) est composé d'un nom, d'attributs et d'un contenu. Il est marqué par des paires de balises ouvrante et fermante.

```
<balise a1=v1 ... an=vn>contenu</balise>
```



# Élément (1)

---

- Nom libre
  - Les balises ouvrante et fermante contiennent le nom de l'élément
- Un *caractère de nom*
  - une lettre, un chiffre, un point, un tiret, un espace souligné, soit deux-points
- Un *nom* est une suite de un ou plusieurs caractères de nom :
  - le premier est soit une lettre, soit un espace souligné, soit un deux-points (cf Espaces de noms)
  - les suivants sont des caractères de nom

Organisation hiérarchique d'éléments  
Les balises doivent s'imbriquer correctement, et non se recouvrir

## Élément (2)

- Contenu :

- Vide (document minimal)

```
<element></element> <element/>
```

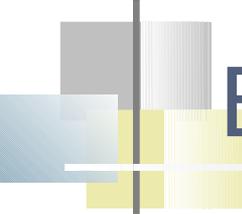
- Une suite de caractères (texte brut)

```
<element>Ceci est un exemple</element>
```

- Une liste d'éléments

```
<agent>  
  <nom>Bond</nom>  
  <age>35</age>  
  <mail>bond@service-secret.en</mail>  
</agent>
```

```
<section>  
  <paragraphe>...</paragraphe>  
  <paragraphe>...</paragraphe>...  
</section>
```



## Élément (3)

---

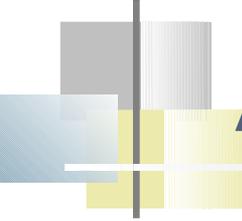
- Contenu mixte

- Une liste d'éléments et/ou de textes c'est-à-dire d'une suite de caractères

`<exemple>Le monde<sigle>XML</sigle>est riche (très riche)</exemple >`

L'élément `exemple` est composé dans l'ordre

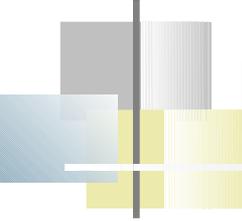
- d'un texte `Le monde`
- d'un élément `<sigle>XML</sigle>`
- d'un texte `est riche (très riche)`



# Attributs

---

- Des couples (nom, valeur)
  - Un nom libre (règles syntaxiques)
  - Une valeur obligatoire : une chaîne de caractères
- Pour qualifier un élément indépendamment de son contenu
- Pour définir des références (liens) internes
  - la structure sous-jacente d'arbre devient alors graphe

