



# Project seminar: Software Technology for Distributed and Mobile Applications

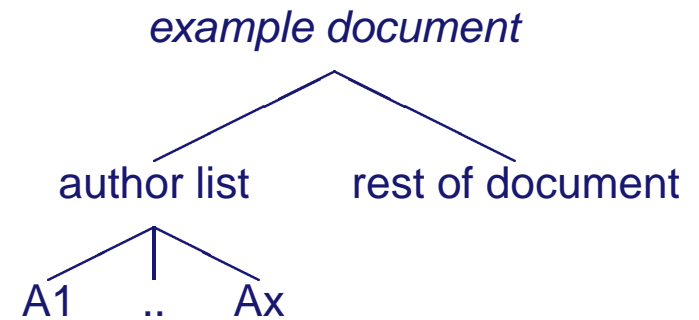
*DTD / XML Schema*

Lars Bo Thulin



# Markup language description

**DTD**, and **XML Schema** is used to *describe* markup languages





# Quick DTD example

```
<?xml version="1.0" ?>

<!DOCTYPE example [

    <!ELEMENT example (authorlist, therest)>
    <!ELEMENT authorlist (author+)>
    <!ELEMENT author (#PCDATA)>
    <!ELEMENT therest (#PCDATA)>

]>

<example>
  <authorlist>
    <author>
      H. C. A.
    </author>
    <author>
      Anders A.
    </author>
  </authorlist>
  <therest>
    this is the rest of the text
  </therest>
</example>
```

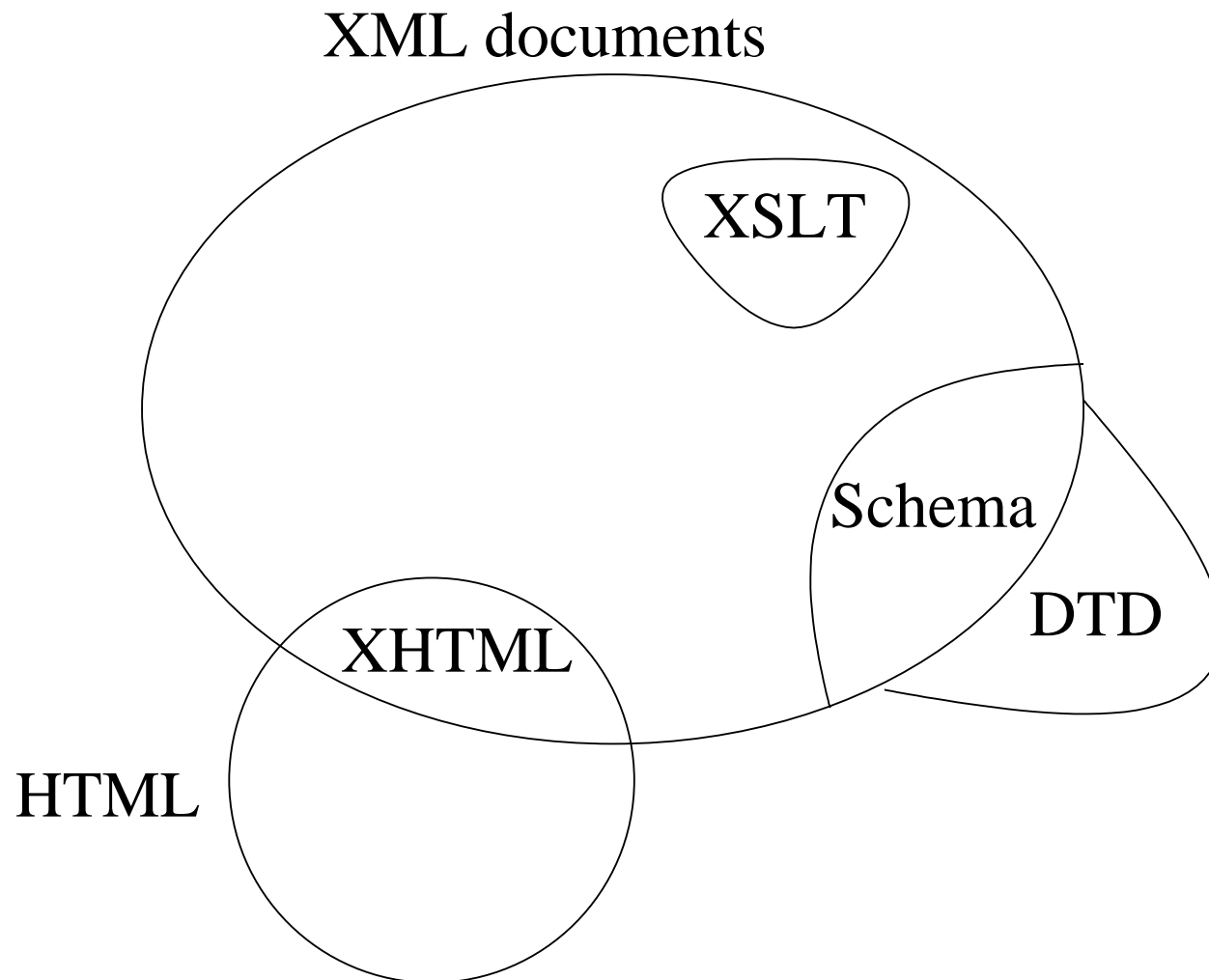


# Valid documents

With a DTD or a Schema it is possible to make a validation of a document  
fx. use the *MarkUp Validation Service* at <http://validator.w3.org/>  
or use a local program downloaded at <http://www.w3.org/XML/Schema>



