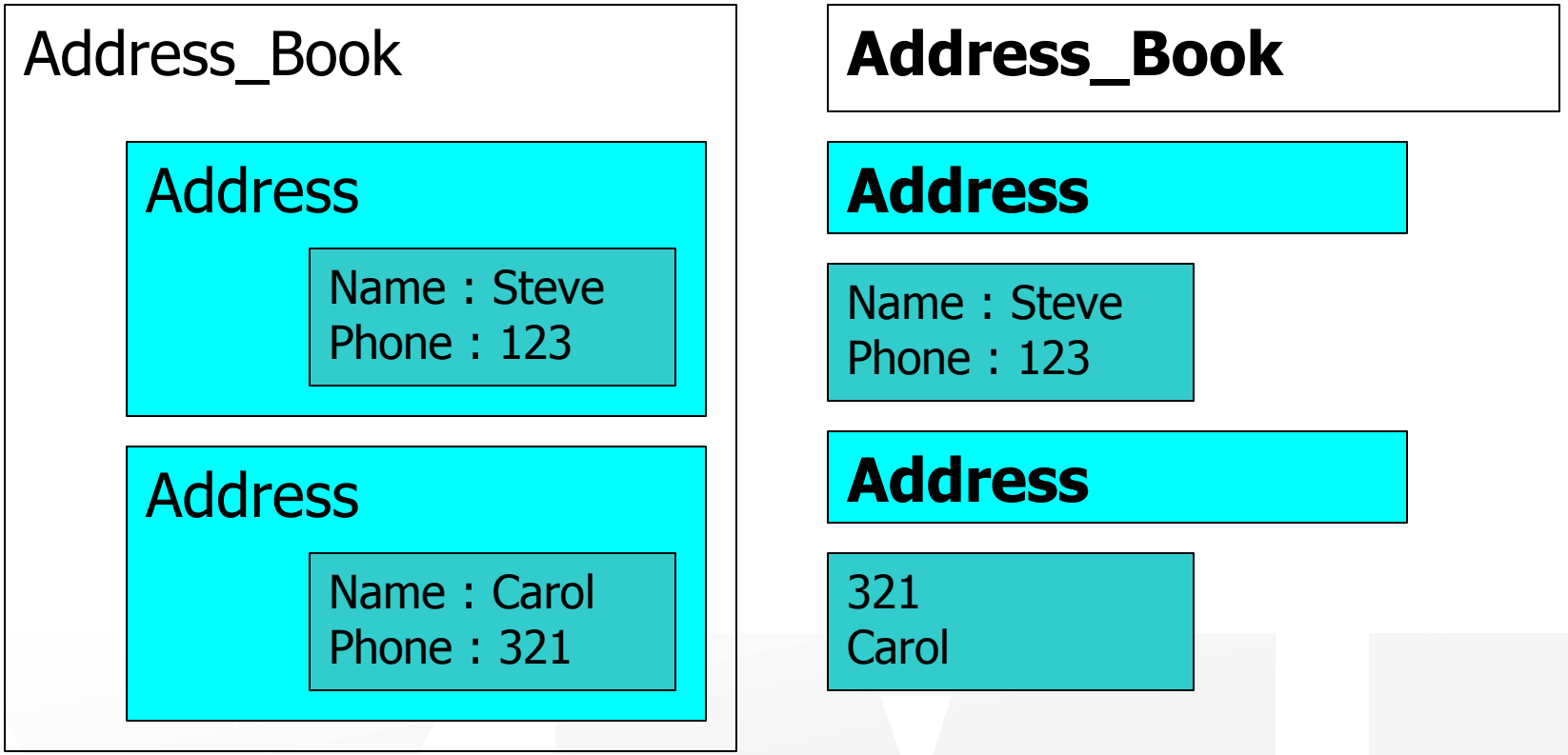
Session [1]