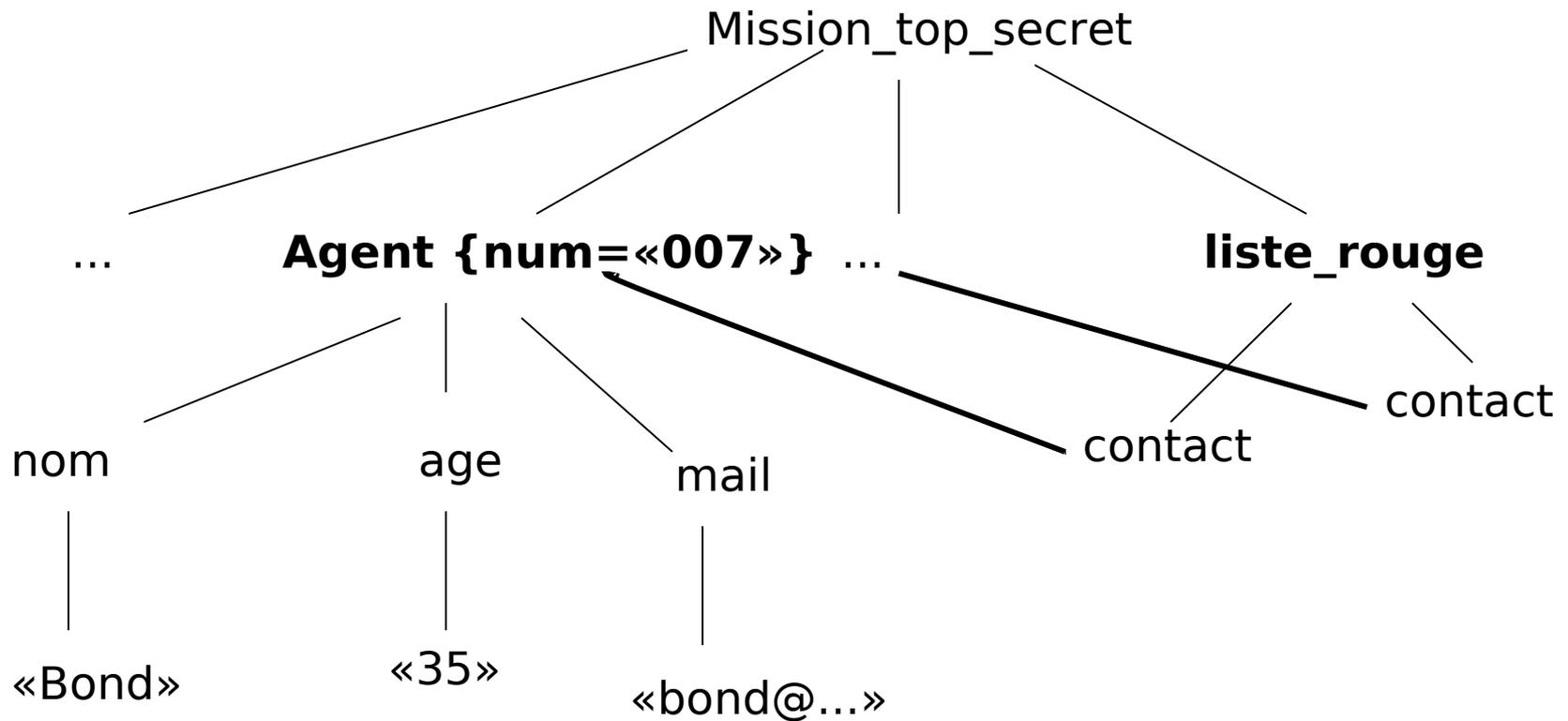


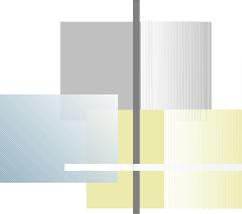
# Exemple

---

```
<?xml version="1.0"?>
<mission_top_secret>
  <agent num="007">
    <nom>Bond</nom>
    <age>35</age>
    <mail>bond@service-secret.en</mail>
  </agent>
  ...
  <agent num="009">
    <nom>...</nom>
    <age>...</age>
    <mail>...</mail>
  </agent>
  <liste_rouge>
    <contact ref="007"/>
    <contact ref="009"/> ...
  </liste_rouge>
</mission_top_secret>
```

# Représentation arborescente, de graphe (vision simplifiée)



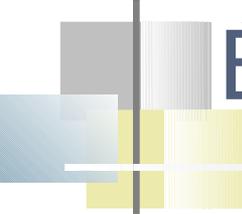


# Exemples

---

```
<produit>
  <nom langue="français">trompette six trous</nom>
  <prix cours="euro">420,12</prix>
</produit>
```

```
<divers>
  <personne id="P1" conjoint="P2">
    <nom>Bond</nom><prénom>James</prénom>
  </personne>
  <personne id="P2" conjoint="P1">
    <nom>Bond</nom><prénom>Manon</prénom>
  </personne>
  <publication auteurs="P1 P2">
    <titre>Les espions en famille</titre>
  </publication>
</divers>
```



# Éléments - Attributs

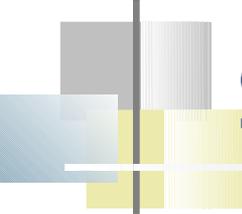
---

- Un attribut apparaît une unique fois au sein d'un élément
- Un même élément peut apparaître plusieurs fois dans un élément
- Les attributs associés à un élément ne sont pas ordonnés
- Les éléments sont ordonnés (ordre de lecture du document)
- Les attributs introduisent une ambiguïté dans la représentation de l'information : *Elément ? Attribut ?*

```
<agent num="007"><nom>Bond</nom><age>35</age></agent>
```

```
<agent num="007" nom="Bond" age="35"></agent>
```

```
<agent num="007" age="35"><nom>Bond</nom></agent>
```



# Section CDATA

---

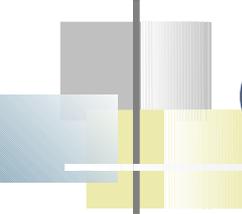
- Un texte brut...

```
<![CDATA[texte]]>
```

