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

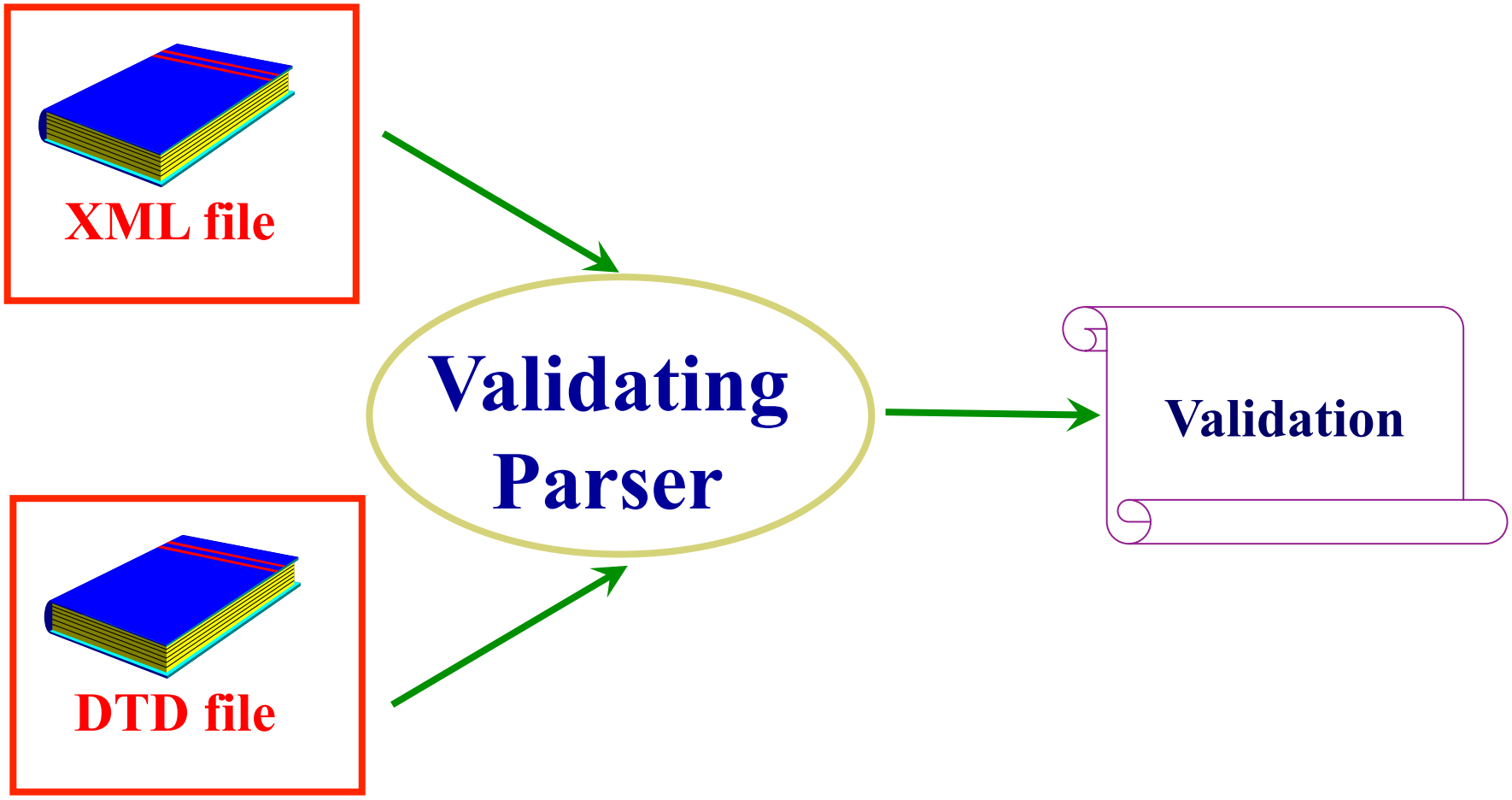
Because XML does not require a DTD, in general, an XML processor cannot require validation of the document.

