Information System

< XML technologies>

[01] XML Motivation & DTD

Joël Dumoulin
joel.dumoulin@hefr.ch
+41 26 429 69 60

Jacky Casas, Leonardo Angelini, Omar Abou Khaled
jacky.casas@hefr.ch, leonardo.angelini@hefr.ch, omar.aboukhaled@hefr.ch

HumanTech
Technology for
Human Wellbeing Institute

Buts et plan

- Buts :
  - Partie 1
    - ✓ Comprendre les origines et l'impact des technologies XML
    - ✓ Présentation des domaines applicatifs des technologies XML
    - ✓ Comprendre les formats de documents
    - ✓ Comprendre les modèles de productions des documents
    - ✓ Les types de documents
      - Linéaires vs structurés
    - ✓ Qu'est ce qu'un document ?
    - ✓ Qu'est ce qu'un modèle de document ?
    - ✓ Qu'est ce qu'un document XML ?

Buts et plan

- Buts :
  - Partie 2
    - ✓ Comprendre les principes de base d'un document structuré
      - Contenu, structure, et présentation
    - ✓ Modéliser une DTD
      - Les règles de construction
      - Structure et syntaxe
    - ✓ Construire un fichier XML
      - Structure et syntaxe
      - "well-formed" vs. valide

PARTIE 1 : XML

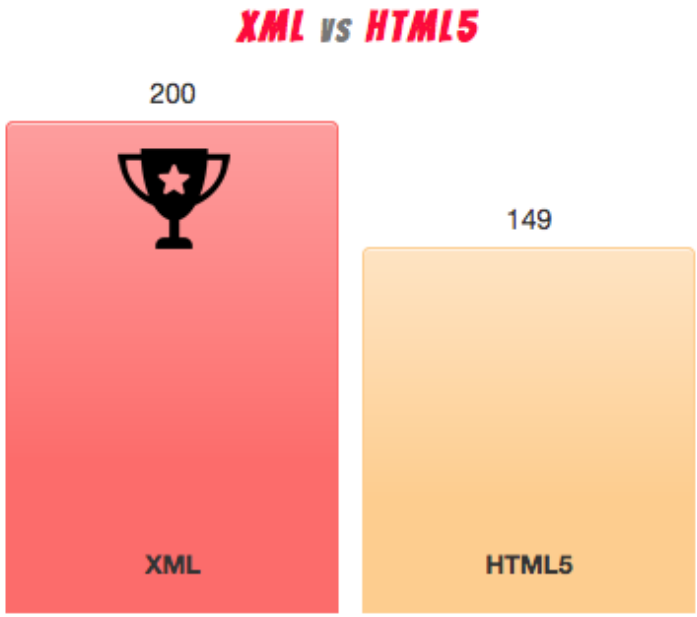
XML?!?

“If I invent another programming language, its name will contain the letter X.”

(N. Wirth, Software Pioniere Konferenz, Bonn 2001)

« Google fight » XML vs …

XML	**426 Mio**
SQL	218 Mio
HTML5	210 Mio
JSON	75 Mio
YML	6 Mio



http://www.googlefight.com/

Qu'est-ce que c'est XML?

Un standard développé par le Consortium du World Wide Web pour le texte structuré

Le "eXtensible Markup Language"

XML

XML sépare le contenu de la structure et de la présentation. Ceci permet de traiter les documents de manière automatique

Un langage de balisage extensible

Qu'est-ce que c'est XML?

"The features of XML make it possible to select certain contents out of a document and present them in various media."
(Huethig Publishing Group, Germany)

"We realized that XML would be the best way to store tax returns, tax schedules, and tax related messages."
(California Board of Equalization, USA)

XML

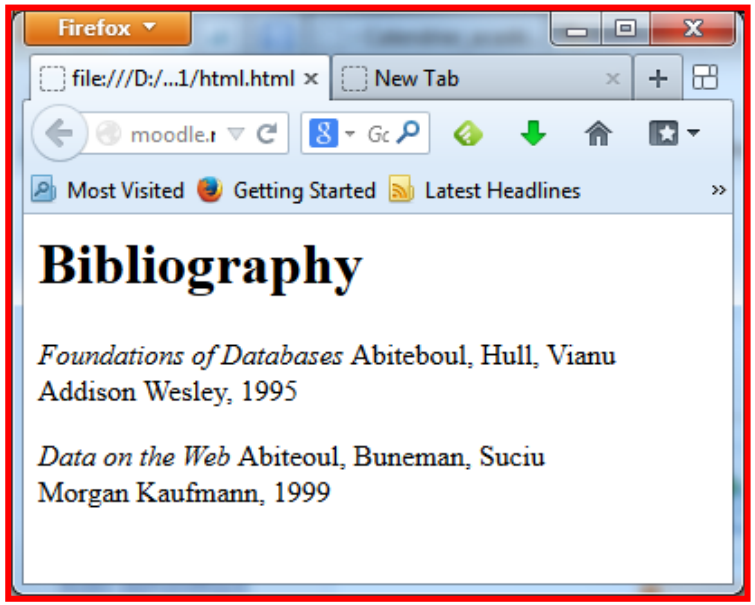
"In a few years' time, no company in the automotive sector will be able to manage without XML."
(Lear Corporation, Fiat Division, Italy)

"With XML it is easy for clients to re-present the information in the way that bests suits their needs."
(Sportsdata Pty Ltd, Australia)

Qu'est-ce que c'est XML?

- Motivation:
  - HTML décrit la présentation
  - XML décrit le contenu
- Extensible
  - On peut définir des nouvelles balises (pas comme HTML)
- International
  - Basé sur la norme Unicode

De HTML à XML



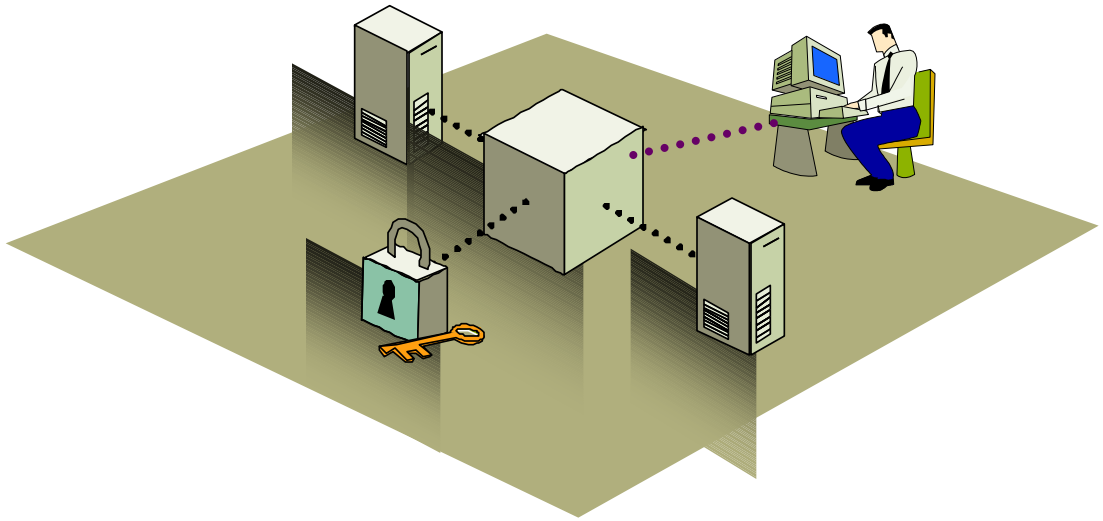
HTML

```
<h1> Bibliography </h1>
<p><i> Foundations of Databases </i>
    Abiteboul, Hull, Vianu
    <br> Addison Wesley, 1995
<p><i> Data on the Web </i>
    Abiteoul, Buneman, Suciu
    <br> Morgan Kaufmann, 1999
```

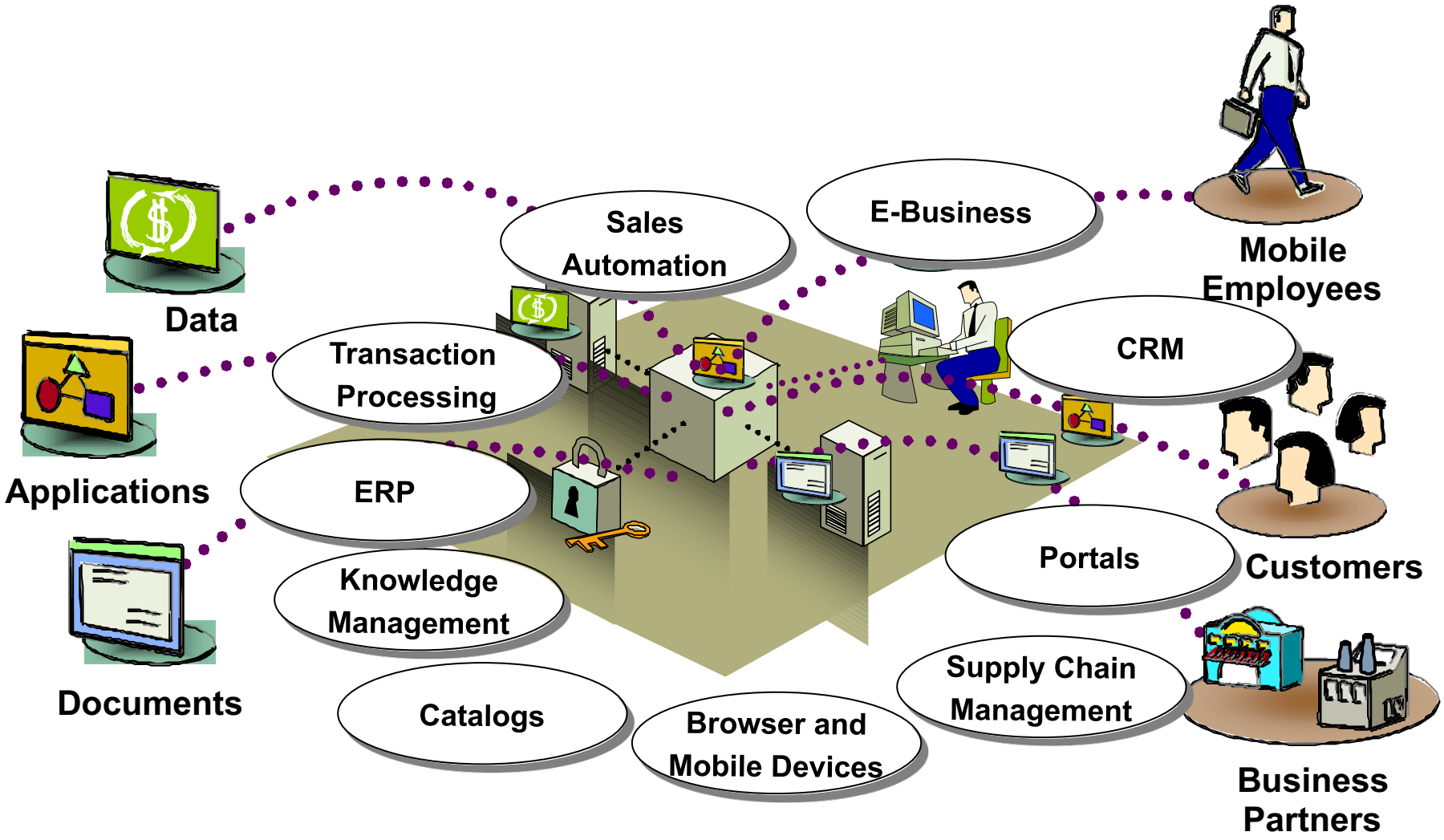
XML

```
<bibliography>
    <book>
        <title> Foundations… </title>
        <author>Abiteboul</author>
        <author> Hull </author>
        <author>Vianu</author>
        <publisher> Addison Wesley </publisher>
        <year> 1995 </year>
    </book>
    …
<bibliography>
```