- Le texte peut contenir n'importe quels caractères sauf la chaîne ]]
- Une section CDATA ne peut donc pas en contenir une autre

« *Voici un exemple d'élément XML <titre>Le monde XML</titre> contenant une chaîne de caractères.* »

```
<exemple>Voici un exemple d'élément XML          <![  
  [CDATA[<titre>Le monde XML</titre>]]> contenant une chaîne  
  de caractères.</exemple>
```



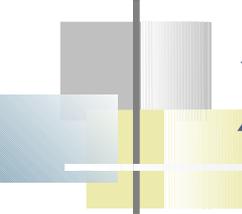
# Commentaires

---

```
<!-- commentaire-->
```

- Il peut contenir toute suite de caractères sauf --
- Un commentaire ne peut pas contenir un autre commentaire
- Il peut être inclus dans le contenu d'un élément (pas dans une balise)

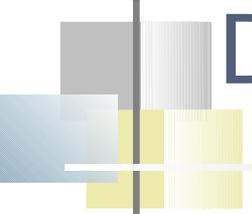
```
<?xml version="1.0"?>
<table>
  <!-- liste de personnes actives-->
  <personne>
    <nom>Bond</nom>
    <bureau>U1</bureau>
  </personne>
  ...
</table>
```



# XML

---

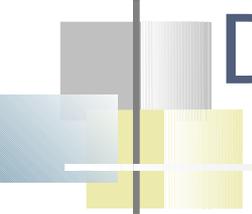
- Présentation générale de XML
- Documents bien formés
- **Documents valides**
- Espaces de noms



# Documents valides

---

- Les éléments qui décrivent un document peuvent être définis dans un DTD (Définition Type Document)
- Une DTD permet de définir des classes de documents
- Un document bien formé conforme à une DTD est dit **valide** par rapport à cette DTD



# Documents valides

---

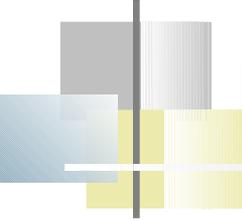
- Définir la structure et le vocabulaire utilisés dans un document XML
- Une grammaire régulière (Document Type Definition - DTD)
  - Une règle est associée à chaque type d'élément  
`<!ELEMENT nom_élément modèle_de_contenu>`
  - Une règle est associée à la définition de chacun de ses attributs  
`<!ATTLIST nom_élément  
                  nom_attribut type déclaration ...>`

# Document valide

- Le prologue contient une DTD
- Le document respecte la structure définie par la DTD
- Le nom de l'élément racine doit être le même que celui de la DTD (ici document)

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?  
>  
<!DOCTYPE document [  
  <!ELEMENT document (personne)>  
  <!ELEMENT personne (nom,prenom)>  
  <!ELEMENT nom (#PCDATA)>  
  <!ELEMENT prenom (#PCDATA)>  
>  
>  
<document>  
  <personne>  
    <nom>Murisasco</nom><prenom>Elisabeth</prenom>  
  </personne>  
</document>
```

Le document est  
contenu en entier  
dans le fichier



# Exemple

---

```
<!DOCTYPE livres [  
<!ELEMENT      livres      livre+                >  
<!ELEMENT      livre       (titre, auteur+, section+, biblio?) >  
<!ELEMENT      titre       (#PCDATA)              >  
<!ELEMENT      auteur      (#PCDATA)              >  
<!ELEMENT      section     (titre, para+)         >  
<!ELEMENT      para        #PCDATA                >  
<!ELEMENT      biblio      (reference)+          >  
<!ELEMENT      reference   #PCDATA                >  
>]
```

- Si la DTD n'apparaît pas après le prologue, elle est stockée ailleurs et on doit y faire référence

```
<!DOCTYPE nom SYSTEM "Uniform Resource Identifier">
```

```
<!DOCTYPE livres SYSTEM "../biblio.dtd">
```

`<!ELEMENT nom_élément modèle_de_contenu>`

# Définition des éléments

---

## ■ Modèles de contenu

- ( nom<sub>1</sub> , nom<sub>2</sub> ) séquence d'éléments
- ( nom<sub>1</sub> | nom<sub>2</sub> ) liste de choix
- ( nom ) ? élément optionnel
- ( nom ) \* élément répétitif
- ( nom ) + élément répétitif
  
- Un contenu mixte a pour modèle de contenu  
(#PCDATA | nom<sub>1</sub> | ... | nom<sub>n</sub>)\*

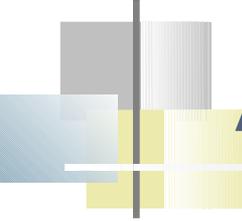
## ■ Types de données

- #PCDATA  
le contenu est du texte, à analyser
- CDATA  
contenu brut non analysé
- EMPTY  
contenu vide

