XML: namespaces

How do you avoid tag conflicts?

Since you can define your own tags, if you reuse XML files from other authors you might find tag conflicts.

These can be avoided by declaring a namespace as an attribute of the root element:

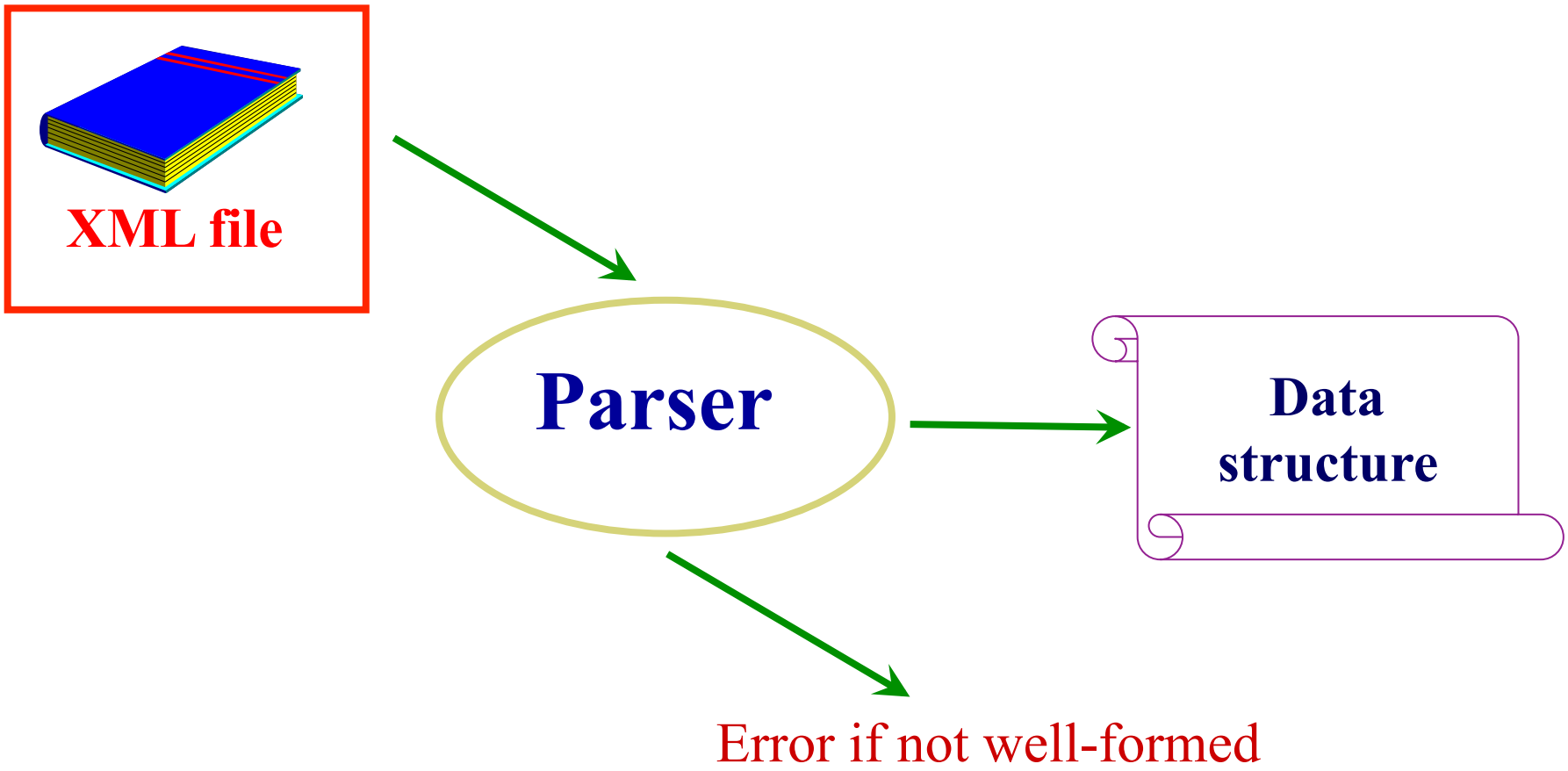
<xsl:stylesheet version =“1.0” xmlns:xsl=“http://www.w3.org/1999/XSL/Transform”>

What is a parser?