Technologies IT: jusqu'à récemment



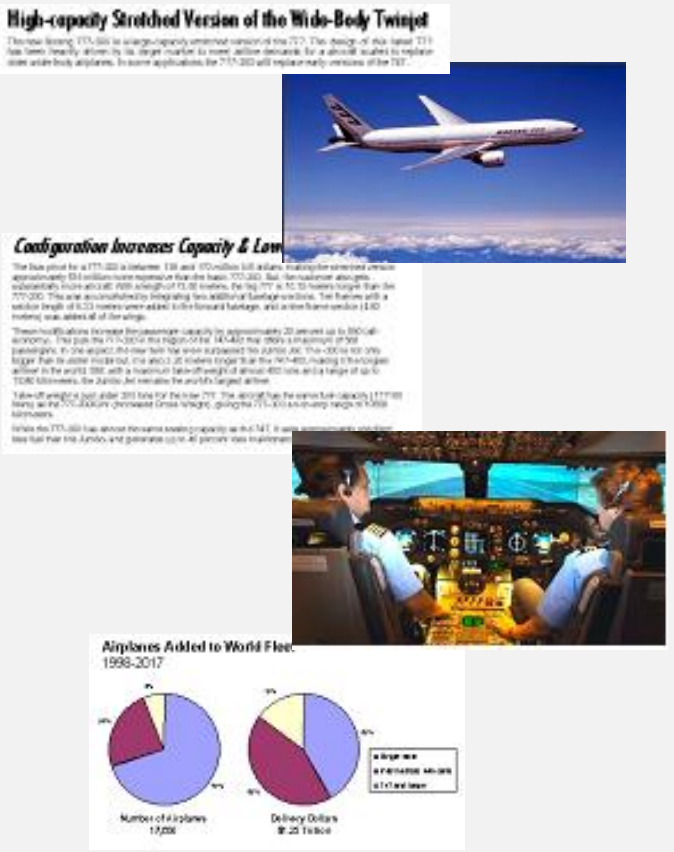
Technologies IT: aujourd'hui



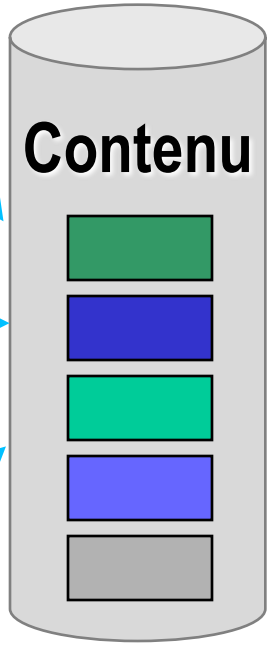
Gestion document vs gestion contenu

Documents

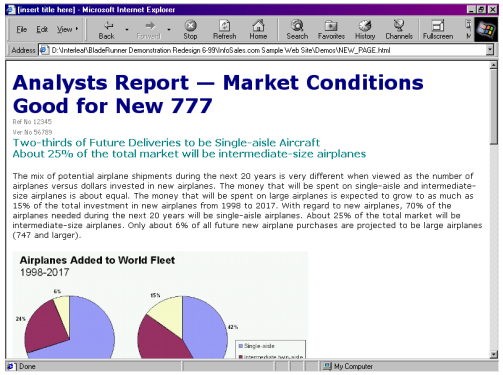
Contenu

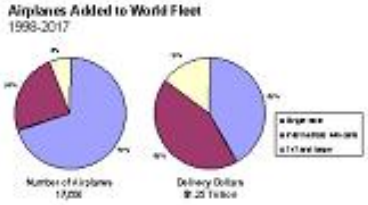


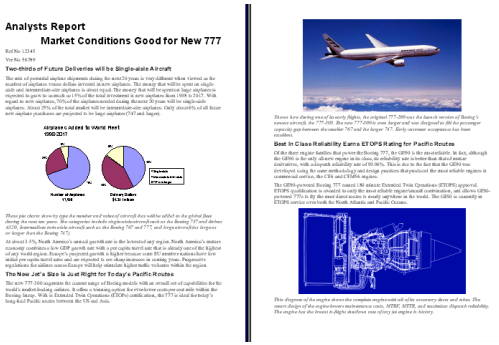
Gestion du contenu



- Extraire
- Assembler
- Formatter
- Publier



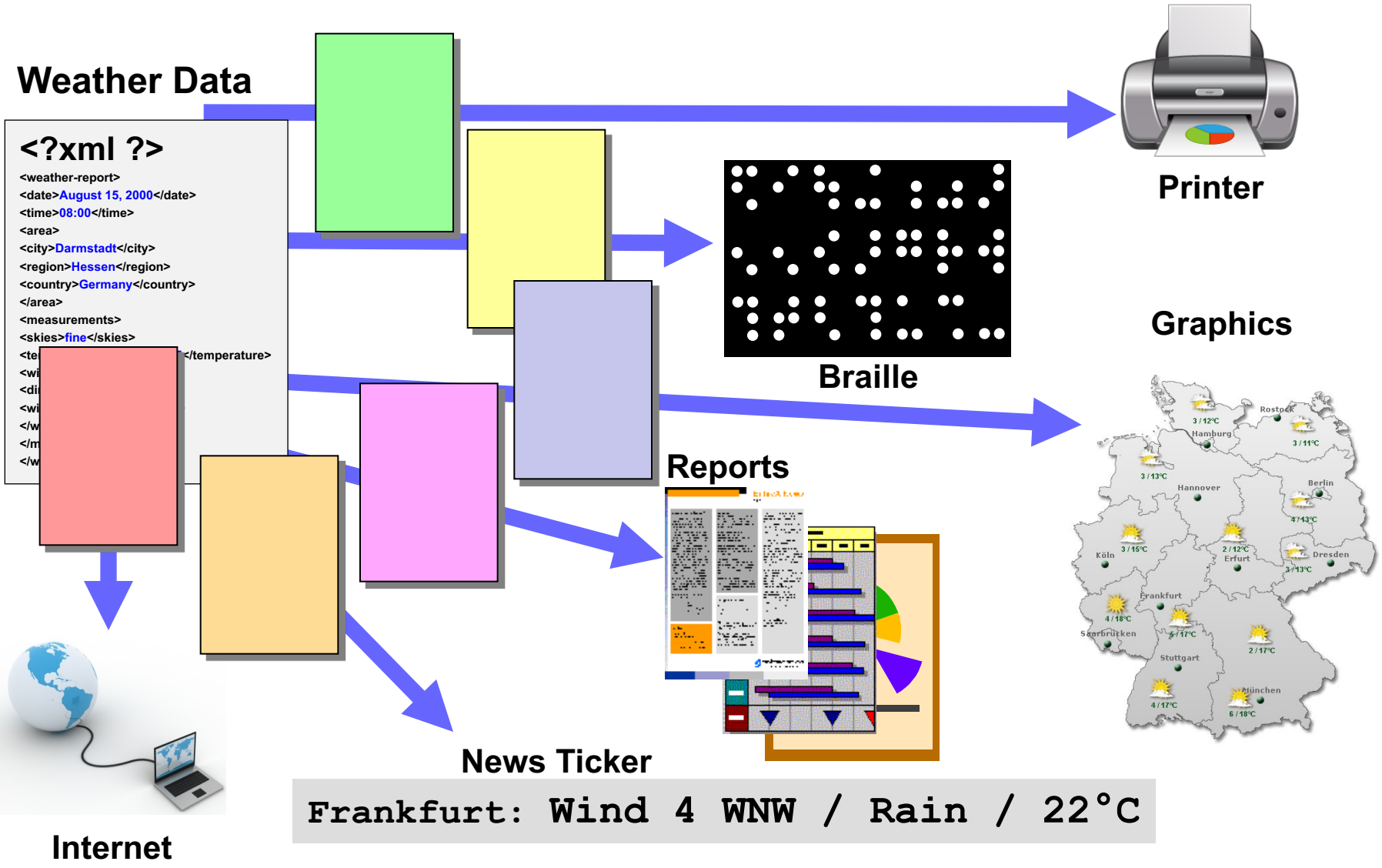




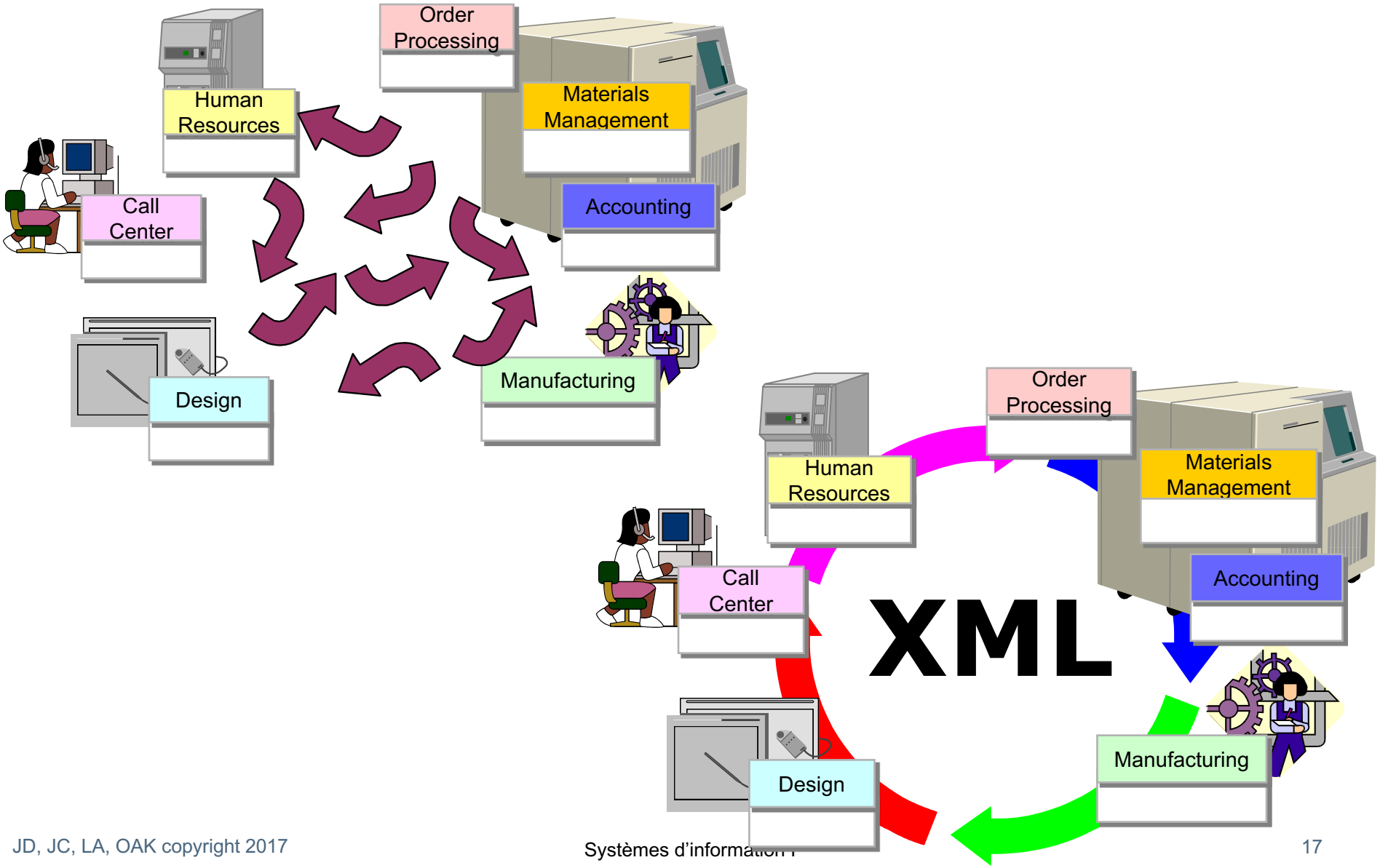
Ex. données météo

```
<?xml version="1.0"?>
<weather-report>
        <date>August 15, 2000</date>
        <time>08:00</time>
        <area>
                <city>Darmstadt</city>
                <region>Hessen</region>
                <country>Germany</country>
        </area>
        <measurements>
                <skies>fine</skies>
                <temperature scale="C">25</temperature>
                <wind>
                        <direction>SW</direction>
                        <windspeed>6</windspeed>
                </wind>
        </measurements>
</weather-report>
```

Ex. données météo



XML et l'intégration des applications



Les raisons du succès d'XML

- XML est un format de représentation des données générique
- XML est à la fois lisible pour les humains et pour les machines
- XML est basé sur une norme internationale (UNICODE)
- XML est un standard du W3C
- XML est indépendant de toute plateforme
- XML n'est pas seulement un format de représentation des données mais un ensemble complet de technologies
  - XML, XML Schema, XSLT, XSL-FO, DOM, SAX, etc.

Le Document !?!

- La métaphore document est un concept de base dans divers secteurs:
  - Business
  - Administration
  - Science
  - Technologie de l'information
  - Etc.
- C'est une évolution de 9'000 ans d'expérience humaine

Le Document !?!

- "Scripture"
  - Graver sur des pierres comme système de comptage (ex. animaux, mesures de grains, etc.)

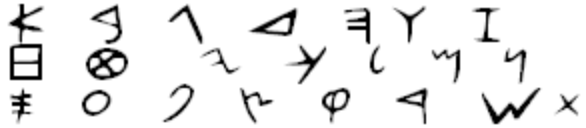
Neolothic

  - Graver des pictographies représentant des femmes, le soleil, etc.

Sumerian pictographs