- **XML Introduction / related technologies**

- September 6, 2000
  Horst Rechner

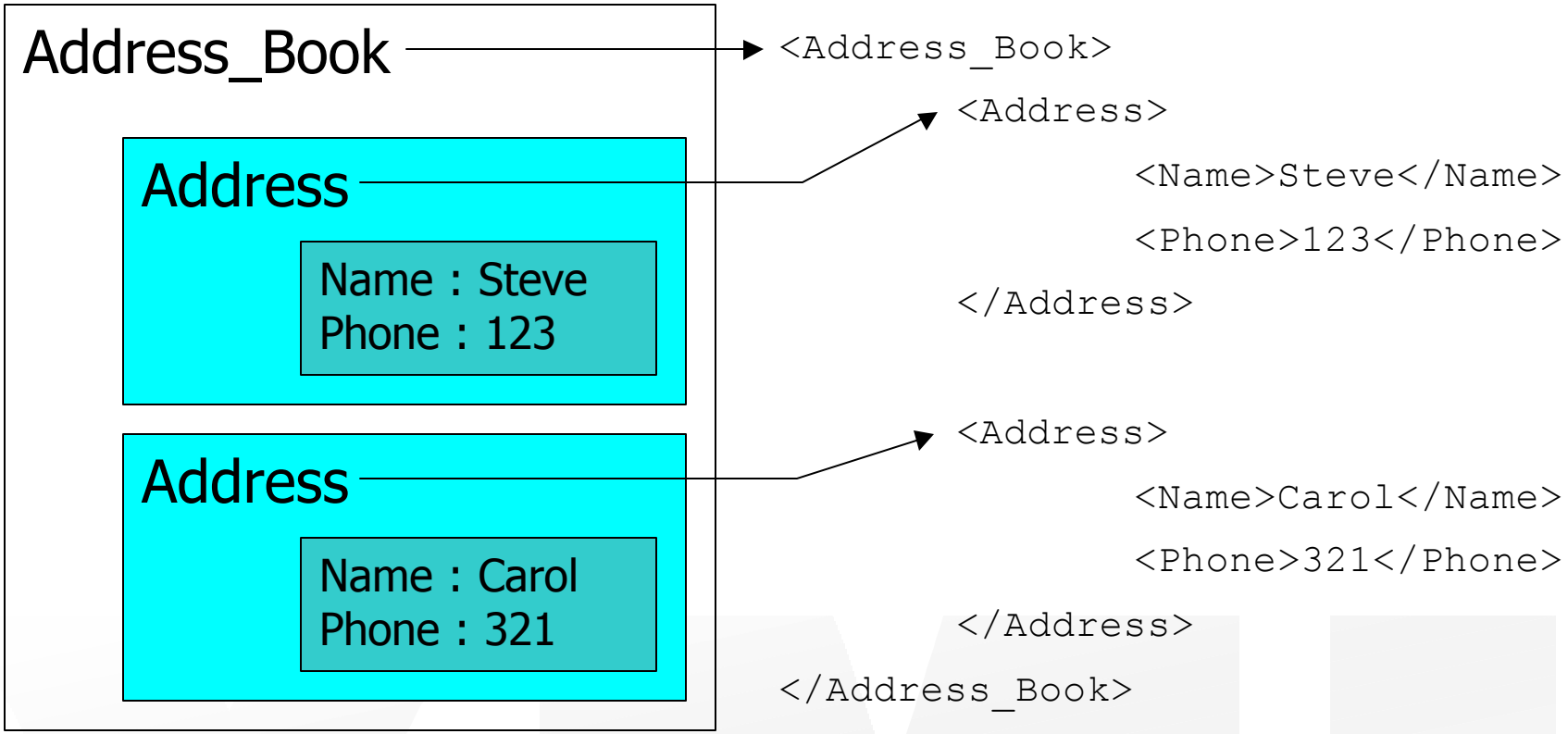
What is XML?

- XML is only a data format !
- XML is structured data (against unstructured HTML)



How does it look?

- Like HTML, but you define the tags.



Why XML?

- Why not HTML?
  - Already overburdened with dozens of inventions from different manufacturers
  - Has a fixed DTD, that was created to describe the layout, not the meaning of the text.
- When it's data, why not use database tables and SQL?
  - Table structure is too inflexible to describe hierarchical data.

Strengths of XML in data-exchange

- Robust data representation
  - Easy to capture metadata
- Easily mutable
  - APIs are easily accessed by existing code
- Platform independent
  - APIs work in all major platforms
- Works with existing technologies
  - SQL, HTML, JAVA
- Asynchronous transfer

Strengths of XML on the web

- Separation of data and design
  - Rendering can be done on the client or server side
  - Design can be reused with different data
    or
    Data can be reused with different designs

Weaknesses of XML in data-exchange

- Amount of exchanged data increases in comparison to traditional electronic data interchange