```
<!ATTLIST nom_élément nom_attribut type déclaration ...>
```

# Définition des attributs

- **Type**
  - *CDATA*, sa valeur doit être une chaîne
  - *ID*, sa valeur doit être un nom qui identifie de façon unique l'élément qui le contient
  - *IDREF*, sa valeur est l'identificateur d'un élément du document
  - *IDREFS*, sa valeur est une liste d'identificateurs d'éléments du document
- **Déclaration**
  - *#REQUIRED* (la présence de l'attribut est obligatoire), *#IMPLIED* (l'attribut est optionnel)
  - *#FIXED* 'valeur' (valeur de l'attribut s'il est présent)
- Il existe des attributs prédéfinis
- Les noms d'attributs sont locaux à chaque type d'élément
  - 2 éléments de type différents peuvent avoir des attributs de même nom



# Autre exemple

```
<! DOCTYPE famille [  
  <!ELEMENT famille (personne*)  
  >  
  <!ELEMENT personne (nom)>  
  <!ELEMENT nom (#PCDATA)>  
  <!ATTLIST personne  
    num ID #REQUIRED  
    mere IDREF #IMPLIED  
    pere IDREF #IMPLIED  
    enfants IDREFS #IMPLIED >]>
```

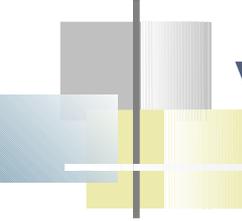
```
<famille>  
  <personne num="Jane"  
    mere="Mary" pere="John">  
    <nom>Jane Doe</nom>  
  </personne>  
  <personne num="John"  
    enfants="Jane Jack">  
    <nom>John Doe</nom>  
  </personne>  
  <personne num="Mary"  
    enfants="Jane Jack">  
    <nom>Mary Doe</nom>  
  </personne>  
  <personne num="Jack"  
    mere="Mary" pere="John">  
    <nom>Jack Doe</nom>  
  </personne>  
</famille>
```

# Documents bien formés

## Documents valides

---

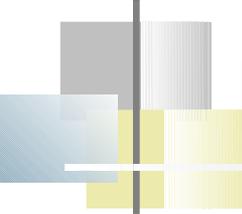
- **Un document XML peut être bien formé**
  - il n'est pas précédé de sa DTD
  - sa description est syntaxiquement correcte
- **Un document XML peut être valide**
  - il est précédé de sa DTD
  - il est conforme à cette DTD



# Validité d'un document

---

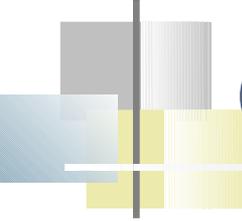
- Le document est validé syntaxiquement puis
- par rapport à sa DTD
  - ID/IDREF
    - Les valeurs de tous les attributs de type ID doivent être distinctes
    - Toute référence IDREF ou IDREFS doit référencer des identifiants existants
  - L'ordre des éléments imposé par la DTD est vérifié



# Limites des DTD

---

- Le type de IDREF ou IDREFS n'est pas contraint
  - la DTD n'impose pas de contrainte sur le type d'objet référencé
- L'ordre est imposé
- Pas de types atomiques
  - pas de contraintes sur les valeurs
- Le type associé à un élément est global :
  - utiliser un même nom de balise pour deux éléments de modèles de contenu différents est impossible
- Le langage utilisé est différent de celui des instances



# Ordre

---

BD        r1 (a,b,c)  
          r2 (c,d)