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

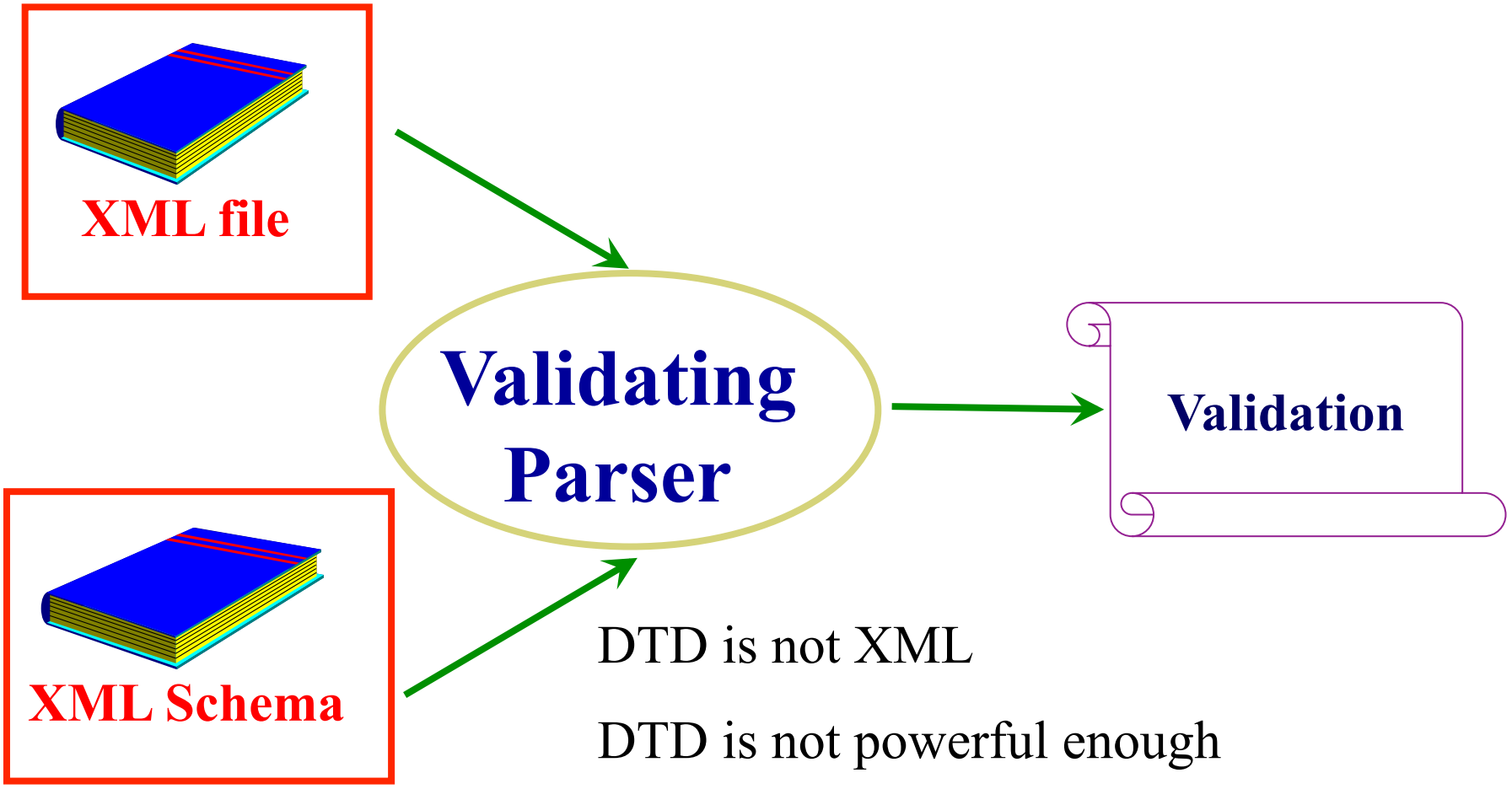
To make a long story short,

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

Constraing &ValidatingXML



Constraing & Validating XML



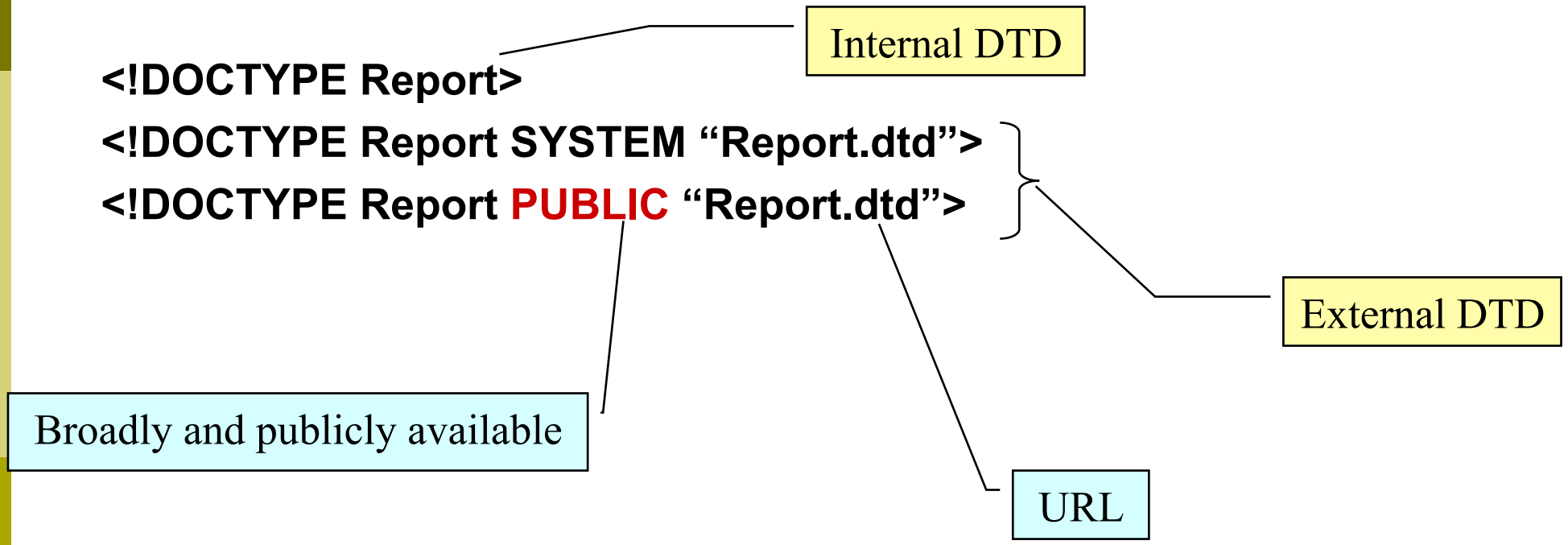
DTD is not XML

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:

