INU3011 Documents structurés

Cours 11

Espaces de noms, Validité par schéma, Modèles normalisés, Aperçu de la TEI

Plan

- Logistique : conférence d'Edith Cannet semaine prochaine, soyez-y !
- Espaces de noms (*Namespaces*)
  - Attributs prédéfinis (`xml:lang`, etc.)
- Validité par schéma (W3C, RelaxNG, Schematron)
- Modèles normalisés
  - TEI : un aperçu

Espaces de noms XML

XML Namespaces

Alias : espaces "nominatifs", "nominaux", "de nommage"

Notes préalables

- Les espaces de noms XML sont principalement conçus pour marcher avec les documents *valides par schémas* (W3C, Relax NG, etc.)
- Possible, mais très lourd avec les DTD
- Ont été développés pour faciliter la *réutilisation de modèles XML* (syntaxe & sémantique)

Problématique (1/3)

- Modéliser représente un effort important
- Il est intéressant de pouvoir réutiliser un modèle, ou une partie de modèle; ex.:
  - Ici dans mon modèle, je voudrais avoir un élément `title`, **tel que défini dans DOCBOOK**
- Comment établir ce lien avec DOCBOOK ?

Problématique (2/3)

- Exemple :

```
<note>
<title>Résumé de lecture</title>
J'ai lu <title>1984</title>
<!-- Ce "title" est comme défini en DOCBOOK -->
et j'ai bien aimé ça.
</note>
```


Problématique (3/3)

Pour mémoire :

```
<title>Vers un cadre
<emphasis>unificateur</emphasis> pour
l'espionnage
  <footnote>
    <para>Au sens britannique du
terme.</para>
  </footnote></title>
```

… est un "title" valide en DOCBOOK !

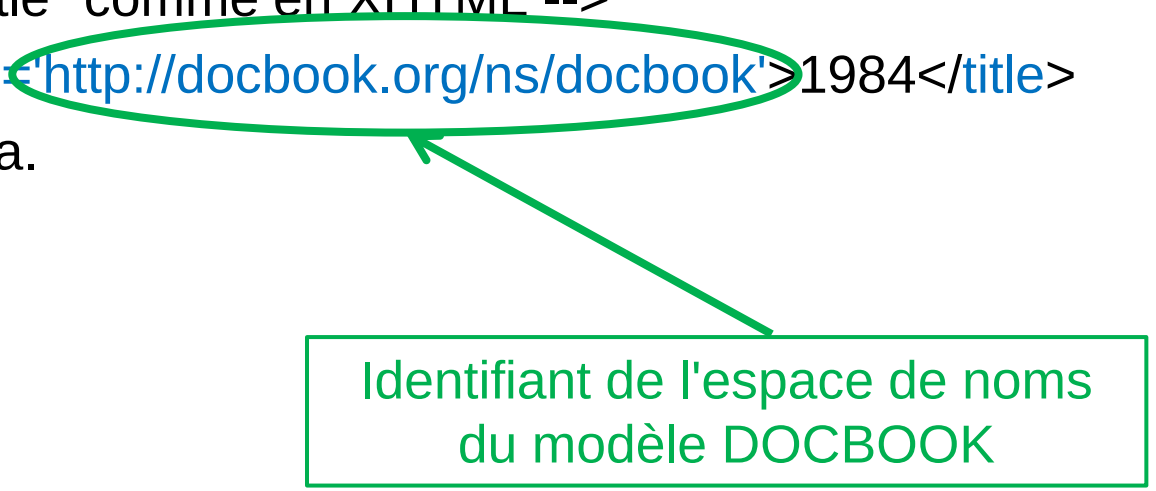
Espaces de noms (1/3)

- C’est ce que les espaces de noms (*namespaces*) XML du W3C permettent
  - Namespaces in XML 1.0 (Third Edition) W3C Recommendation 8 December 2009
- Chaque modèle possède son *namespace* (espace de noms), qu'on identifie par un identifiant, qui est planétairement unique
  - Ces identifiants ont la *forme* d'une URL / URI, mais n'en sont pas réellement (voir plus loin)

Espaces de noms (2/3)