Weaknesses of XML on the web

- You have to write well-formed XHTML

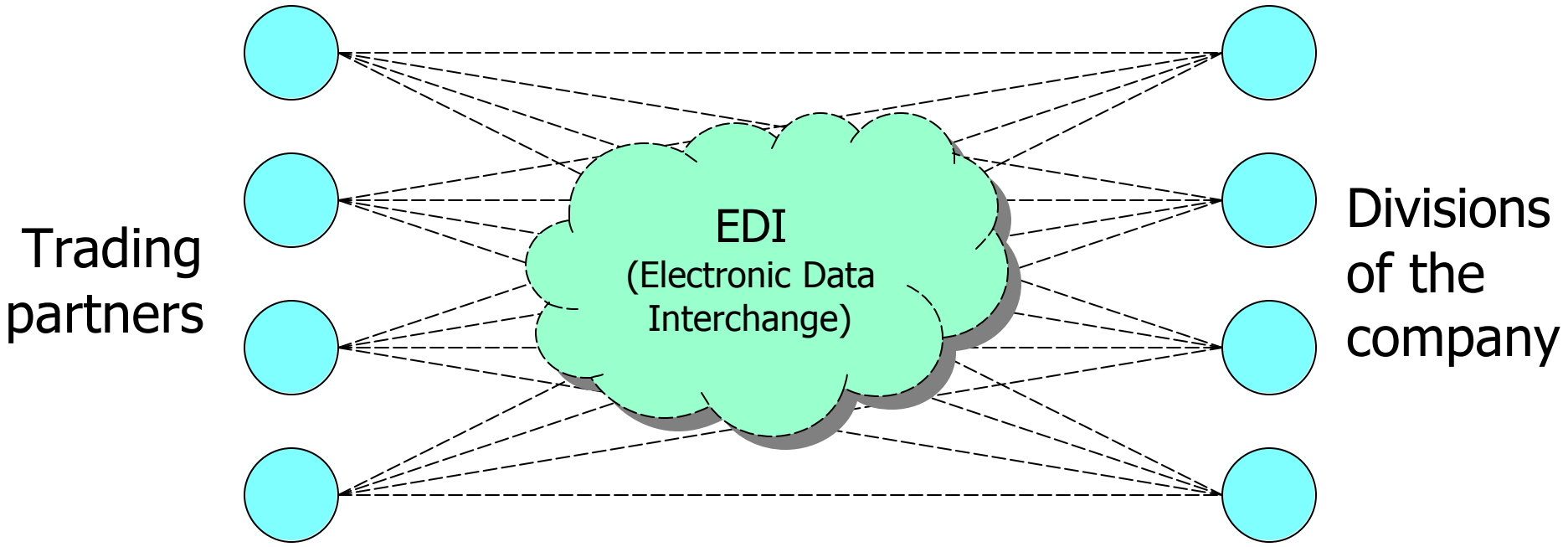
EAI

- Enterprise Applications Integration is…
- "Integration of business processes for the purpose of conducting electronic business"
- "Sharing/Exchange of data between systems to provide a unified interface