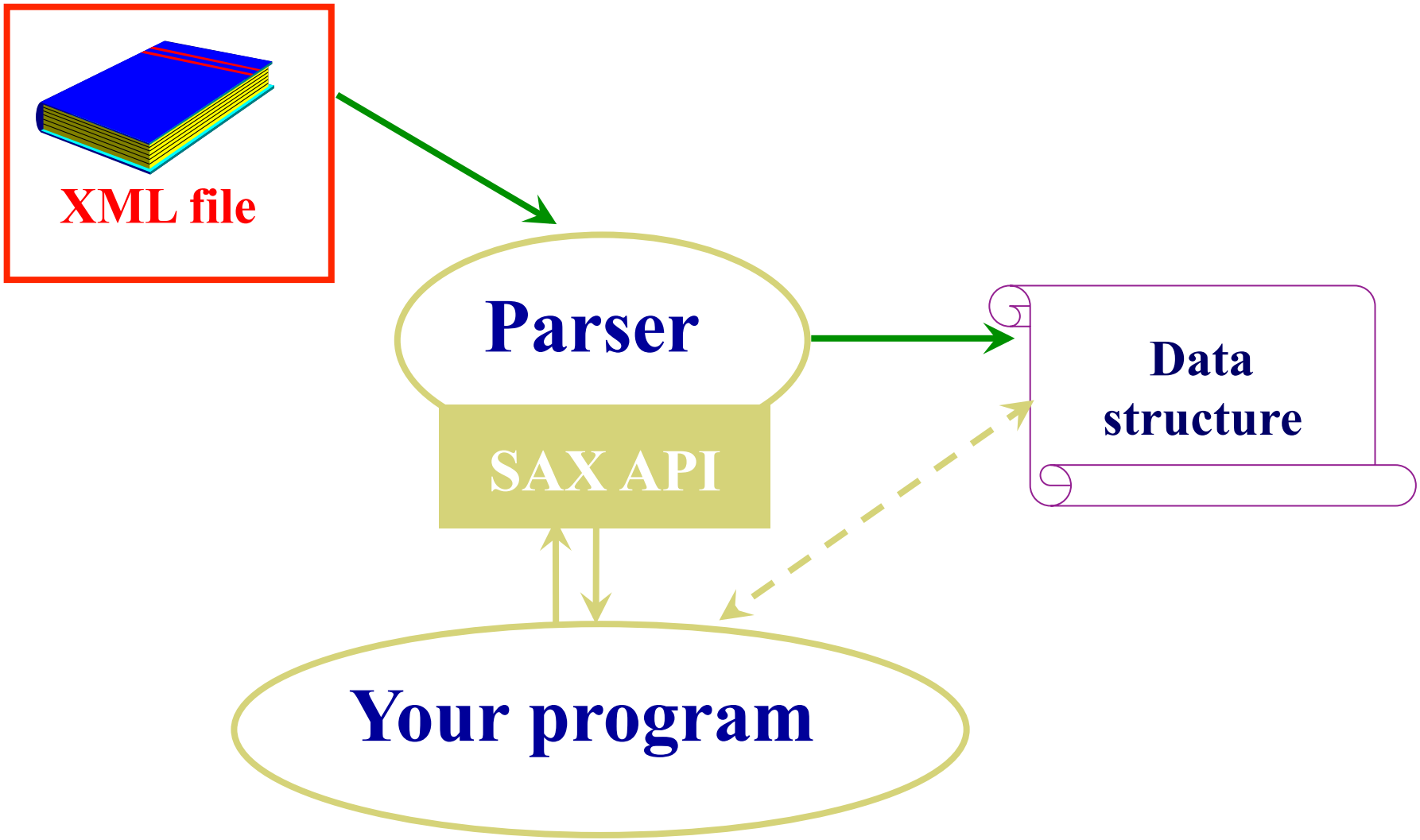
A parser, in this context, is a software tool that preprocesses an XML document in some fashion, handing the results over to an application program.