- Notre exemple :

```
<note>
<title>Résumé de lecture</title>
<!-- ceci est un "title" comme en XHTML -->
J'ai lu <title xmlns='http://docbook.org/ns/docbook'>1984</title>
et j'ai bien aimé ça.
</note>
```



Espaces de noms (3/3)

- N.B. : Élimine le potentiel conflit de nom

```
<note>
<title>Résumé de lecture</title>
<!-- ceci est un "title" comme en XHTML -->
J'ai lu <title xmlns='http://docbook.org/ns/docbook'>1984</title>
<!-- ceci est un "title" comme en DOCBOOK -->
et j'ai bien aimé ça.
</note>
```

L'attribut `xmlns` (1/3)

- Déclare un *namespace* par défaut, qui s'applique à tous les descendants de l'élément :

```
<mods xmlns='http://www.loc.gov/mods/v3'>
  <titleInfo>
    <title>1984</title>
  </titleInfo>
</mods>
```

- Ici, tous les éléments (y compris `mods`) sont dans l'espace de noms `http://www.loc.gov/mods/v3`

L'attribut `xmlns` (2/3)

- Le *namespace* par défaut peut être occulté localement* :

```
<note xmlns='http://www.loc.gov/mods/v3'>
    <title xmlns='http://docbook.org/ns/docbook'>1984</title>
    <copyrightDate>1999</copyrightDate>
</note>
```

- Ici, `note` et `copyrightDate` sont dans `http://www.loc.gov/mods/v3`, mais `title` est dans `http://docbook.org/ns/docbook`

*Une telle occultation locale se rencontre assez rarement en pratique

L'attribut `xmlns` (3/3)

- Si un modèle possède un espace de noms (p.ex. DOCBOOK), la référence à l'espace de noms, dans les documents conformes à ce modèle, se résume souvent à l'attribut xmlns sur l'EPHN; ex.:

```
<article xmlns="http://docbook.org/ns/docbook">
  <title>...</title>
  ...
</article>
```

- La totalité du document est dans l'espace de noms du modèle

Exemples

TEI, MODS et DOCBOOK
dans
300-Ex-espaces-de-noms

Préfixes d'espace de noms (1/2)

- On peut associer un préfixe à un *namespace* avec `xml:`*`préfixe`*; p.ex.:

```
<fiche xmlns:mo='http://www.loc.gov/mods/v3'>
```

- À l'intérieur de cet élément, tous les éléments nomm[illegible]éfixe sont associés au *nam*[illegible]estion. Ex.:

Déclaration du préfixe mo: associé au *namespace*

```
<fiche xmlns:mo='http://www.loc.gov/mods/v3'>
  <auteur>Jean Dupont</auteur>
  <mo:titleInfo>
    <mo:title>Guerre et paix</mo:title>
  </mo:titleInfo>
</fiche>
```


Préfixes d'espace de noms (2/2)

- Utilisé notamment en XSLT :

```
<xsl:stylesheet version="1.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
   <xsl:template ...>
     ...
   </xsl:template>
...
</xsl:stylesheet>
```

- Permet d'avoir à l'intérieur des gabarits des éléments *sans namespace*, notamment des éléments HTML5

Identifiants d’espace de noms

- Les identifiants d’espace de noms adoptent la forme d’URI (Uniform Resource Identifiers)
  - Il y a deux sortes d’URI:
    - Les URL (Uniform Resource Locators)

      Ex.: http://docbook.org/ns/docbook

      Pointent habituellement à "quelque chose sur le Web", mais ce n'est pas leur fonction première
    - Les URN (Uniform Resource Names)

      Ex.: urn:un-projet:avec-un-nom-unique:enfin-on-espere

Lien *namespace* ↔ schéma

- Il est possible de lier un document à un schéma qui n'est pas une DTD (équivalent de <!DOCTYPE… avec une DTD)
- Mais souvent, l'appartenance à un *namespace* donné est suffisant pour qu'un outil XML (p.ex. oXygen) sache automatiquement quel schéma utiliser pour valider le document

Exemples
TEI, DOCBOOK
et autres

dans

300-Ex-espaces-de-noms

"Hors espace de noms"

- Si un élément n'a pas de préfixe, et si aucun de ses ancêtres n'a d'attribut `xmlns=`, alors l'élément n'est dans aucun espace de noms
  - On dit aussi parfois (rarement) qu'il est dans "l'espace de noms global"
- C'est ce qu'on a fait depuis le début du trimestre, et qu'on va continuer à faire pour nos documents