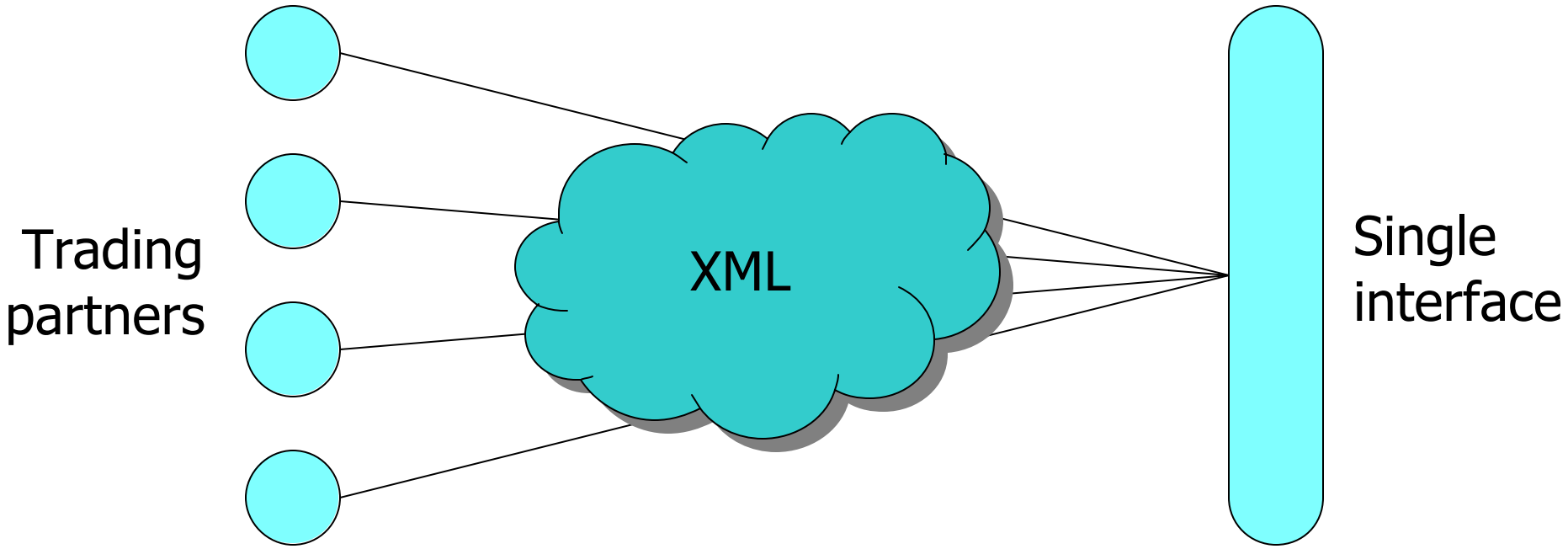
XML + EAI = XAI

- Need to provide a single representation and unified access of enterprise data
- XML bridges the gap between meanings
- Mapping from one meaning to another