The primary purpose of the parser is to do most of the hard work up front and to provide the application program with the XML information in a form that is easier to work with.

Making sense of XML: the Parser



Making sense of XML:the Parser



Tree-based vs Event-based API

- **Tree-based API**
  A tree-based API compiles an XML document into an internal tree structure. This makes it possible for an application program to navigate the tree to achieve its objective. The Document Object Model (DOM) working group at the W3C is developing a standard tree-based API for XML.
- **Event-based API**
  An event-based API reports parsing events (such as the start and end of elements) to the application using *callbacks*. The application implements and registers event handlers for the different events. Code in the event handlers is designed to achieve the objective of the application. The process is similar (but not identical) to creating and registering event listeners in the Java Delegation Event Model.

what is SAX?

SAX is a set of interface definitions

For the most part, SAX is a set of interface definitions. They specify one of the ways that application programs can interact with XML documents.

(There are other ways for programs to interact with XML documents as well. Prominent among them is the Document Object Model, or DOM)

SAX is a standard interface for event-based XML parsing, developed collaboratively by the members of the XML-DEV mailing list. SAX 1.0 was released on Monday 11 May 1998, and is free for both commercial and noncommercial use.

The current version is SAX 2.0.1 (released on 29-January 2002)

See http://www.saxproject.org/

Some available Parser

Apache Xerces	http://xml.apache.org
IBM XMLJ4	http://alphaworks.ibm.com/tech/xmlj4
James Clark's XP	http://www.jclark.com/xml/xp
OpenXML	http://www.openxml.org
Oracle XML Parser	http://technet.oracle.com/tech/xml
Sun Microsystem Project X	http://java.sun.com/products/xml
Tim Bray's Lark and Larval	http://www.textuality.com/Lark

Introduction to XML

DTD

What is a DTD?

A DTD is usually a file (or several files to be used together) which contains a formal definition of a particular type of document. This sets out what names can be used for elements, where they may occur, and how they all fit together.

It's a formal language which lets processors automatically parse a document and identify where every element comes and how they relate to each other, so that stylesheets, navigators, browsers, search engines, databases, printing routines, and other applications can be used.

A DTD contain metadata relative to a collection of XML docs.

Valid documents

a *valid* XML document is one that conforms to an existing DTD in every respect.

For example...

Unless the DTD allows an element with the name *"color"*, an XML document containing an element with that name is not valid according to that DTD (but it might be valid according to some other DTD).

An *invalid* XML document can be

a perfectly good and useful XML document.