# Some bubbles





# What is possible with DTD

It is possible to define elements

- only containing text  
`<!ELEMENT name (#PCDATA)>`
- from other elements  
`<!ELEMENT callname (name)>`
- by sequences of other elements  
`<!ELEMENT names (name,callname)>`
- containing alternative elements  
`<!ELEMENT nametag (name|callname)>`
- with variable number of other elements  
`<!ELEMENT employee ((title*,name+,id?,initials)|id)>`
- with attributes (metadata)  
`<!ATTLIST employee special (yes|no) #IMPLIED >`



# What is possible with DTD - cont.

It is possible to declare the DTD

- **internal** in the XML file (in a blob to avoid destruction of the wellformedness)  
-like in the quick DTD example
- **external** in a separate file
  - named by a FPI: Formal Public Identifier
  - referenced by both FPI and URL/URI

FPI is used to find the newest version of the DTD

If no public accessible server knows the FPI then the URL is used.

Entities is used as shorthands in DTDs and XML documents. Binary data as pictures are inserted as non-parsed entities.



# Quick DTD example - external

file **example.dtd**:

```
<!ELEMENT example (authorlist, therest)>
<!ELEMENT authorlist (author+)>
<!ELEMENT author (#PCDATA)>
<!ELEMENT therest (#PCDATA)>
```

file **example.xml**:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE example SYSTEM "http://www....example.dtd">

<example>
  <authorlist>
    <author>
      H. C. A.
    </author>
    <author>
      Anders A.
    </author>
  </authorlist>
  <therest>
    this is the rest of the text
  </therest>
</example>
```



# What is *not* possible with DTD

- local elements
- constraining the type of information in elements (like integers in the range 1 to 12)



# XML Schema, central concepts

A Schema consists mainly of: element declarations, global and local type definitions

elements have types:  
types are simple or complex



# Quick example revisited

file example.xsd:

```
<?xml version="1.0" ?>
<xsd:schema xmlns:xsd= "http://www.w3.org/2000/10/XMLSchema">

<xsd:element name="example">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="authorlist" type="authorlisttype"/>
      <xsd:element name="therest" type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>

<xsd:complexType name="authorlisttype">
  <xsd:list>
    <xsd:element name="author" type="xsd:string"/>
  </xsd:list>
</xsd:complexType>

</xsd:schema>
```



## example - cont.

file example.xml:

```
<?xml version="1.0" ?>

<example xmlns:xsi= "http://www.w3.org/2000/10/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://www.diku...example.xsd">

<example>
  <authorlist>
    <author>
      H. C. A.
    </author>
    <author>
      Anders A.
    </author>
  </authorlist>
  <therest>
    this is the rest of the text
  </therest>
</example>
```



# links

Go to <http://www.w3.org/XML/Schema> to see:

- a list of XML Schema relevant tools
- a list of XML Schema usages
- **specifications** – I have used *XML Schema Part 0: Primer* as an introduction, whereas the *XML Schema Part 1: Structures* and *XML Schema Part 2: Datatypes* forms a complete normative description of the XML Schema language



# Exchange of data in XML documents

When more than one actor exchange or work on some data it is important to use standardized Schemas to avoid the data impedance problem

Example:

Den danske regering ønsker at digitalisere den offentlige sektor. At indføre en digital forvaltning.

på sigt skal alle offentlige myndigheder udveksle information med hinanden og andre ved brug af XML.

en infostrukturbase - (med) et website - er oprettet i forbindelse med fastlæggelse af de fornødne XML standarder

it will be possible to look up all public data interface descriptions, and gain information about what data are available and how data are accessed.



# oio

some examples from the repository containing XML schemas, schema fragments, interface descriptions and process descriptions

See more at <http://isb.oio.dk/info/index.htm>

```
<simpleType name="CivilRegistrationNumberType">
  <restriction base="string">
    <pattern value="(((0[1-9]|1[0-9]|2[0-9]|3[0-1])(01|03|05|07|08|10
      |((0[1-9]|1[0-9]|2[0-9]|30)(04|06|09|11))
      |((0[1-9]|1[0-9]|2[0-9])(02))))
      [0-9]{6})|0000000000"/>
  </restriction>
</simpleType>
```

```
<simpleType name="StreetNameForAddressingType">
  <restriction base="string">
    <minLength value="1"/>
    <maxLength value="20"/>
  </restriction>
</simpleType>
```