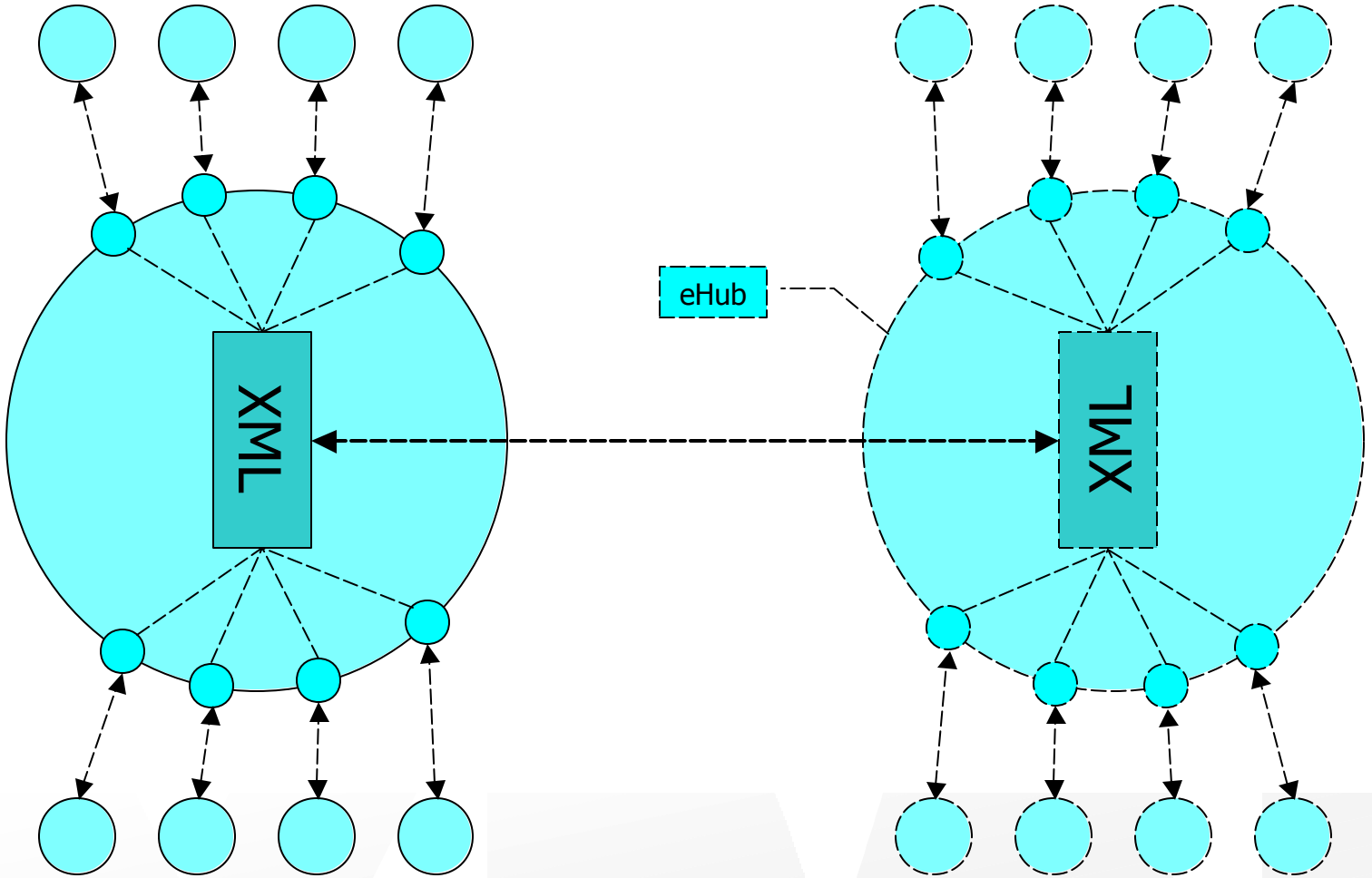
Step 1: Many to Many Solution



Step 2: Single Face Solution

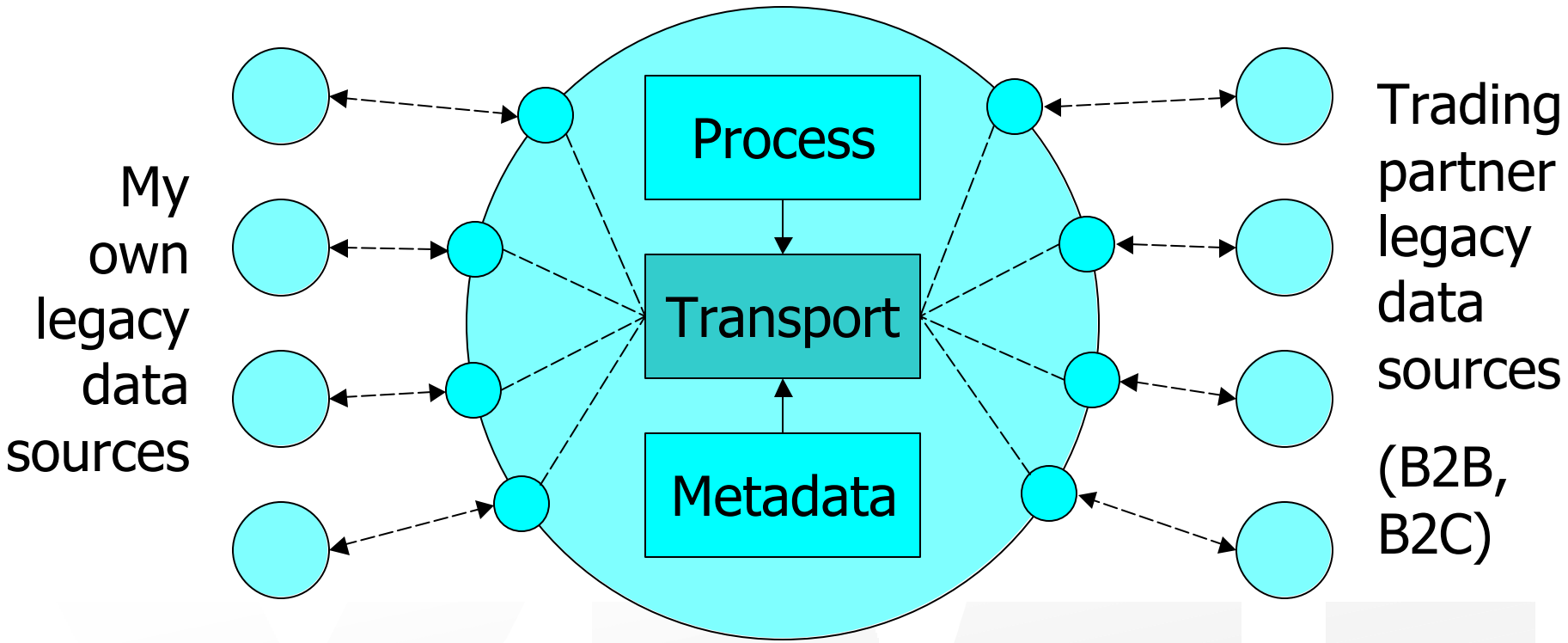


Step 3: Linked Hub Solution



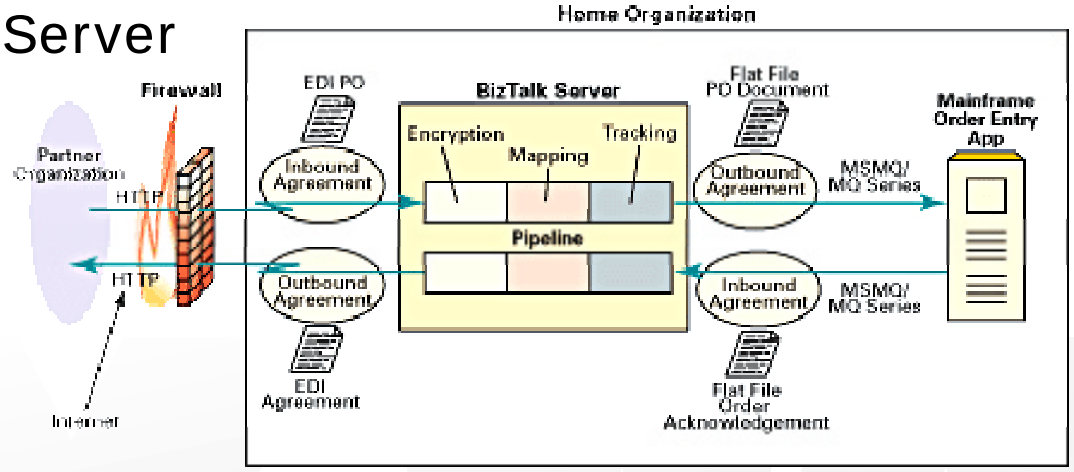
XML and legacy data sources

- Internal functions link all partners



What is Microsoft BizTalk Initiative?

- BizTalk.org
  - Website to learn about XML and XDR
- BizTalk Framework
  - Schema specifying the structure of BizTalk messages
- BizTalk Jumpstart Kit
  - Classes for VB programmers to enable XML into applications
- BizTalk Server



Enough of business talk,