```
<db>
  <r1><a>a1</a><b>b1</b><c>c1</c></r1>
  <r1><a>a1</a><b>b1</b><c>c1</c></r1>
  <r2><c>c2</c><d>d2</d></r2>
  <r2><c>c3</c><d>d3</d></r2>
  <r2><c>c4</c><d>d4</d></r2>
</db>
```

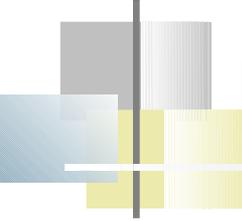
```
<! DOCTYPE bd [
  <!ELEMENT bd (r1*, r2*)>
  <!ELEMENT r1 (a, b, c)>
  <!ELEMENT r2 (c, d)>
  <!ELEMENT a (#PCDATA)>
  <!ELEMENT b (#PCDATA)>
  <!ELEMENT c (#PCDATA)>
  <!ELEMENT d (#PCDATA)>]>
```

Pour r2 : `<!ELEMENT r2 ((c, d) | (d, c))>`

Pour r1 : 6 possibilités

Dans la règle *bd*, r1 doit apparaître avant r2

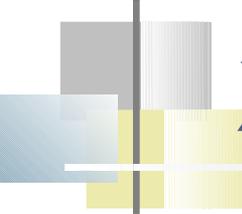
```
<!ELEMENT bd ((r1 | r2)*)>
```



# Limites des DTD

---

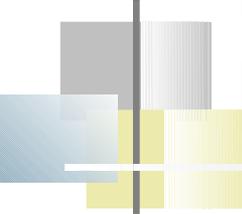
- Il existe des solutions dans des propositions complémentaires
  - Namespace : espace de noms (garantir l'unicité des noms)  
*Recommandation 14/01/1999*  
<http://www.w3.org/TR/REC-xml-names/>
  - Schémas XML <http://www.w3.org/XML/Schema>  
*Recommandation XML schema part 0 : Primer 02/05/01*  
*Recommandation XML schema part 1 : Structures 02/05/01*  
*Recommandation XML schema part 2 : Datatypes 02/05/01*



# XML

---

- Présentation générale de XML
- Documents bien formés
- Documents valides
- **Espaces de noms**



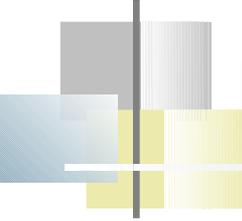
# Espaces de noms - XML NameSpace

---

- Utiliser dans un même document des ensembles d'éléments et d'attributs provenant d'origines diverses ?
  - Composer un document valide en utilisant des DTD existantes, publiques, validées
  - Composer un document bien formé par copier/coller d'autres documents, valider le document résultat

## → **Des conflits de noms sont possibles**

- Les espaces de noms garantissent l'unicité des noms d'éléments et d'attributs



# Espaces de noms : un exemple

---

<exemple

```
xmlns:ex="divers/personnes1.dtd"
```

```
xmlns:cr="http://www.univ-tln.fr/reserve/personnes2.dtd" >
```

```
<ex:personne ex:age='35'>
```

```
  <ex:nom>Dupont</ex:nom>
```

```
  <ex:prenom>Jacques</ex:prenom>
```

```
</ex:personne>
```

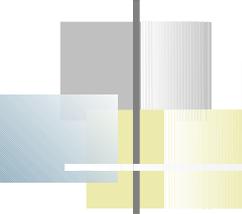
```
<cr:personne>
```

```
  <cr:nom>Dupont Jacques</cr:nom>
```

```
  <cr:age>plus de trente ans</cr:age>
```

```
</cr:personne>
```

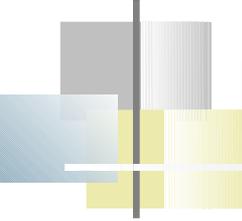
</exemple>



# Espace de noms

---

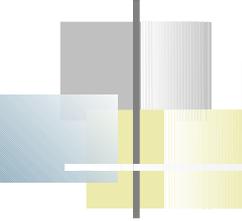
- Un espace de noms est
  - un ensemble de noms utilisés dans des documents pour les éléments et les attributs
  - identifié par une URI (*Uniform Resource Identifier*)
    - Cette URI n'a pas à donner accès à une ressource réelle sur le Web
    - Elle est simplement utilisée en tant qu'identifiant



# Espace de noms

---

- La combinaison d'un nom d'élément ou d'attribut `nom` avec l'URI d'un espace de noms `espace` garantit l'unicité de ce nom
  - on parle de **nom étendu**
  - `espace` est le nom de l'espace de nom
  - `nom` est le **nom local**
- Pour simplifier la syntaxe, on utilise un **préfixe** que l'on associe à l'URI de l'espace de noms plutôt que l'URI elle-même
  - on parle de **nom qualifié**



# Espace de noms : déclaration

---

<exemple