  - Introduction de la notion du son, uniquement des consonnes

Phoenician alphabets



Définition – document

- Qu'est-ce qu'un document ?
  - Écrit servant de preuve ou de titre - *Dictionnaire Larousse*
  - Toute chose écrite qui peut servir à nous renseigner - *Dictionnaire Quillet-Flammarion*
  - Un document est l'ensemble d'un support d'information et des données enregistrées sur celui-ci sous forme en général permanente et lisible par l'homme ou par une machine - *ISO, Organisme international de normalisation*
    - ✓ Document électronique est représenté par un codage numérique

Types de documents

- Type de document
  - Une classe de documents partageant une structure fondamentale commune.
  - La définition ISO 8879 est donnée par « une classe de document possédant des caractéristiques similaires ; par exemple journal, article, manuel technique, ou mémo. (4.102) »
- On distingue deux grandes familles:
  - **Documents linéaires**
  - **Documents structurés**

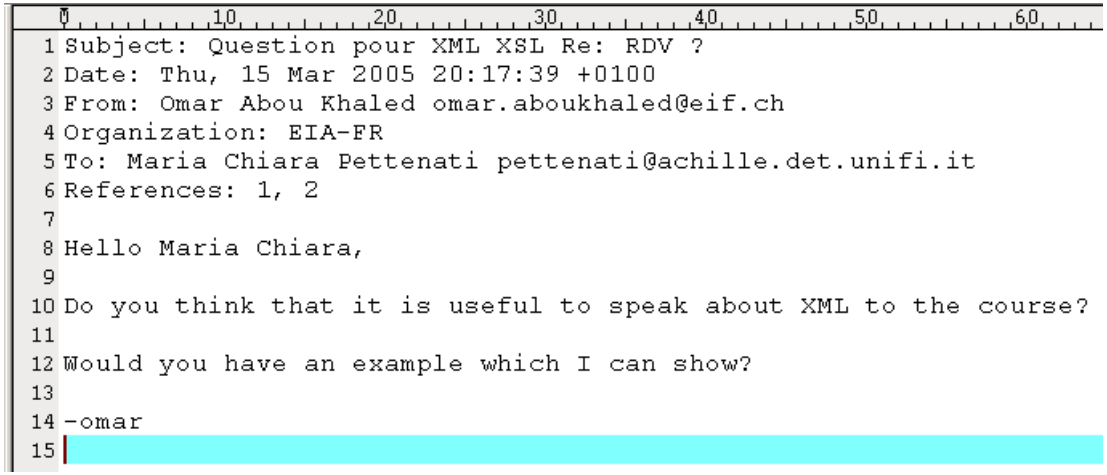
Documents linéaires

- Les documents linéaires sont modélisés par un flot de données qui mêle le contenu du document aux informations de présentation et/ou de typage.

- Les formats de documents linéaires sont très proches de leurs modèles :
  - Ils sont une simple transcription de l'information contenue dans le modèle.

Documents linéaires

- L'email est un document sous forme de texte plein
  - comportant un en-tête comprenant un ensemble de champs (expéditeur, destinataire, date et heure, sujet, etc.) et un corps contenant le message proprement dit.
- Ne contient que la composante textuelle du document
- Aucune information supplémentaire n'est représentée dans ce modèle.



```
Subject: Question pour XML XSL Re: RDV ?
Date: Thu, 15 Mar 2005 20:17:39 +0100
From: Omar Abou Khaled omar.aboukhaled@eif.ch
Organization: EIA-FR
To: Maria Chiara Pettenati pettenati@achille.det.unifi.it
References: 1, 2

Hello Maria Chiara,

Do you think that it is useful to speak about XML to the course?

Would you have an example which I can show?

-omar
```