Valid documents

Validity is not a requirement of XML

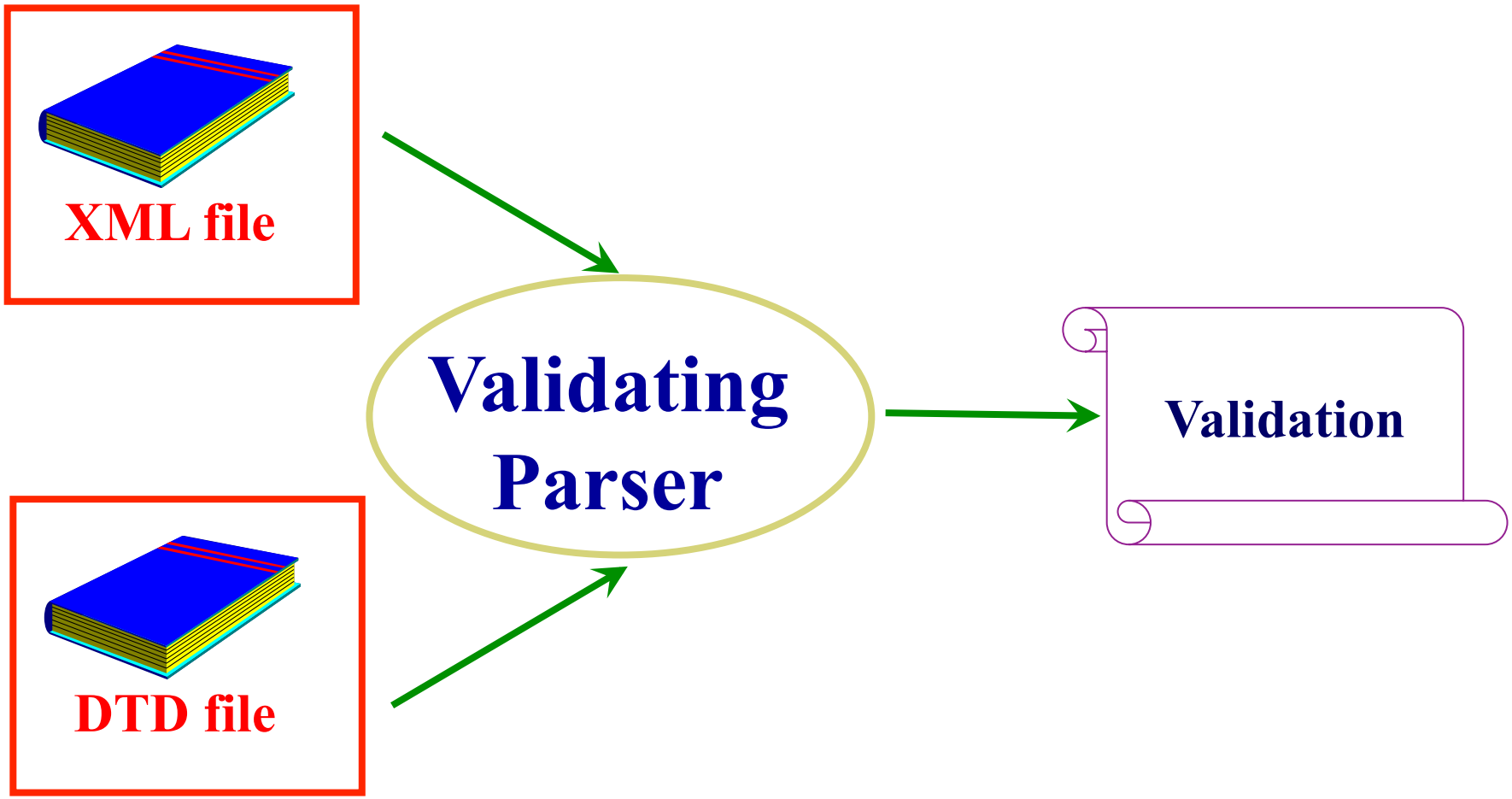
Because XML does not require a DTD, in general, an XML processor cannot require validation of the document.

Many very useful XML documents are not valid, simply because they were not constructed according to an existing DTD.