- let’s get down to business

Will XML replace HTML?

- No
- HTML is still used to render the data
- Redefined as a XML vocabulary (XHTML)
  HTML 5.0 will be XHTML 1.0
- Existing HTML can be transformed into XHTML by making it well-formed (nesting, closing-tags)

HTML Tidy: http://www.w3.org/People/Raggett/tidy/

How can you view XML?

- MS Internet Explorer 5
  - Supports an invalid hybrid solution (data islands)
  - Includes an implementation of the older XSLT working draft with limited functionality

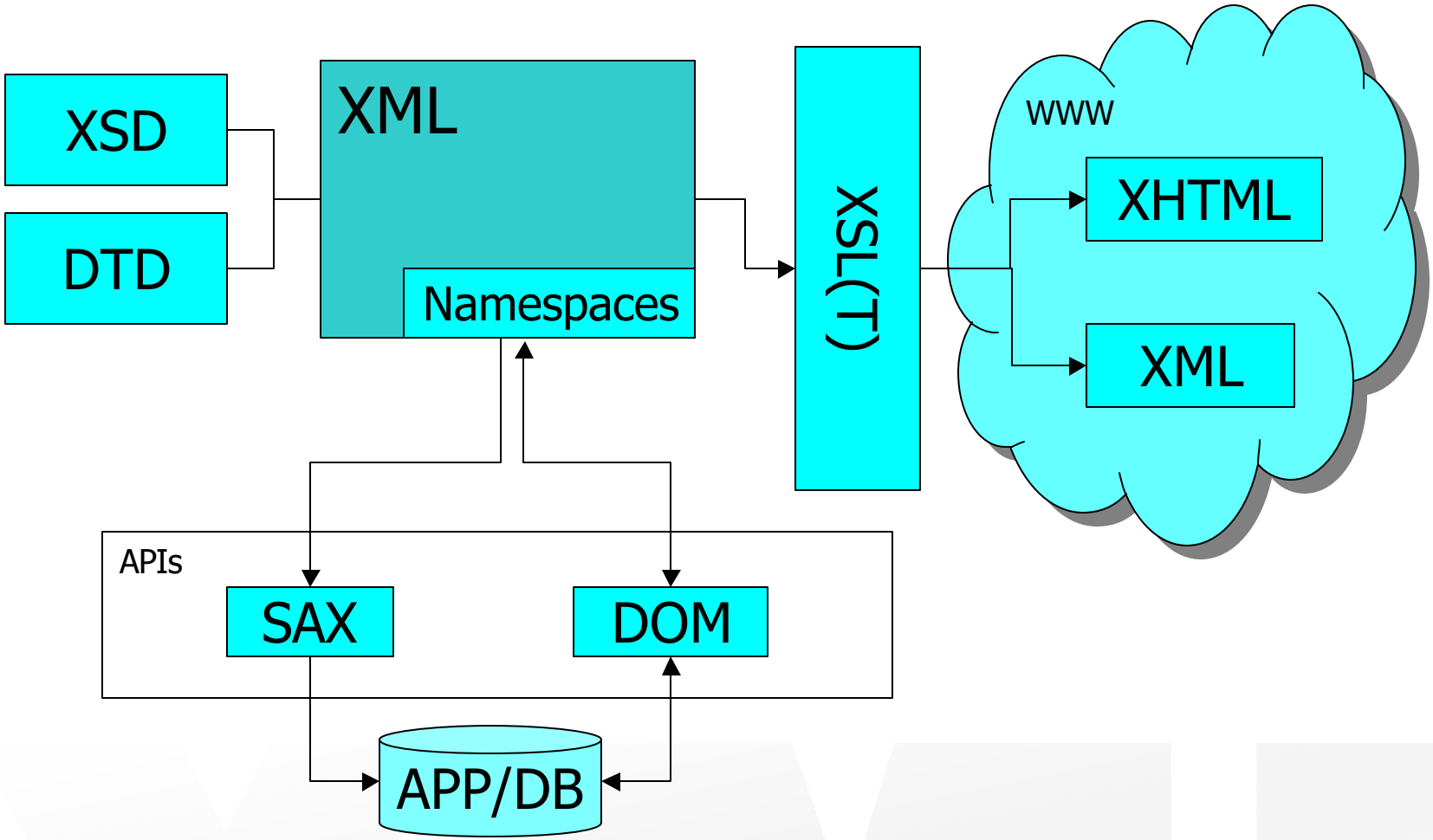
How can you view XML?