Documents linéaires

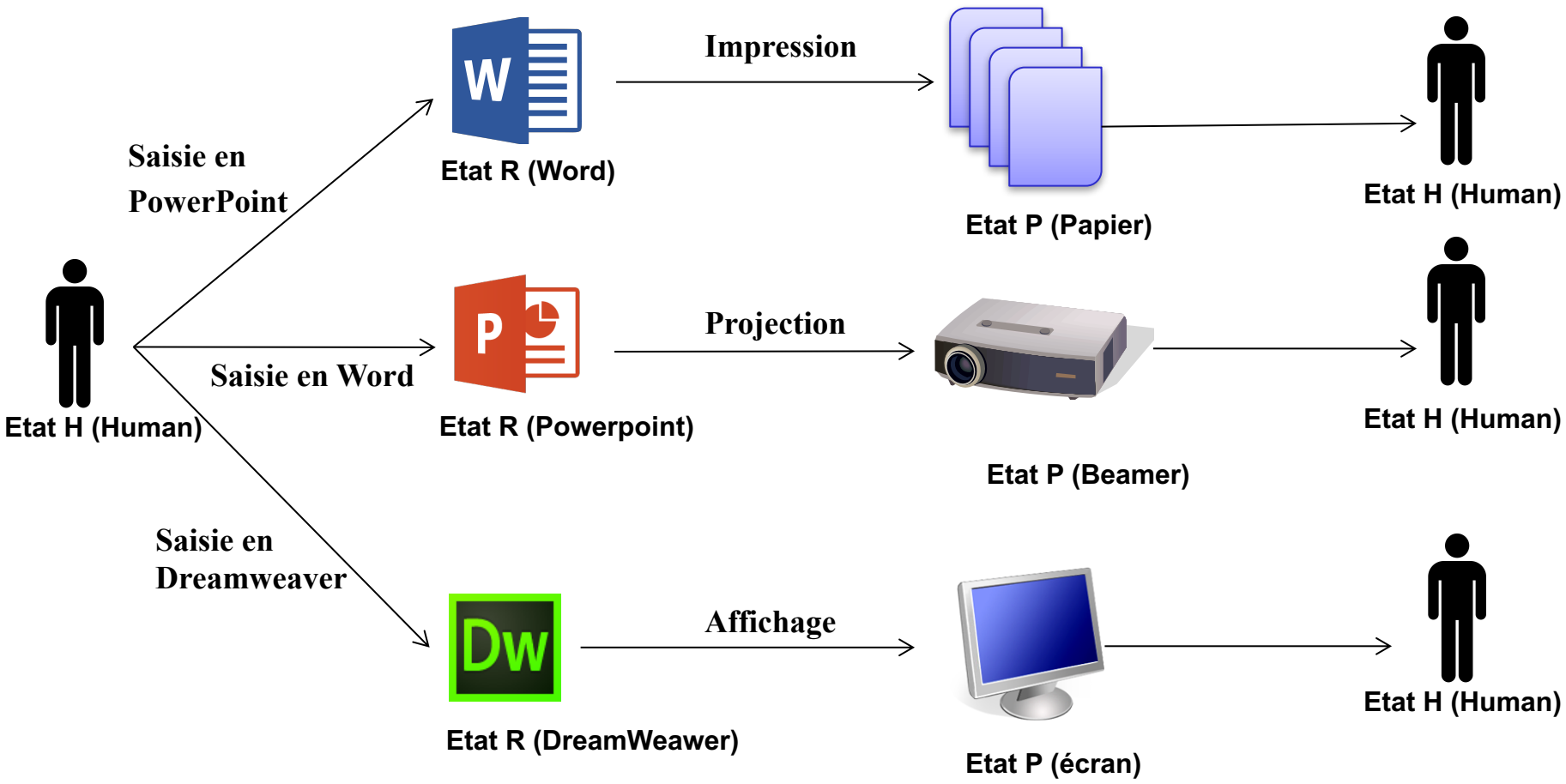
- Documents formatés
- Le contenu textuel du document est entrecoupé d'indications de changement de format du flot de texte (RTF - Rich Text Format, Latex, HTML)
- Les indications peuvent être sous forme binaire ou sous forme textuelle

```
<P CLASS="western" STYLE="margin-bottom: 0cm"><SPAN LANG="fr-FR"><B>Subject:</B>
Question pour XML XSL Re: RDV ?</SPAN></P>
<P CLASS="western" STYLE="margin-bottom: 0cm"><B>Date:</B> Thu, 15
Mar 2005 20:17:39 +0100</P>
<P CLASS="western" STYLE="margin-bottom: 0cm"><B>From: </B>Omar Abou
Khaled omar.aboukhaled@eif.ch</P>
<P CLASS="western" STYLE="margin-bottom: 0cm"><SPAN LANG="it-IT"><B>Organization:</B>
EIA-FR</SPAN></P>
<P CLASS="western" STYLE="margin-bottom: 0cm"><SPAN LANG="it-IT"><B>To:</B>
Maria Chiara Pettenati pettenati
<P CLASS="western" STYLE="margin
1, 2</SPAN></P>
<P CLASS="western" STYLE="margin
</P>
<P STYLE="margin-bottom: 0cm">He
</P>
<P LANG="it-IT" CLASS="western"
</P>
<P CLASS="western">Do you think
to the course?
</P>
<P CLASS="western"><BR><BR>
</P>
<P CLASS="western">Would you hav
<P CLASS="western" STYLE="margin
```

```
\asianbrkrule\rsidroot8473276\newtblstyruls\nogrowautofit \fet0\sectd \linex0\headery708\footery708\colsx7(
\pnucltr\pnstart1\pnindent720\pnhang {\pntxta .}}{\*\pnseclvl3\pndec\pnstart1\pnindent720\pnhang {\pntxta .
\pnlcltr\pnstart1\pnindent720\pnhang {\pntxtb (}{\pntxta )}}{\*\pnseclvl7\pnlcrm\pnstart1\pnindent720\pnhar
{\pntxtb (}{\pntxta )}}\pard\plain \ql \li0\ri0\widctlpar\aspalpha\aspnum\faauto\adjustright\rin0\lin0\itap
Subject:}{\lang1036\langfe1033\langnp1036\insrsid8473276\charrsid13047040  }{\lang1036\langfe1033\langnp10
\par }{\b\insrsid8473276\charrsid8658658 Date:}{\insrsid13047040  Thu, 15 Mar 2005}{\insrsid8473276  20:17:
\par }{\b\insrsid8473276\charrsid8658658 From: }{\insrsid13047040 Omar Abou Khaled}{\insrsid8473276  }{\ins
\par }{\b\lang1040\langfe1033\langnp1040\insrsid8473276\charrsid13047040 Organization:}{\lang1040\langfe10
\lang1040\langfe1033\langnp1040\insrsid8473276\charrsid13047040
\par }{\b\lang1040\langfe1033\langnp1040\insrsid8473276\charrsid13047040 To:}{\lang1040\langfe1033\langnp1(
\lang1040\langfe1033\langnp1040\insrsid13047040  }{\lang1040\langfe1033\langnp1040\insrsid13047040\charrsi
\par }{\b\lang1036\langfe1033\langnp1036\insrsid8473276\charrsid8658658 References:}{\lang1036\langfe1033\
\lang1036\langfe1033\langnp1036\insrsid8473276\charrsid8473276  2
\par }{\i\insrsid13047040
\par }\pard \ql \li0\ri0\widctlpar\aspalpha\aspnum\faauto\adjustright\rin0\lin0\itap0\pararsid13047040 {\la
\par }{\lang1040\langfe1033\langnp1040\insrsid12650585\charrsid13047040
\par }{\insrsid13047040\charrsid13047040 Do you think that it is useful to speak about XML to the course?
\par }{\insrsid12650585\charrsid13047040
\par }{\insrsid13047040\charrsid13047040 Would you have an example which I can show?
\par }{\i\insrsid13047040
\par -omar}{\i\insrsid13047040\charrsid13047040
\par }\pard \ql \li0\ri0\widctlpar\aspalpha\aspnum\faauto\adjustright\rin0\lin0\itap0\pararsid8473276 {\i\
\par }}
```

Modèle de production

- Processus traditionnel de publication via traitement de texte
  - **HRPH :** Human ➔ Rendering ➔ Presentation ➔ Human



Documents linéaires

- Désavantages
  - Le style est mêlé au contenu
  - Réutilisation des données exige une duplication de l'effort de saisie
- Solution ?
  - Changer le modèle de production

Documents structurés

- Les documents structurés sont fondés sur les relations logiques entre les différents composants du document plutôt que sur leur apparence et leur position dans la page.
- Cette différence se répercute sur les formats de stockage qui dissocient l'information de structure du document des informations nécessaires à son exploitation :
  - Formatage, édition, etc.
- Représenter les documents sous forme structurée facilite :
  - L'édition et la mise à jour du contenu
  - Le contrôle de la présentation
  - Le travail collaboratif
  - La recherche d'information
  - etc.

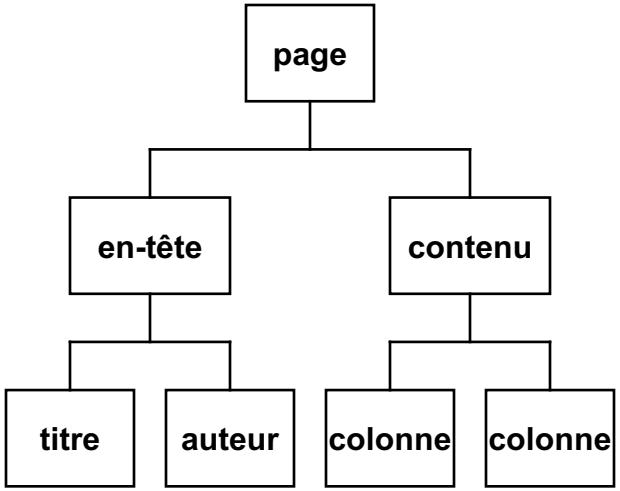
Documents structurés

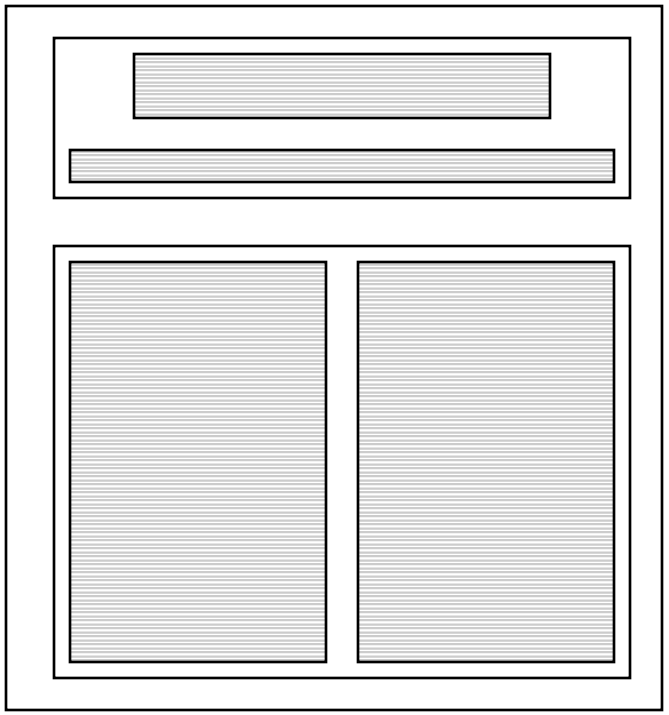
- Structure logique
  - Décrit la relation entre les éléments
  - Décrit le point de vue de l'auteur / du lecteur
  - Contient des chapitres, sections, titres, figures, légendes, définitions, citations, etc.

- chapitre
  - titre
    - texte
  - introduction
    - paragraphe
      - texte
      - image
    - paragraphe
      - texte
  - section
    - titre_section
      - texte

Structures physiques

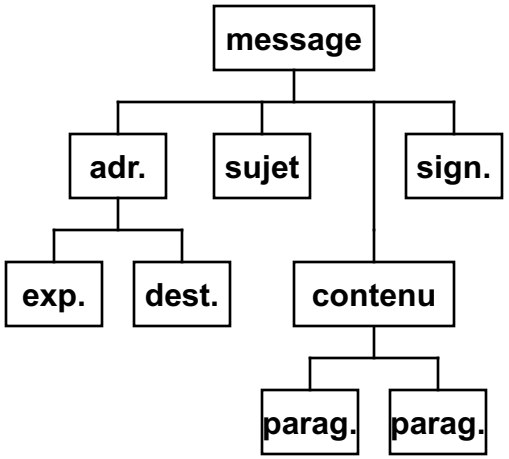
- Structure physique
  - Décrit le rendu du documents (format, style, …)
  - Décrit la mise en page (présentation)
  - Contient des pages, régions, colonnes, blocs, lignes, etc.
  - Les structures physiques sont généralement décrites par des boîtes juxtaposées et/ou imbriquées





Exemple: structures d'un message

- Structure logique



Rolf Ingold

Elena Mugellini

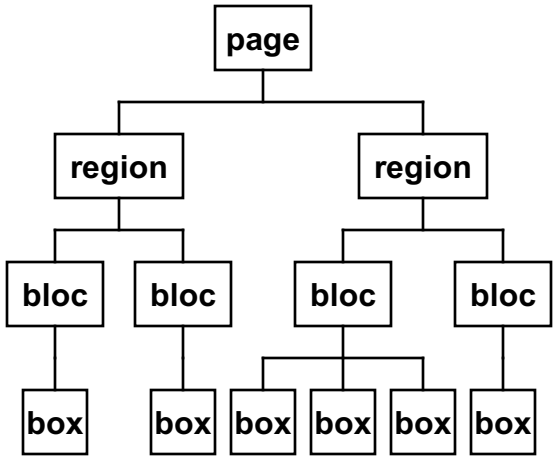
Question pour le cours de SI 1

Elena, penses-tu qu'il soit utile de parler de XML au cours ?