Le préfixe `xml:` (1/3)

- Les *noms d'attribut* peuvent aussi être dans un *namespace*
- Le préfixe **`xml:`** est réservé et associé en permanence à l'espace de noms

  http://www.w3.org/XML/1998/namespace
- Certains attributs de cet espace de noms ont une signification spéciale :

  `xml:id`, **`xml:lang`**, **`xml:space`**, **`xml:base`**

Le préfixe `xml:` (2/3)

- xml:id
  - Permet de donner un id *unique* (dans le document courant) à n'importe quel élément
- xml:lang
  - Indique la langue *naturelle* associée à l'élément
- xml:space
  - valeurs possibles: `default` et `preserve`
  - Indique comment les blancs (espaces, tabulations, sauts de ligne) *devraient* être traitées par les applications (parfois sans effet)

Le préfixe `xml:` (3/3)

- Ces attributs avec préfixe `xml:` peuvent être utilisés, *sans être déclarés*, sur n'importe quel élément dans les documents schéma-valides
- Dans les documents DTD-valides, ils doivent être déclarés dans la DTD comme n'importe quel autre attribut, *si on veut en permettre l'utilisation* (ce qui n'est absolument pas nécessaire)

Extraits de la norme XML (1/3)

- Les noms commençant par "xml" ou par n'importe quelle chaîne de la forme :

  `(('X'|'x') ('M'|'m') ('L'|'l'))`

  sont réservés (pas interdits, mais risqué)
- Le ":" ne doit pas être utilisé dans un nom XML, sauf en lien avec les espaces de noms
- Ces restrictions s'appliquent aux noms…
  - d'éléments (incluant les préfixes d'espaces de noms)
  - d'attributs (incluant les préfixes d'espaces de noms)
  - d'entités déclarées (pas vues dans Tour d'horizon)

Extraits de la norme XML (2/3)

- L'espace de noms `http://www.w3.org/XML/1998/namespace` :
  - Ne peut être déclaré comme espace par défaut, ni lié à un préfixe autre que *xml:*
  - *xml:* ne peut être lié à aucun autre espace de noms
- Le nom d'attribut `xmlns` ne peut servir qu'à déclarer un espace par défaut

Extraits de la norme XML (3/3)

- Le préfixe *xmlns:* ne peut servir qu'à associer des préfixes à des espaces de noms
- Il est lié en permanence à l'espace `http://www.w3.org/2000/xmlns/`
- Il ne doit pas être déclaré
- L'espace en question ne peut pas être déclaré comme espace par défaut, ni lié à un autre préfixe

Validité par schéma (W3C, RelaxNG, Schematron)

Validité par schéma (1/2)

- Les DTD expriment des *contraintes de validité* pour un type de documents (au-delà du bien-formé)
  - C'est le seul mécanisme intégré à XML même
- D'autres formalismes existent
  - On les appelle de façon générique "schémas"
  - Ils ne sont ***pas*** intégrés à la norme XML elle-même; ils sont définis séparément

Validité par schéma (2/2)

- Un se distingue par sa popularité:
  - XML Schema (Second Edition)
    W3C Recommendation 28 October 2004
  - Appelés couramment "schémas XML" ou "schémas XML du W3C" pour être précis
- On distingue "validité par DTD" (*DTD-valide*) et "validité par schéma" (*schéma-valide*)
  - Sous-entend habituellement *schéma du W3C*

Schémas du W3C (1/2)

- La syntaxe des schémas est beaucoup plus verbeuse que celle des DTD
- *Beaucoup* plus complexe que les DTD comme formalisme
- Ajouts fondamentaux (p/r aux DTD) :
  - Validation des contenus textuels (plutôt que le simple #PCDATA des DTD)
  - Meilleure intégration avec les *espaces de nom*

Schémas du W3C (2/2)

- Autres ajouts importants:
  - Séparation type / nom d'élément
  - Éléments "locaux" (plusieurs modèles de contenu possibles pour le même nom d'élément, dépendant du contexte)
- Adoption massive (souvent partielle) malgré certaines critiques sévères