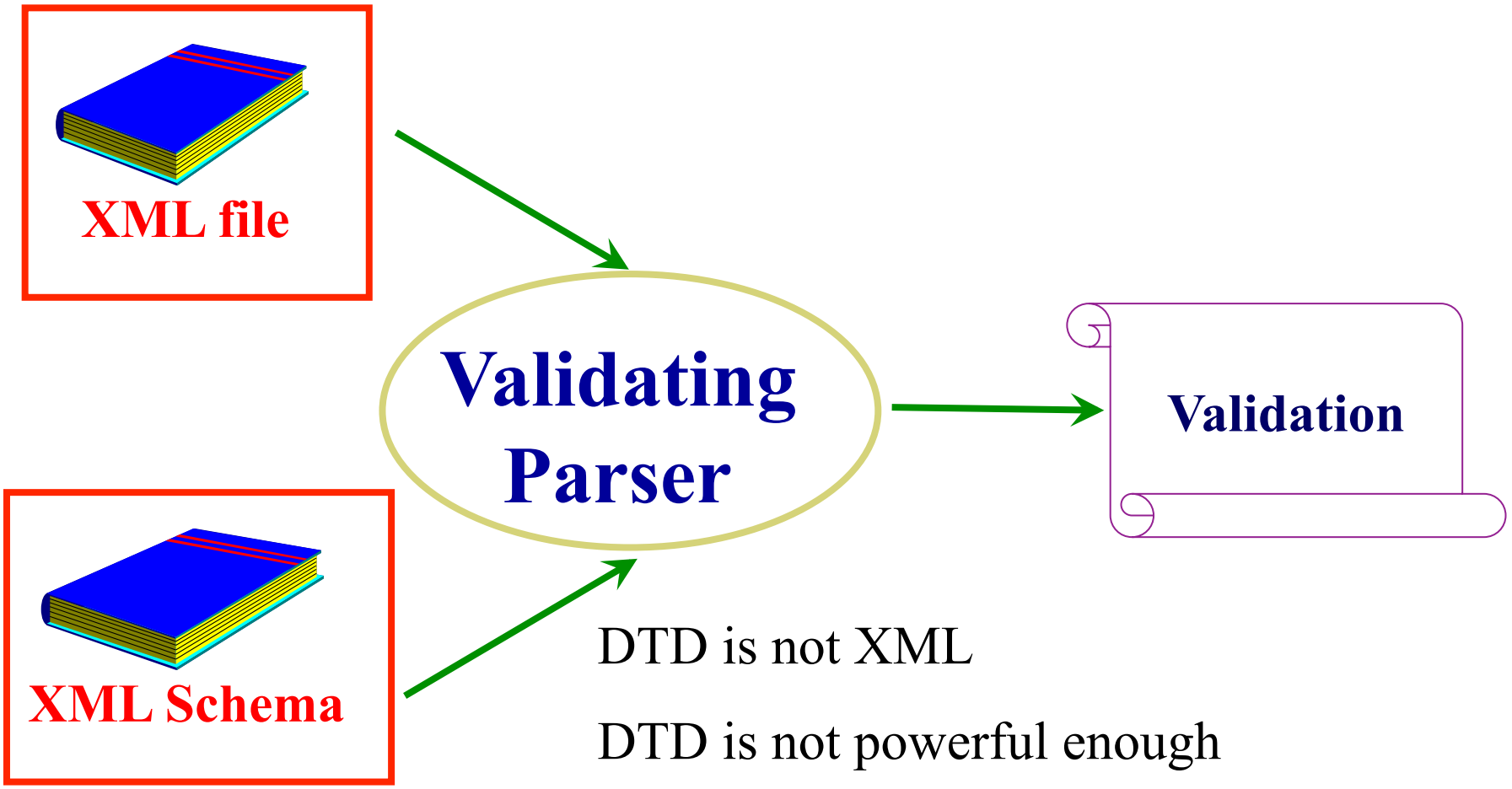
To make a long story short,

validation against a DTD can often be very useful, but is not required.