Aurais-tu un exemple que je peux montrer ?

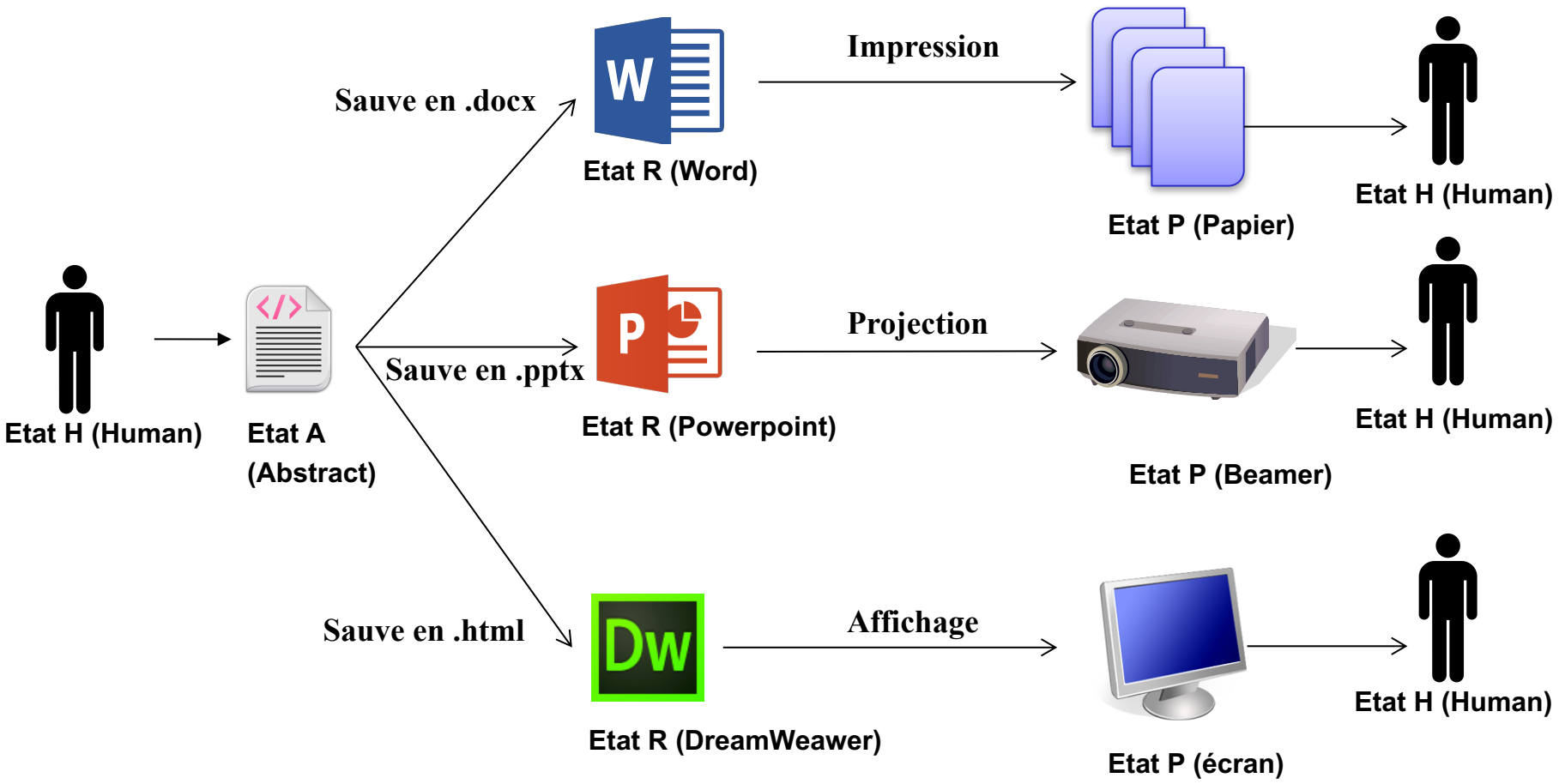
Rolf

- Structure physique



Publication via document structuré

- Processus XML/SGML : H ➔ A ➔ R ➔ P ➔ H
  - Human → Abstraction → Rendering → Presentation → Human



Modèle de document

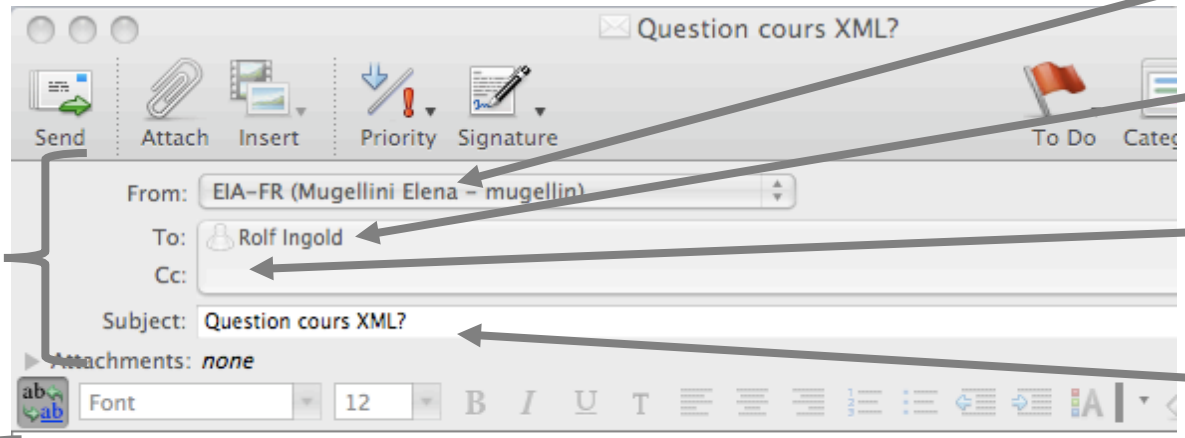
- Modèle:
  - *« Une représentation schématique d'un processus, d'une démarche raisonnée » - Larousse.*
- Un modèle est un document possédant certaines caractéristiques qui servent de base à la création d'autres documents du même style.
- L'utilisation d'un modèle, pour la création d'un document, permet :
  - de réduire au minimum les tâches de mise en page (orientation, taille des marges, en tête/pied de page, etc.)
  - de prédéfinir la forme et l'emplacement des différents éléments du texte (titres, sous titres, paragraphes de texte, tableaux, images…).

Descriptions génériques

- Besoin de définir les règles de structuration
  - Ensemble des classes d'éléments
  - Contraintes
    - ✓ sur l'ordre des éléments
    - ✓ sur l'imbrication des éléments
    - ✓ sur le nombre d'éléments
  - Formalisation à l'aide d'une grammaire avec les opérateurs suivants
    - ✓ agrégats (rassemblement d'un ensemble d'élément)
    - ✓ choix (ou alternatives)
    - ✓ répétition (ev. avec contraintes numériques)

Le texte : un message électronique

- Exemple d'un message électronique



Salut Rolf,

penses-tu qu'il soit utile de parler de XML au cours ?
Aurais-tu un exemple que je peux montrer ?

Merci et bonne journée,

Elena

Dr. Elena Mugellini ▪ Professeure ▪ Responsable groupe MISG ▪ www.misg.ch
Département TIC ▪ Ecole d'ingénieurs et d'architectes de Fribourg
Bd Pérolles 80 ▪ CP 32 ▪ CH-1705 Fribourg (Suisse)
tel +41 (0)26 429 68 70 ▪ fax +41 (0)26 429 66 00 ▪ www.eif.ch

Expéditeur

Destinataire

Destinataire en copie

Un en-tête

Sujet

Un corps de message

Texte du message

Signature

Exemple: struct. génér. des messages

- La structure générique d'un message exprime les règles suivantes
  - Un message comprend un expéditeur, un ou plusieurs destinataires, un sujet, un contenu et une signature
  - Le sujet suit le(s) destinataire(s) qui sui(ven)t l'expéditeur
  - Le contenu est composé d'une suite de paragraphes
  - La signature est facultative

Représentation formelle d'un message électronique

- courrier → en-tête, corps-message
- en-tête → émetteur, destinataire, cc?, sujet?
- émetteur → adresse-électronique
- destinataire → adresse-électronique+
- cc → adresse-électronique+
- sujet → caractères
- corps-message → message, signature?, adresse-postale?
- message → caractères
- signature → caractères
- adresse-postale → caractères
- adresse-électronique → nom, @, prestataire
- nom → caractères
- prestataire → caractères

Représentation formelle

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
un message comprend un expéditeur, un ou plusieurs destinataires, un sujet, un contenu et une
  signature, le sujet suit le(s) destinataire(s) qui sui(ve)t l'expéditeur. le contenu est
  composé d'une suite de paragraphes
la signature est facultative
-->
<!ELEMENT Message (Subject, Sender, Receiver+, Body, Date, Signature?)>
<!ELEMENT Subject (#PCDATA)>
<!ELEMENT Sender (#PCDATA)>
<!ELEMENT Receiver (#PCDATA)>
<!ELEMENT Body (Parag*)>
<!ELEMENT Parag (#PCDATA)>
<!ELEMENT Date (#PCDATA)>
<!ELEMENT Signature (#PCDATA)>
<!ATTLIST Message (Confidential | Public) "Public">
```

Exemple XML

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<Message Status="Public" xmlns="unifr">
  <Subject>Subject: Question cours XML</Subject>
  <Sender>
        Date: Thu, 15 Mar 2005 20:17:39 +0100From: Elena Mugellini
        elena.mugellini@hefr.ch Organization: EIA-FR
    </Sender>
  <Receiver>To: Rolf Ingold rolf.ingold@unifr.ch</Receiver>
  <Body>
        Salut Rolf, penses-tu qu'il soit utile de parler de
        XML au cours ? Aurais-tu un exemple que je peux
        montrer ? Merci et bonne journée
  </Body>
  <Signature>Elena</Signature>
</Message>
```