MODS

- *Metadata Object Description Schema*
- Développé depuis 2002 par la *Library of Congress* états-unienne
- Pour information bibliographique
  - Dublin Core < MODS < MARCXML
- Utilisé par exemple par les *Amherst College Digital Collections* (ACDC)
- Schéma MODS 3.8 (MODS)

Exemples

400-Ex-schemas-W3C

Relax NG

- Langage de contraintes très populaire
- Relax NG est spécifié par OASIS et fait partie de la norme ISO/IEC *Document Schema Definition Languages* (DSDL)
- Des schémas Relax NG existent pour plusieurs modèles normalisés, notamment JATS, DOCBOOK et la TEI

Schematron

- Permet de formuler des contraintes impossibles à exprimer dans les autres langages de contraintes, p.ex.:
  - Si tel attribut est spécifié, alors tel autre attribut est interdit
- Habituellement utilisé *conjointement* avec un autre langage de contraintes (DTD, Schémas W3C, RelaxNG, etc.)
- Fait aussi partie de DSDL

Modèles XML normalisés

Méthodologie de mise sur pied d'un système (d'information)

1. Discover (découverte: faisabilité, étude des besoins)
2. Design (conception)
3. Develop (développement)
4. Deploy (déploiement, implantation)
5. *Opération du système*
6. *Évaluation du système*

Plusieurs genres de systèmes / projets XML

- Création d'un corpus de textes électroniques (humanités numériques)
- Diffusion de ressources documentaires (e.g. revues savantes, fonds d'archives)
- La compatibilité, l'interopérabilité et la pérennité des données / documents sont habituellement des préoccupations importantes

Problématique

- La question du choix d'un modèle se pose donc *peu importe le genre de système* :
  - Modèle sur mesure (ce que vous faites dans le cours) : répond exactement aux besoins
  - Un modèle préexistant, voire *normalisé* est parfois une possibilité
    - N.B.: Un impératif d'interopérabilité n'impose *jamais* un choix de modèle, car l'interopérabilité peut aussi être atteinte par import/export

Quelques modèles XML « documentaires » normalisés

- TEI (Text Encoding Initiative) https://www.tei-c.org/
- DocBook https://www.docbook.org/
- EAD (Encoded Archival Description) https://www.loc.gov/ead/
- Journal Article Tag Suite (JATS) https://dtd.nlm.nih.gov/
- MARC21 https://www.loc.gov/standards/marcxml/

Rappel du C2

Modèle normalisé : c'est quoi ?

- DTD (rare) ou schéma dans un autre langage de contraintes (souvent)
- Souvent le modèle peut être récupéré dans différents langages de contraintes
  - Ex.: On peut travailler en TEI avec une DTD, un schéma XML du W3C, une grammaire Relax NG, etc.
  - Une des versions est habituellement identifiée comme "normative" (ex.: Schéma W3C pour MODS)

Text Encoding Initiative (1/4)

- Historique
  - Créé en 1987
  - Humanités, littérature, linguistique
  - D'abord SGML, maintenant XML
  - La TEI en ligne
- Documentation: « Guidelines »
  - Plusieurs versions; P5 est la courante
  - Très modulaire
  - Très extensible (ex.: correspondance)

TEI (2/4)

- L’entête (« TEI header »)
  - Les "métadonnées" du document
  - Connu indépendamment du reste de la TEI
  - Très élaboré, détaillé
- Toute la TEI est extrêmement flexible et peu contraignante
  - C’est à chaque projet d’édicter ses règles
  - Spécialisation et extension via un « ODD » (*One Document Does it All*)

TEI (3/4)

- D’abord conçu pour la transcription (l’encodage numérique) de textes non numériques
- Peut cependant accommoder très bien - et est beaucoup utilisé pour - les documents nés-numériques
- La plupart des projets TEI utilisent la validité par schéma W3C ou Relax NG

"Écosystème" d'un modèle XML normalisé

Rappel du C4

- Ressources partagées :
  - feuilles de style, documents-types, documentation, ressources pédagogiques
  - textes d'aide à la saisie
  - autres applications de traitement spécialisées (visualisation, statistiques, …)
- Réseaux d'entraide pour auteurs (encodeurs)