- Netscape Navigator 6
  - Netscape Navigator 6 and open-source Mozilla support XML
  - Support seems to be more robust and less slick than MS Internet Explorer

How can you view XML?

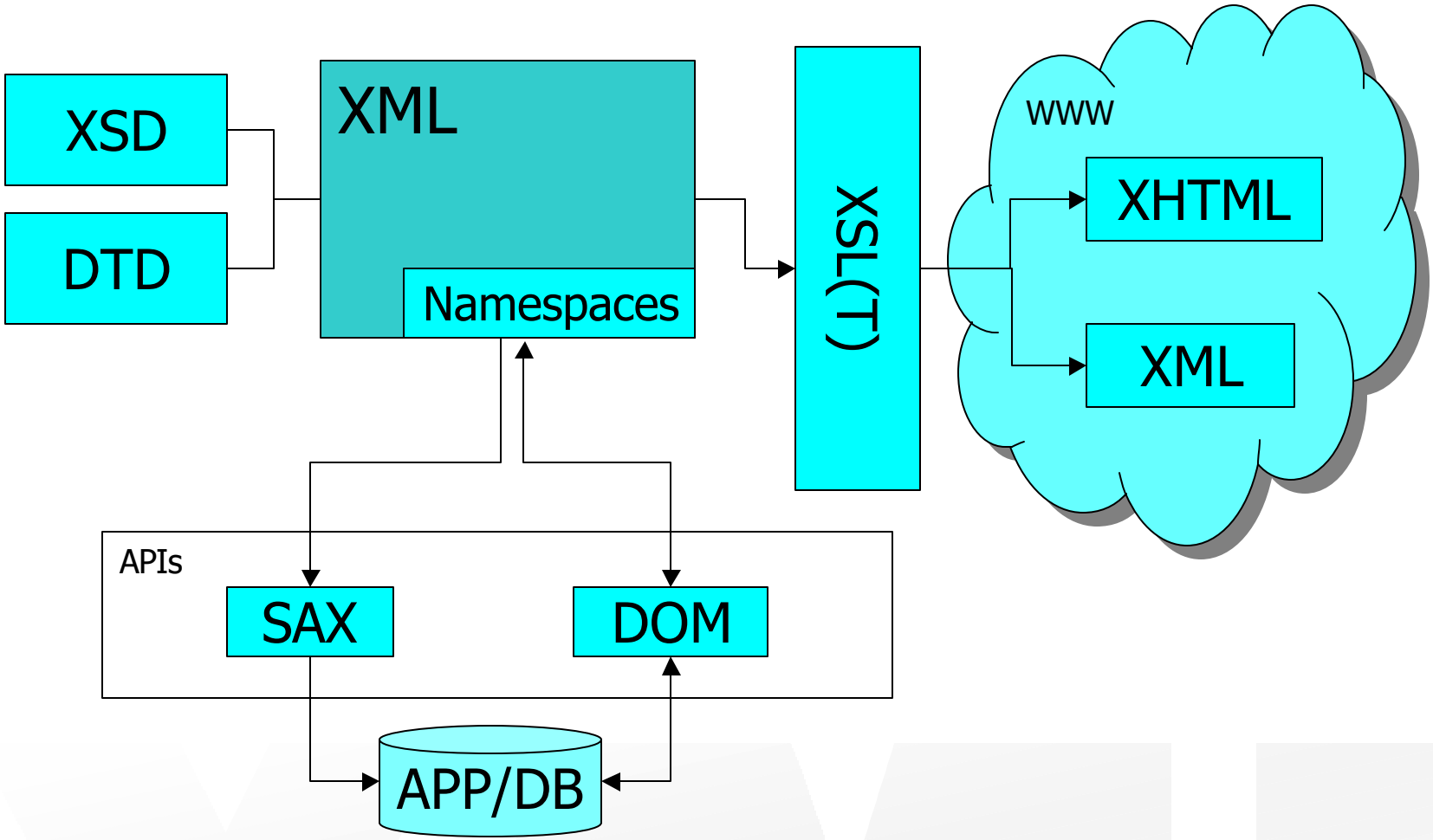
- Opera 4.0
  - Supports XML, CSS and XSL
  - Most complete implementation of XML thus far
  - Very small & good performance
  - Not free!