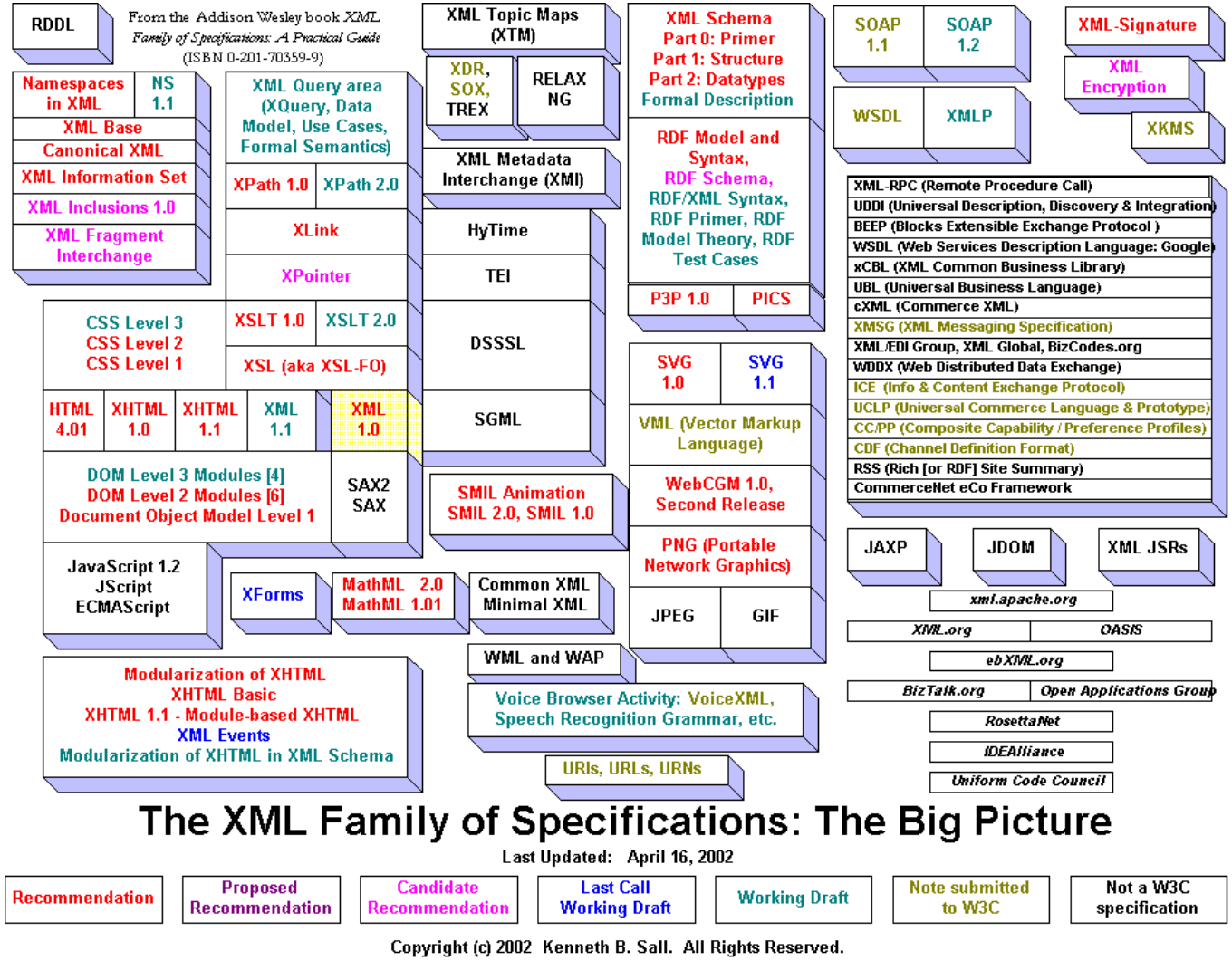
Documents structurés : XML

- XML
  - est une méthode universelle et standardisée de représentation textuelle de données structurées
    - ✓ Un format de document, un format de données, un métalangage, un mode de structuration de l'information
- A quoi peut servir XML ?
  - Publication électronique internationalisée
    - ✓ Indépendant des plateformes, Indépendant des langues, Indépendant des média
  - Applications Web
    - ✓ Échanges entre bases de données, Gestion de collections de documents, Traitement distribué, Manipulation de données côté client, Transactions commerciales

Resumé XML

- Meta langage
- Avec XML chacun peut définir ses propres balises
- XML sépare les données des informations nécessaires pour la présentation des données
- XML est "self-describing“

L'origine des espèces de document



Structuration et modèles de données

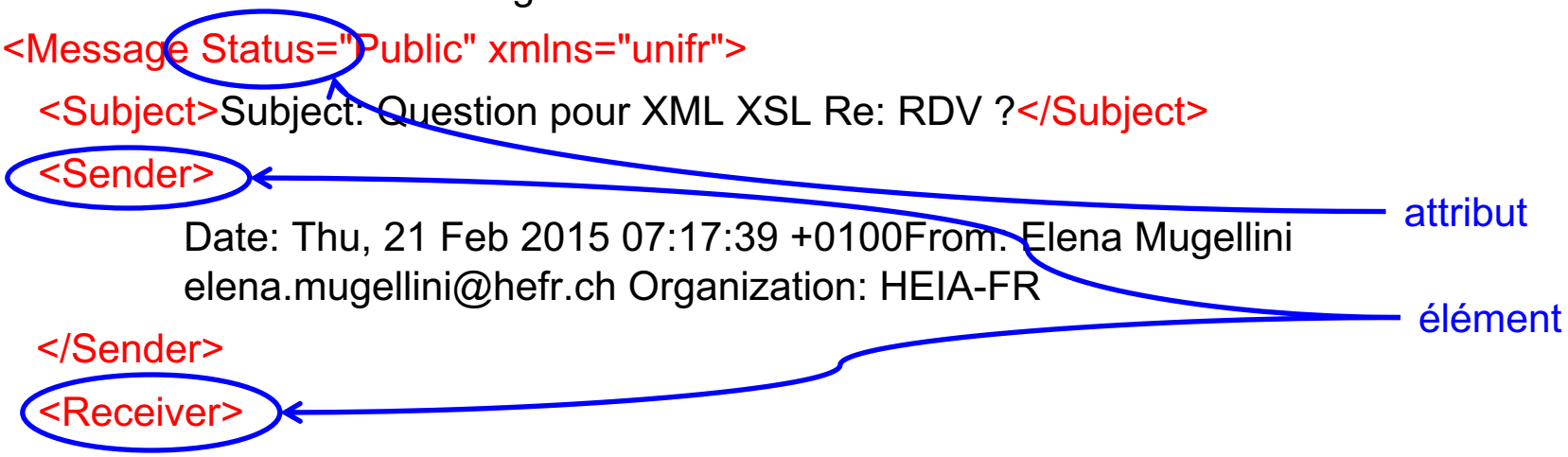
PARTIE 2 : DTD

Buts et plan

- Buts
  - Comprendre les principes de base d'un document structuré
    - ✓ Contenu, structure, et présentation
  - Modéliser une DTD
    - ✓ Les règles de construction
    - ✓ Structure et syntaxe
  - Construire un fichier XML
    - ✓ Structure et syntaxe
    - ✓ "well-formed" vs valide

Example document XML

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<Message Status="Public" xmlns="unifr">
  <Subject>Subject: Question pour XML XSL Re: RDV ?</Subject>
  <Sender>
        Date: Thu, 21 Feb 2015 07:17:39 +0100From: Elena Mugellini
        elena.mugellini@hefr.ch Organization: HEIA-FR
  </Sender>
  <Receiver>
        To: Rolf Ingold rolf.ingold@unifr.ch
</Receiver>
<Body>
  Hello Rolf, Do you think that it is useful to speak about XML to the course?
  Would you have an example which I can show?
</Body>
  <Signature>Elena</Signature>
</Message>
```



Les principes de base d'un document I

- En général on développe des modèles pour des documents:
  - Qui ont une longue durée de vie
  - Qui seront utilisés par plusieurs utilisateurs
- Il est très important d'investir du temps dans la structure de ses documents.
  - Moins de maintenance
  - Moins d'effort de programmation
- Trois concepts à retenir
  - Modèle de contenu
  - Type de document
  - Instance de document

Les principes de base d'un document II

- Trois aspects dictent l'existence d'un document comme outil de transport de l'information
  - **Le contenu**: ce qui est dans l'instance du document
    - ✓ Ex: une lettre contient
      - Le nom du destinataire: Toto
      - Le corps de la lettre: Bla bla bla
      - La signature: Titi
  - **La structure**: comment les éléments d'information sont liés.
    - ✓ Groupements et relations entre composants
    - ✓ Elle définit les connaissances que les outils de traitement peuvent avoir sur le contenu du document
      - Ex: si on définit dans la lettre, le nom et le prénom comme des éléments distincts, les outils de traitement peuvent les distinguer facilement. Par contre si on définit le corps de la lettre comme un seul élément, il sera très difficile de distinguer les différentes rubriques.
  - **La présentation**: l'apparence du document pour un humain
    - ✓ Le même document doit être affiché sur un papier, sur une TV, sur un smartphone, etc.
    - ✓ Le point essentiel est de séparer la structure informationnelle du document de son apparence.

DTD : Document Type Definition

- Décrit la grammaire d'un document XML (ou SGML - Standard Generalized Markup Language)
- Donne l'ensemble des règles qui permettent d'interpréter le balisage d'un document XML.
  - Ces règles décrivent les éléments d'un document et les contextes de leurs apparitions:
    - ✓ Permet de placer des contraintes sur la hiérarchie des éléments d'un XML
    - ✓ Permet de spécifier des attributs (informations additionnelles) associés aux éléments
- Avantages:
  - Automatisation des processus de manipulation
- On distingue deux types de documents
    - ✓ **Data-centric** (orienté donnée)
      - Utilisation pour échange des données entre plateformes
    - ✓ **Document-centric** (orienté document ou narratif)
- La DTD prend son sens pour les documents XML orientés-document.
  - Focalisation sur la hiérarchie plus que sur le typage de données

Data-centric

- Les documents **Data-centric** se caractérisent par :
  - Structure assez régulière
  - Data avec granularité fine
  - Peu de modèle de contenu mixte
  - L'ordre des éléments n'est pas significatif
- Ce type de document est généralement destiné à la consommation par des machines

```
<Orders>
  <SalesOrder SONumber="12345">
       <Customer CustNumber="543">
                 <CustName>ABC Industries</CustName>
                 <Street>123 Main St.</Street>
                 <City>Chicago</City>
                 <State>IL</State>
                 <PostCode>60609</PostCode>
       </Customer>
       <OrderDate>981215</OrderDate>
       …
```

Document-centric

- Les documents **Document-centric** se caractérisent par :
  - Structure irrégulière
  - Data avec large granularité
  - Plus de modèle contenu mixte
  - L'ordre des éléments est toujours significatif
- Exemple