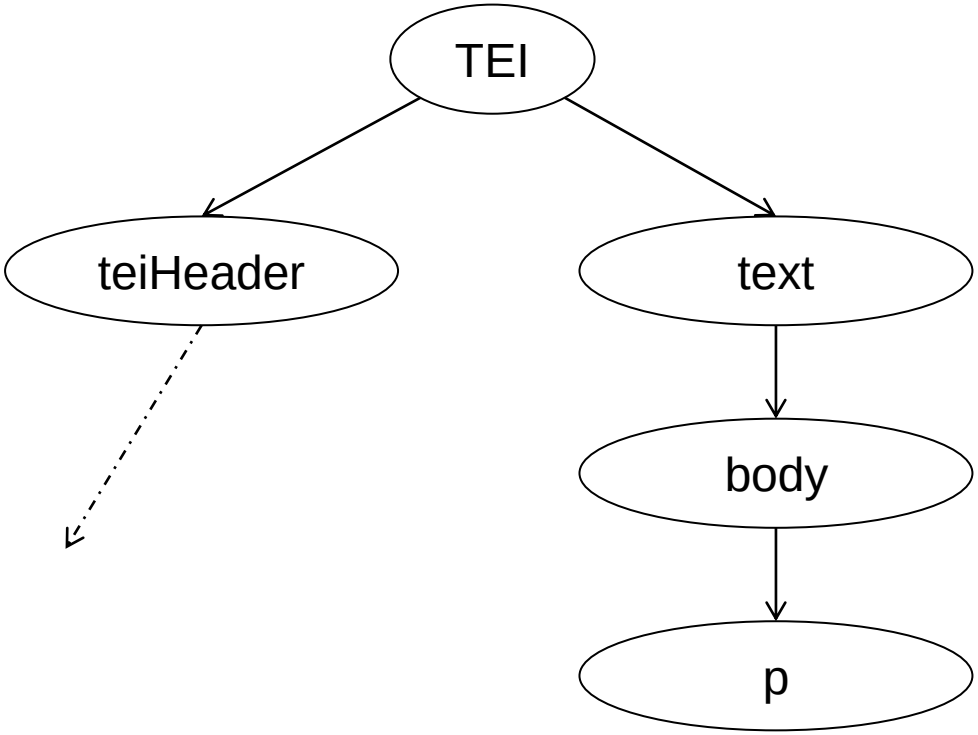
TEI (4/4)

- Quelques ressources de l'*écosystème* TEI (à part les Guidelines et le site Web)
  - Feuilles de style standard
    - CSS et XSLT (incluses dans oXygen et plusieurs autres outils)
  - Réseaux d'entraide pour auteurs (encodeurs)
    - TEI avec la liste tei-fr
    - Conférences, formations diverses, etc.
  - Projet TEI by Example