```
xmlns:ex="divers/personnes1.dtd"
```

```
xmlns:cr="http://www.univ-tln.fr/reserve/personnes2.dtd" >
```

- La déclaration d'un namespace est à l'intérieur d'un élément
- On lui associe un préfixe (il peut être réservé, usuel ou librement choisi)
- Ce préfixe est déclaré par l'attribut prédéfini **xmlns**

```
xmlns:prefixe="uri de l'espace de noms"
```

- Le préfixe permet de faire référence à la ressource associée et d'identifier un ensemble de noms d'éléments et d'attributs
- La déclaration de l'espace de nom est visible dans l'élément le contenant et dans tous ses descendants sauf si une autre déclaration l'annule ou s'il est annihilé (`xmlns=""`)

Un processeur XML qui supporte les espaces de noms remplace les préfixes par l'URI associée

# Espace de noms : utilisation

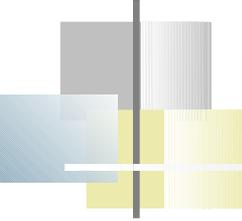
Le préfixe est placé avant le nom d'élément ou d'attribut extrait de la ressource associée

`prefixe:nom_local`

<exemple

```
xmlns:ex="divers/personnes1.dtd"
xmlns:cr="http://www.univ-tln.fr/reserve/personnes2.dtd" >
<ex:personne ex:age='35'>
  <ex:nom>Dupont</ex:nom>
  <ex:prenom>Jacques</ex:prenom>
</ex:personne>
<cr:personne>
  <cr:nom>Dupont Jacques</cr:nom>
  <cr:age>plus de trente ans</cr:age>
</cr:personne>
```

</exemple>



# Espace de noms par défaut

---

<exemple

```
xmlns:ex="divers/personnes1.dtd"
```

```
xmlns:cr="http://www.univ-tln.fr/reserve/personnes2.dtd"
```

```
xmlns:="cv.dtd" >
```

```
<ex:personne ex:age='35'>
```

```
  <ex:nom>Dupont</ex:nom>
```

```
  <ex:prenom>Jacques</ex:prenom>
```

```
</ex:personne>
```

```
<formation>M2PRO Informatique</formation>
```

```
<job>Administrateur BDR</job>
```

```
<cr:personne>
```

```
  <cr:nom>Dupont Jacques</cr:nom>
```

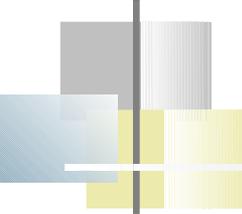
```
  <cr:age>plus de trente ans</cr:age>
```

```
</cr:personne>
```

```
  <formation>MSG</formation>
```

```
  <job>Ingénieur qualité</job>
```

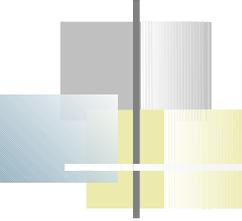
</exemple>



# Préfixes réservés et usuels

---

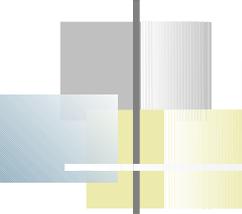
- Le préfixe `xmlns` est lié à l'espace de noms  
<http://www.w3.org/2000/xmlns/>
- Les préfixes commençant par les lettres `xml` sont réservés
- Chaque langage de définition ou de manipulation de données XML possède son propre espace de noms ainsi que son préfixe par défaut
  - il est recommandé de les utiliser



# Edition d'un document XML

---

- Éditeurs de texte classiques
- Éditeurs syntaxiques qui permettent d'éditer des documents XML bien formés (Emacs)
- Analyseurs (parseurs) qui vérifient
  - la validité d'une DTD
  - la validité d'un document par rapport à une DTD
- Analyseurs validants qui combinent les fonctionnalités d'éditeurs syntaxiques et des analyseurs



# Bilan XML

---

- Un mode de structuration de données
  - Un format de document
  - Un format de données
  - Un format d'échange et de publication
- Chaque utilisateur
  - est libre de définir ses propres structures de document
  - peut se conformer à des structures types
- Chaque communauté peut donc proposer des structures normalisées
- La conformité à une DTD permet l'automatisation des traitements et assure une possibilité de contrôle de validité

Un processeur XML qui supporte les espaces de noms remplace les préfixes par l'URI associée