```
<para>
  <quote speaker="omar">"je ne crois pas en XML"</quote> omar a
  dit. Il a surement fait une blague. Normalement il dit le
  contraire: <quote speaker="omar">"XML c'est super"</quote>.
</para>
```

- Deux remarques:
  - Si on enlève les balises, la phrase reste lisible pour l'être humain
  - L'ordre de l'information est capitale pour comprendre le sens de la phrase. On ne peut pas réarranger l'ordre des balises sans perdre le sens.

Développement d'une DTD – I

- Degré de difficulté:
  - ✓ Dépend de l'information à modéliser
  - ✓ Dépend de l'usage qui sera fait de cette information
- 1. Identifier les éléments:
  - ✓ Contenu: signification de l'information (adresse, maison, etc.)
  - ✓ Structure: répartition de l'information (adresse = rue + ville + etc.)
  - ➢ Éviter le balisage de mise en forme !!!
    - ✓ (paragraphes, tableaux, etc.)
- 2. Structurer les éléments:
  - ➢ Organisation sous forme hiérarchique
  - ➢ Chercher les éléments conteneurs d'autres éléments
    - ✓ Création d'éléments qui regroupent d'autre éléments d'une manière logique (listes, tables, etc.)
      - – Facilite la numérotation automatique, etc.

Développement d'une DTD - II

- 3. Établir des règles:
  - ✓ Spécification d'éléments obligatoires et facultatifs
  - ✓ Soigner et calibrer la rigidité ou la souplesse de votre DTD
- 4. Spécifier les attributs:
  - Affectation des attributs pour chaque élément (taille, couleur, etc.)
  - Parfois on se rend compte qu'il faut déplacer des éléments vers les listes des attributs
  - Pas de règles pour le choix entre élément et attribut (débat ouvert)
- 5. Logiciels
  - Éditeur DTD
    - ✓ En format d'arbre directement
    - ✓ En format tableau
    - ✓ À plat

Éléments ou attributs ?

- Débat ouvert
- Voici quelques éléments
  - Un éditeur XML intervient plus facilement sur les éléments que sur les attributs
  - Lorsque le fichier est destiné à des traitements humain mieux vaut utiliser les éléments
  - Les processeurs XML interrogent les valeurs d'attributs plus facilement qu'ils vérifient les contenu d'un élément
  - Pour le travail collaboratif, il est plus simple de fractionner les documents sur la base d'éléments que sur les attributs

Les composants d'une DTD

- On distingue quatre types de composants:
  - Les **déclarations** de type **d'élément**
    - ✓ Element type declarations
  - Les **déclarations** des listes **d'attributs**
    - ✓ Attribute-list declarations
  - Les **déclarations** de **notations**
    - ✓ Notation declarations
  - Les **déclarations d'entités**
    - ✓ Entity declarations

Element Type Declarations I

- Element Type Declarations
  - **<!ELEMENT *element_name content_model*>**
  - ***element_name*** : tout nom permis par XML, respecte la casse.
  - ***content_model*** : définit ce que l'élément peut contenir
    - ✓ Donnée ou ensemble d'éléments fils possibles
- La DTD permet quatre différents modèles de contenu:
  - `Element, Mixed, EMPTY, ANY`
- **Element Content Model**
  - Un élément containeur qui contient uniquement d'autres éléments
  - Définit la relation d'imbrication entre les éléments
    - ✓ Ex. `<!ELEMENT authors (author)>`
    - ✓ Ex. `<!ELEMENT book (title, authors, publisher)>`

Element Type Declarations II

- **Element Content Model** (suite)
  - Définition des occurrences
    - ✓ Le **+** : indique une répétition d'occurrences, l'élément sera présent **au moins une fois**
      - Ex. `<!ELEMENT authors (author+)>`
    - ✓ L'***** : indique une répétition optionnelle d'occurrences, l'élément sera présent **zéro ou plusieurs fois**
      - Ex. `<!ELEMENT authors (author*)>`
    - ✓ Le **?** : indique une occurrence optionnelle, l'élément sera présent **au plus une fois** (0 ou 1 fois)
      - Ex. `<!ELEMENT book (title, authors, publisher ?)>`
    - ✓ Sans occurrence : indique que l'élément existe **une seule fois** uniquement.
      - Ex. `<!ELEMENT book (title, authors, publisher)>`

Element Type Declarations III

- **Element Content Model** (suite)
  - Définition des séquences
    - ✓ La **,** : indique une séparation d'éléments avec un ordre d'apparition dans l'arbre
      - Ex. `<!ELEMENT book (title, authors, publisher)>`
  - Permettre les choix
    - ✓ La **|**: indique un choix exclusif et sépare les éléments dans une liste de choix
      - Ex. `<!ELEMENT book (title | publisher)>`
  - Le groupement
    - ✓ Avec les **()** permet un regroupement pour un élément complexe
      - Ex. `<!ELEMENT book (title, authors, (pubDate | publisher), description)>`

Element Type Declarations IV

- **Mixed Content Model**
  - Selon la spécification XML 1.0 le modèle de contenu mixte peut être de deux types:
    - ✓ Contenu contenant uniquement une chaîne de caractères
      - Ex. `<testId>1</testId>`
      - Déclaration: `<!ELEMENT testId (#PCDATA)>`
        - » `PCDATA`= parsed character data
    - ✓ Contenu contenant une chaîne de caractère et d'autres éléments fils
      - Déclaration: `<!ELEMENT p (#PCDATA | b | em)*>`
      - Selon XML 1.0:
        - » `#PCADATA` en premier
        - » Il faut utiliser absolument et uniquement l'* à l'extérieur.

```
<p>Without question, <b>Weird Al</b> is the <em>greatest</em>
singer <b>of all time!</b></p>
```

Element Type Declarations V

- **EMPTY Content Model**
  - Pour définir un élément qui ne contient rien du tout.
  - En général, on déclare un élément vide pour lui attacher une liste d'attribut
    - ✓ Ex. `<cover/>` ou `<cover></cover>`
    - ✓ Déclaration : `<!ELEMENT cover EMPTY>`
- **ANY Content Model**
  - Pour définir pour n'importe quel type de contenu, n'importe quel élément fils ou `PCDATA`.
  - Pas très utilisés et va dans le sens contraire de la validation.
  - Déclaration : `<!ELEMENT author ANY>`

Attribute-List Declarations I

- Attribute-list declaration
  - **<!ATTLIST *element_name attribute_name datatype default_type*>**
    - ✓ ***element_name*** définit l'élément auquel sera attaché l'attribut
    - ✓ ***attribute_name*** définit le nom de l'attribut
    - ✓ ***datatype*** définit le type de données pour la valeur de l'attribut
    - ✓ ***default_type*** définit l'occurrence de l'attribut (exigée ou optionnelle), ou il définit une valeur fixe ou une valeur par défaut.

```
<!ELEMENT catalog (CD | cassette | record | MP3)*>
<!ATTLIST catalog name CDATA #IMPLIED>
```

- Datatypes
  - Pour chaque attribut on doit définir un datatype.
  - Il existe 10 différents types:
    - ✓ `CDATA, ID, IDREF, IDREFS, ENTITY, ENTITIES, NMTOKEN, NMTOKENS, NOTATION, (enumerated value).`

Attribute-List Declarations II

- **CDATA** (character data)
  - Contient une chaîne de caractères ou du texte
  - Ex. `<book categorie="XML XHTML HTML" >…</book>`
  - Déclaration :
    - ✓ `<!ELEMENT book (title, author, description)>`
    - ✓ `<!ATTLIST book categorie CDATA #IMPLIED>`
    - ✓ Pas très précis comme type !
- **ID**: contient un nom unique
- **IDREF**: référence un autre élément contenant la même valeur identifié en tant que `ID`
- `IDREFS`: contient une série de `IDREFs` délimité par des espaces vides

Attribute-List Declarations III

- ID et IDREF(S)
  - ID et IDREF sont utilisés ensemble.
  - Considérations spéciales :
    - ✓ Chaque élément peut avoir uniquement un seul ID
    - ✓ L' ID peut être soit #IMPLIED soit #REQUIRED
    - ✓ L' ID aura un nom selon la spécification XML
  - Le IDREF(S) sert de renvoi vers des données. Il respecte les mêmes considérations.
    - ✓ Il peut référencer uniquement un ID interne au document
  - Peut être considéré comme un mécanisme de lien interne au document.
  - Ex.

```
<!ELEMENT book (title, author, description)>
<!ATTLIST book id_name ID #IMPLIED>
…
<!ELEMENT reference EMPTY>
<!ATTLIST reference idRef_name IDREF #REQUIRED>
```

Attribute-List Declarations IV

ID et IDREF(S) (suite) Ex.

```
<?xml version="1.0"?>
<!ELEMENT author (#PCDATA)>
<!ELEMENT book (title, author,
  description)>
<!ATTLIST book id ID #IMPLIED>
<!ELEMENT description (#PCDATA)>
<!ELEMENT publications (book+,
  reviews)>
<!ELEMENT reference EMPTY>
<!ATTLIST reference idRef IDREF
  #REQUIRED>
<!ELEMENT review (author, reference)>
<!ELEMENT reviews (review+)>
<!ELEMENT title (#PCDATA)>
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE publications SYSTEM "Untitled.dtd">
<publications>
  <book id="Text">
          <title>Text</title>
          <author>Text</author>
          <description>Text</description>
  </book>
  <book id="Text1">
          <title>Text</title> <author>Text</author>
          <description>Text</description>
  </book>
  <reviews>
          <review>
               <author>Text</author> <reference idRef="Text"/>
          </review>
          <review>
              <author>Text</author> <reference idRef="Text1"/>
          </review> </reviews>
</publications>
```

Attribute-List Declarations V

- **ENTITY:**
  - Utilisation d'une donnée à des multiples endroits
  - Servir de remplaçant pour les caractères réservés ou spéciaux
- **NMTOKEN:** `CDATA` avec des restrictions dans l'utilisation des caractères
  - `NMTOKENS`: contient une liste de `NMTOKEN` délimités par des espaces vides
- **NOTATION:** Référence à des données non XML avec information sur la manière dont elles doivent être traitées
- **ENUMERATION:** Limite la liste des valeurs d'un attribut
  - Ex. `<!ATTLIST pubDate year (1999 | 2000 | 2001 | unknow) "unknow">`

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Fund SYSTEM "Fund.dtd">

<Fund>
 <Name>XYZ Fund Global Growth</Name>
 <NumberShares>22</NumberShares>
 <DataProvider>&LIP;</DataProvider>
</Fund>
```