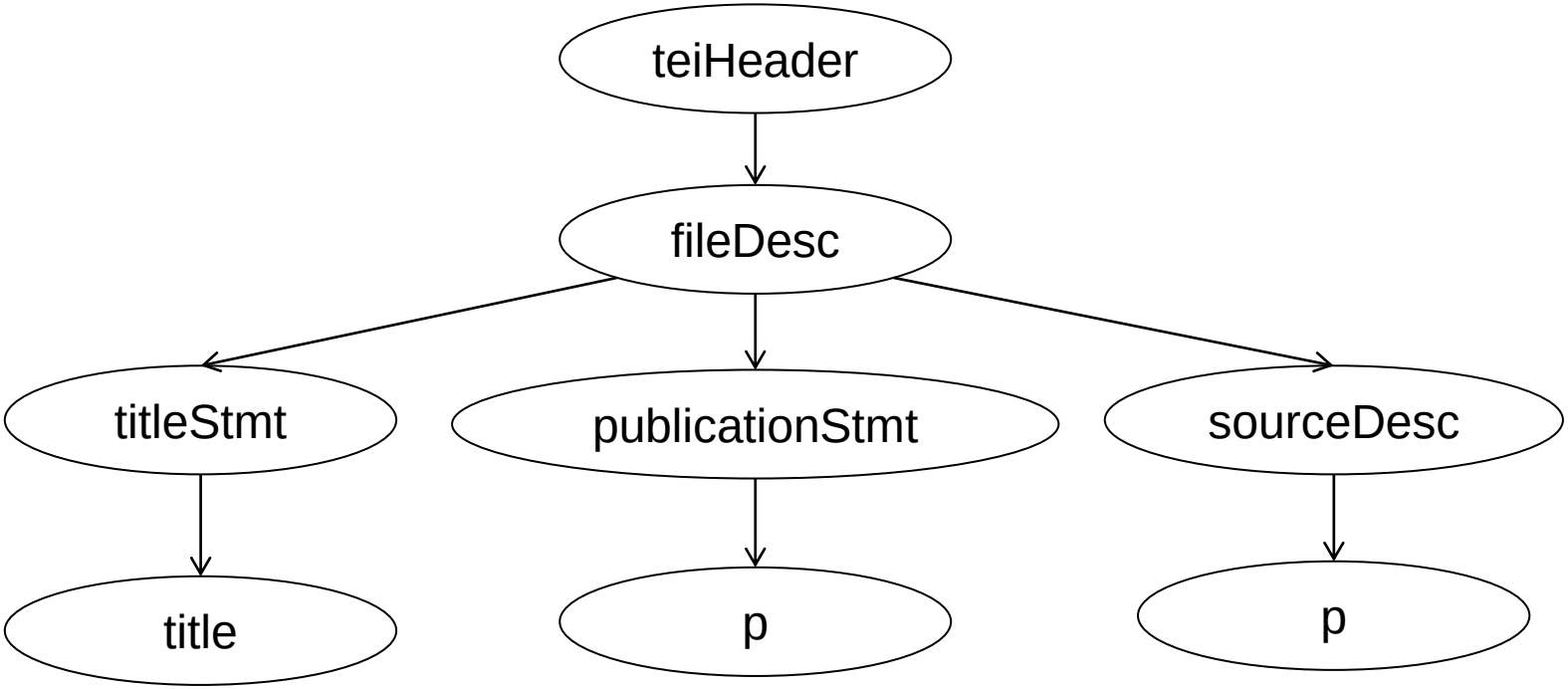
TEI Lite (légère)

- Une des "saveurs" de la TEI, qu'on obtient en choisissant certains modules de la TEI et pas d'autres
- TEI Lite correspond à un choix de modules qui convient à plusieurs types de documents généraux (un peu comme le HTML)

Structure minimale d'un document TEI



Zoom sur teiHeader



Exemple 500-Ex-TEI

Démo dans oXygen

Contrainte Schematron

Modèle normalisé : 

- Facilite le partage d'information avec une collectivité de partenaires
- Réseautage et partage d'expériences sur les méthodologies
- Bassin de professionnels compétents avec le modèle
- "Écosystème" de ressources communes…

Modèle normalisé : 👎 (1/2)

- En général beaucoup plus complexe et lourd qu'un modèle sur mesure
- Parfois un "surarmement" par rapport aux besoins réels (complexité inutile)
- Évolution du modèle lourde et politiquement chargée

Modèle normalisé : 👎 (2/2)

- Sémantique souvent "molle", car devant s'adapter à divers contextes
  - Représente souvent des compromis entre les besoins de divers partenaires
- Il faut compenser par des protocoles de rédaction "locaux" propres à notre projet
  - Danger de "distance sémantique" entre documents et information, qui peut mener à des erreurs de saisie ou d'interprétation

Exemple

- Documenter un jeu vidéo avec DocBook
  - On veut taguer les noms des personnages
  - Choix possible : élément `personname`
  - Il faut alors documenter ce choix dans une documentation locale, qui complète le "DocBook 5.1: The Definitive Guide"
  - Probablement "overkill", car peut contenir :

    firstname, honorific, lineage, othername, surname, givenname, et plus !

Modèle normalisé : conclusion

- Il vaut *parfois* mieux développer son propre modèle (ou sa propre extension d'un modèle normalisé), qui colle parfaitement aux besoins, et viser une *interopérabilité* par exportation / importation avec le modèle normalisé (sans extension)

Caractéristiques souhaitables d'un modèle normalisé

- Modulaire :
  - On prend juste ce qu'on veut
  - Ex.: Personnalisation de la TEI
- Extensibilité locale :
  - On ajoute ce dont on a besoin
  - Ex.: Éléments / attributs d'autres *espaces de noms* peuvent être ajoutés à certains endroits dans le modèle

StudiUM