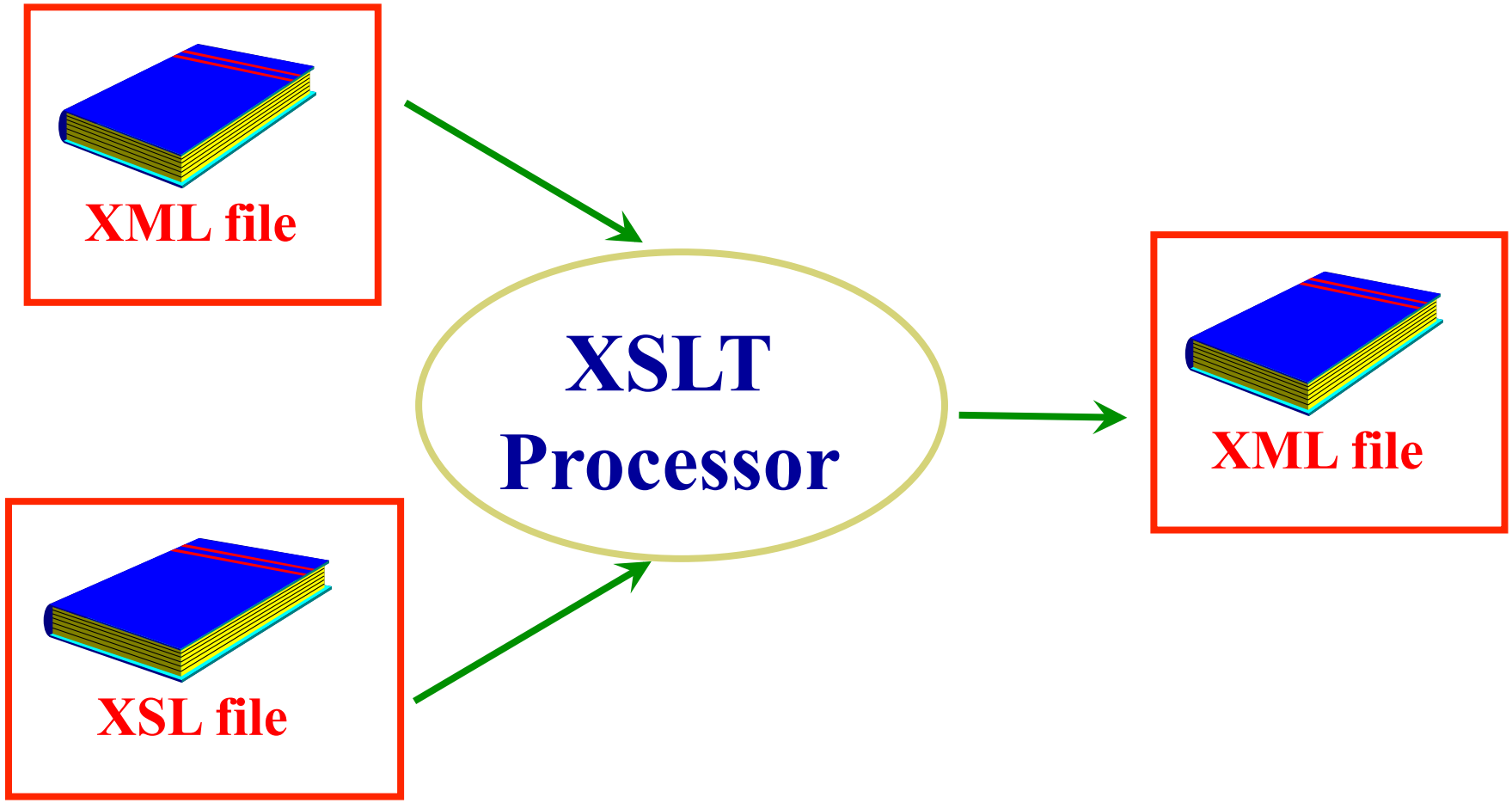
Another XML-based standard: XML Schema

For more info see:
http://www.w3.org/XML/Schema

Introduction to XML

XSL - INTRODUCTION

Transforming XML



XSL is complex

XSL is complex (much more complex than XML).
- **Designing an XSL stylesheet, to be used by a rendering engine to properly render an XML document, can be a daunting task.**

Microsoft has developed an XSL debugger, and has made it freely available for downloading.