Constraing &ValidatingXML



Constraing & Validating XML



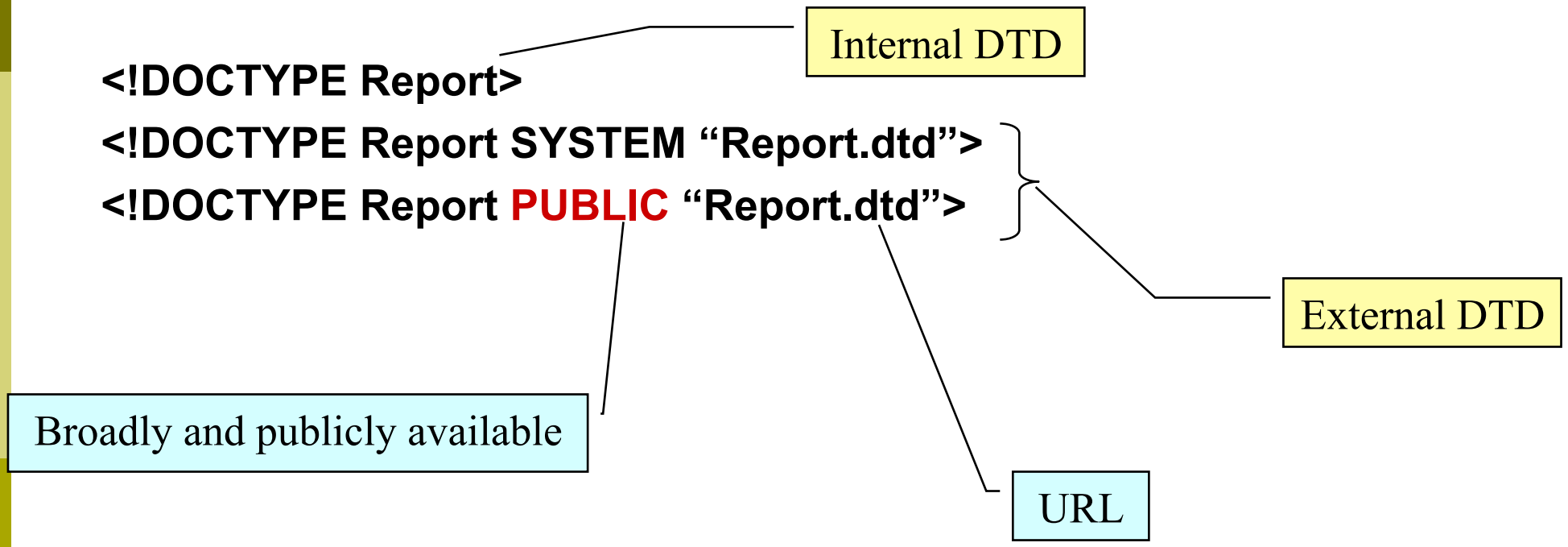
DTD is not XML

DTD is not powerful enough

(e.g. at least 3, no more than 5)

Where are the DTDs?

A DTD can be external or internal to a document.



DTD Markup: ELEMENT

```
<!ELEMENT name content-model>
<!ELEMENT book (preface?,chapter+,index)>
<!ELEMENT preface(paragraph+)>
<!ELEMENT paragraph (#PCDATA)>

<!ELEMENT chapter (title,paragraph+,reference*)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT reference (#PCDATA|URL)>
<!ELEMENT URL (#PCDATA)>

<!ELEMENT index(number,title,page_number)>
<!ELEMENT number(#PCDATA)>
<!ELEMENT page_number(#PCDATA)>
```

? Zero or one
+ One or more
* Zero or more
, sequence
| or (not xor!)

DTD Markup: ATTLIST

```
<!ATTLIST element-name attribute-name type default>
<!ELEMENT Product (#PCDATA)>
<!ATTLIST Product
    Name CDATA #IMPLIED
    Rev CDATA #FIXED "1.0"
    Code CDATA #REQUIRED
    Pid ID #REQUIRED
    Series IDREF
    Status (InProduction|Obsolete)
        "InProduction"
>
```

TYPES:
CDATA character data
ID Unique key
IDREF Foreign Key
(…|…) Enumeration

DEFAULT:
#IMPLIED optional, no default
#FIXED optional, default supplied.
If present must match default
#REQUIRED must be provided

DTD Markup: ENTITY

Entities are a sort of macro

General Entity

<!ENTITY author “Marco Ronchetti, Universita’ di Trento”>

External Parsed Entity

<!ENTITY content SYSTEM “content.xml”>

<Tag>&content &author</Tag>

External to the DTD

Parameter Entity

<!ENTITY % AI “CDATA #IMPLIED”>

<!ATTLIST Product Name %AI>

Internal at the DTD

The main problem of DTD's...

They are not written in XML!

Solution:

Another XML-based standard: XML Schema

For more info see:

http://www.w3.org/XML/Schema