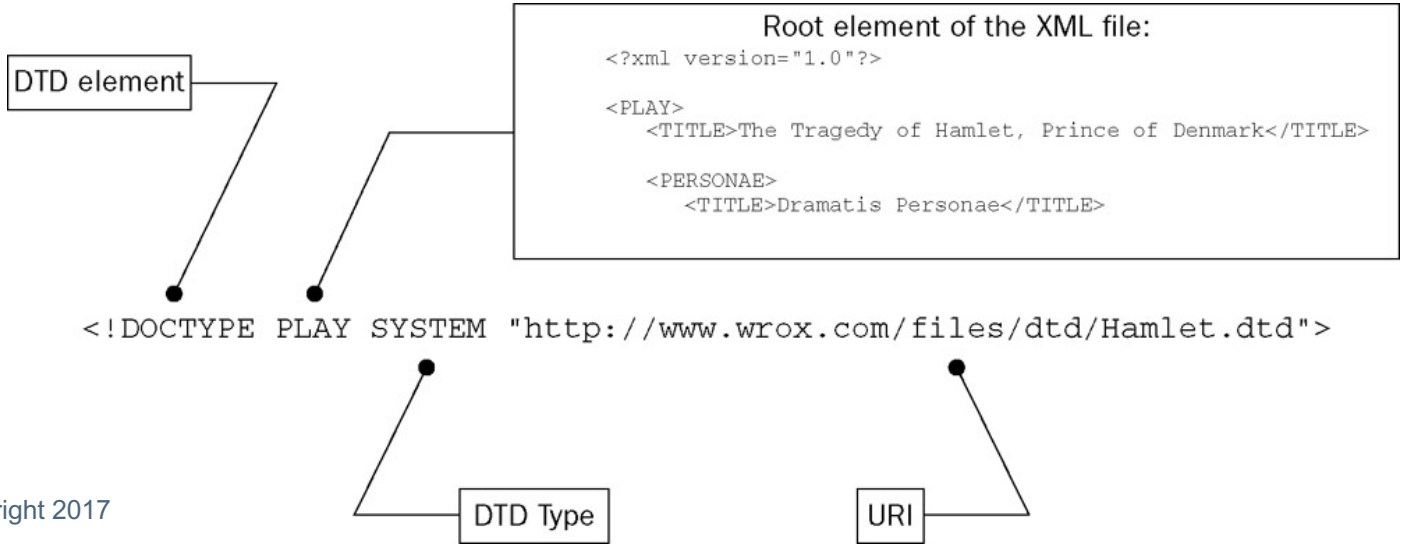
```
<?xml version="1.0" encoding="UTF-8"?>

<!ELEMENT Fund (Name, NumberShares, DataProvider)>
<!ELEMENT Name (#PCDATA)>
<!ELEMENT NumberShares (#PCDATA)>
<!ELEMENT DataProvider (#PCDATA)>
<!ENTITY LIP "Lipper Inc., A Reuters Company">
```

Attribute-List Declarations VI

- Default types
  - Il existe quatre options:
    - ✓ **REQUIRED** : exige la présence de l'attribut
      - `<!ATTLIST element_name attribute_name datatype #REQUIRED>`
    - ✓ **IMPLIED** : l'attribut est optionnel
      - `<!ATTLIST element_name attribute_name datatype #IMPLIED>`
    - ✓ **FIXED** : l'attribut est fixe
      - `<!ATTLIST element_name attribute_name datatype #FIXED "fixed_value">`
    - ✓ **"value"** : valeur par défaut
      - `<!ATTLIST element_name attribute_name datatype "default_value">`
- Multiple Attributes
  - Ex. `<book isbn="01234556" edition="1" cat="XML XHTML HTML" id="test">`
  - Déclaration :
    - ✓
```
<!ATTLIST book isbn CDATA #REQUIRED
              edition CDATA #REQUIRED
              cat NMTOKENS #REQUIRED
              id ID #IMPLIED>
```

DTD externes

- DTD interne : plus de lisibilité
- DTD externe : plus performant comme approche du fait qu'une DTD n'est pas dédiée à un seul document mais plusieurs
- La DTD interne est prioritaire sur la DTD externe:
  - ✓ DTD externe: déclaration globale
  - ✓ DTD interne: déclaration spécifique



Les composants d'un XML - I

- Un document XML contient du texte
- Ce n'est pas obligatoirement un fichier, il peut être :
  - Stocké dans une base de données
  - Créé à la volée en mémoire par un programme
  - Être la combinaison de plusieurs fichiers imbriqués
  - Ne jamais exister sous forme de fichier permanent
- Un XML est constitué d'unités élémentaires de stockage nommées entités
  - Une entité contient
    - ✓ Soit du texte (des caractères)
    - ✓ Soit des données binaires (des images, etc.)
    - ✓ Jamais les 2 à la fois pour le même élément

Les composants d'un XML - II

- Syntaxe d'une déclaration XML:
  - **<?xml version ="version_number" encoding="encoding_declarations" standalone="standalone_status"?>**
  - *Exemple:* `<?xml version="1.0" encoding="UTF-8" standalone="no"?>`

nom	valeur	Descriptions
Version_number	1.0	Indique la version
Encoding_declarations	UTF-8, UTF-16, ISO 8859-1 to ISO 8859-9, UCS-4, etc.	Les jeux de caractères: - ASCII codage entre 0 et 127 (7 bits) - ISO 8859-1 (Latin/1) entre 0 et 225 - ISO 8859-1 = ASCII + caractères des langues européennes - ISO 8859-2 = ASCII + caractères des langues de l'europe centrales - Etc. - Unicode codage 16 bits - UTF-8 Unicode compressé: au lieu de deux octets pour chaque caractère, on utilise un seul <?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?> Pour supporter les accents
Standalone_status	yes,no	S'il existe une DTD externe ou une entité externe ➔ **"standalone='no' "** S'il existe uen DTD interne ➔ **" standalone='yes' ".**

Les composants d'un XML - III

Référence d'entité	*Caractère symbolisé*
&amp;	&
&lt;	<
&gt;	>
&quot;	"
&apos;	'

- Les commentaires (comme html)
- Références d'entités symboliques
  - Les caractères (&), (<), etc. peuvent apparaître sous leur forme littérale seulement quand ils sont utilisés comme délimiteurs de balisage ou dans un commentaire, une instruction de traitement, ou une section CDATA.

```
<Value>Do if 5 < 3</Value>
```
✗

```
<Value>Do if 5 &lt; 3</Value>
```
✓

```
<Value><![CDATA[Do if 5 < 3]]></Value>
```
✓

Les composants d'un XML - IV

- Les sections littérales **CDATA**
  - Les sections de type `CDATA` sont utiles lorsque on veut empêcher toute interprétation de ce qu'elles contiennent (les méta-caractères)
  - Ex:

```
<![CDATA[
<?xml version="1.0" standalone="yes"?>
<salutation>
 Bonjour les enfants
<salutation>
]]>
```

  - Pas de possibilité d'imbrication de `CDATA`
- Les balises
  - `<salutation> … </salutation>`
  - Les balises vides : `<salutation/>`
  - Les noms des balises:
    - ✓ On commence toujours par une lettre de l'alphabet ou par (_)
    - ✓ Les autres caractères peuvent être : chiffres, lettres, _ , tiré, les points
    - ✓ Les espaces entre les mots sont interdites

<_8balise>

<Section.paragraphe>

<Stefano_carrino>

XML "well-formed"

- Les documents **bien formés** se conforment à toutes les contraintes applicables et ne font pas directement ou indirectement la référence aux déclarations externes.
- S'il n'y a aucune déclaration de type de document (DTD) en service, le document doit commencer par une déclaration autonome.
- Pourquoi XML "well-formed"
  - On est le seul auteur du document
    - ✓ Plus de flexibilité
  - La validation demande plus de ressources
    - ✓ Si une application génère automatiquement le document
      - – Moins de source d'erreurs

XML "valide"

- Il est « **well-formed** » et il a une DTD attachée à lui (incluse ou référencée)
- Il est **conforme à cette DTD**
  - Pas d'attributs supplémentaires
  - Éléments ordonnés et imbriqués correctement
- La validation passe par les parseurs qui détectent (I):
  - Le document doit commencer par une déclaration XML.
  - Toute balise ouverte est fermée.
  - Les balises vides terminent avec />
  - Contient un seul élément racine
  - …

XML "valide"

- La validation passe par les parseurs qui détectent (II):
  - Les valeurs des attributs doivent être limitées par des guillemets
  - Les balises peuvent être imbriquées et non pas mélangées

```
<chaos>This is <disaster>not</chaos> well formed</disaster>
```

  - Les deux méta-caractères < et & sont utilisés pour démarrer respectivement une balise et une référence d'entité
  - Les seuls références d'entités qui doivent apparaître sont :
    - ✓ & < > " '

XML "valide"

- Pourquoi un XML valide ?
  - Pour standardiser dans un cadre du multi-auteurs
  - Faciliter l'échange entre les applications
  - Pas besoin de programmer pour détecter les erreurs (le parseur n'inclus pas les documents non conformes)
    - ✓ Ex: une société hiérarchisée
      - Dept. Finance
      - Dept. Marketing
      - Dept. Juridique
      - Etc.

Instance XML - bien formé? valide?

```
<?xml version="1.0" standalone=yes"?>
<!DOCTYPE MEMO [
<!ELEMENT MEMO     (TO,FROM,SUBJECT,BODY,SIGN)>
 <!ATTLIST MEMO     importance   (HIGH|MEDIUM|LOW) "LOW">
 <!ELEMENT TO       (#PCDATA)>
 <!ELEMENT FROM     (#PCDATA)>
 <!ELEMENT SUBJECT  (#PCDATA)>
 <!ELEMENT BODY     (P+)>
 <!ELEMENT P        (#PCDATA)>
 <!ELEMENT SIGN     (#PCDATA)>
 <!ATTLIST SIGN     signatureFile CDATA #REQUIRED
                    email         CDATA #IMPLIED>
 ]>
<MEMO importance="HIGH">
      <TO>Tutorial Takers</TO>
      <FROM>Tutorial Writer</FROM>
      <SUBJECT>Your impressions</SUBJECT>
      <BODY>
        <P>Now that you are almost done the tutorial, you must be getting an idea of what XML is about. These
            emerging technologies sometimes  take time though before they catch on.</P>
        <P>Did you find the tutorial helpful? Which areas did you find confusing? How would you improve them?</P>
      </BODY>
      <SIGN email="rlander@pdbeam.uwaterloo.ca">Richard</SIGN>
</MEMO>
```

<!DOCTYPE début déclaration
NAME nom
SYSTEM indication d'existence d'une DTD
"report.dtd » Le système doit être conforme à cette DTD
[ ] si c'est une DTD interne
> fin déclaration

Conclusion et résumé

- Apprendre le processus de conception d'un document structuré
  - ✓ XML orienté data vs document
- Modéliser avec une DTD
  - ✓ Déclaration d'élément
    - Content model: Element, Mixed, EMPTY, ANY, occurrences, etc.
  - ✓ Déclaration d'attribut
    - Datatype (CDATA, ID, IDREF, etc.), type (REQUIRED, IMPLIED, etc.)
  - ✓ Entité et notation
  - ✓ DTD interne et externe
- Construire un fichier XML
  - ✓ Structure et syntaxe
  - ✓ "well-formed" vs "valide"