What will we cover?

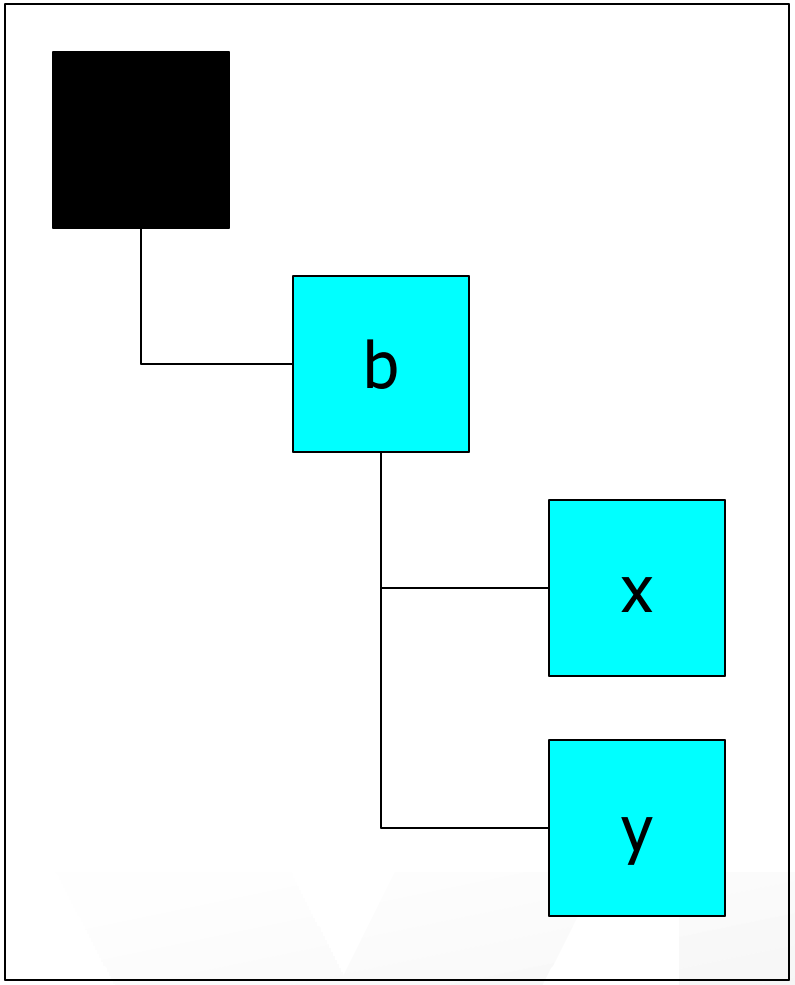


All Standards on this slide are reasonably stable, except XSL and XSD!

Modeling



Modeling



Enforce Structure

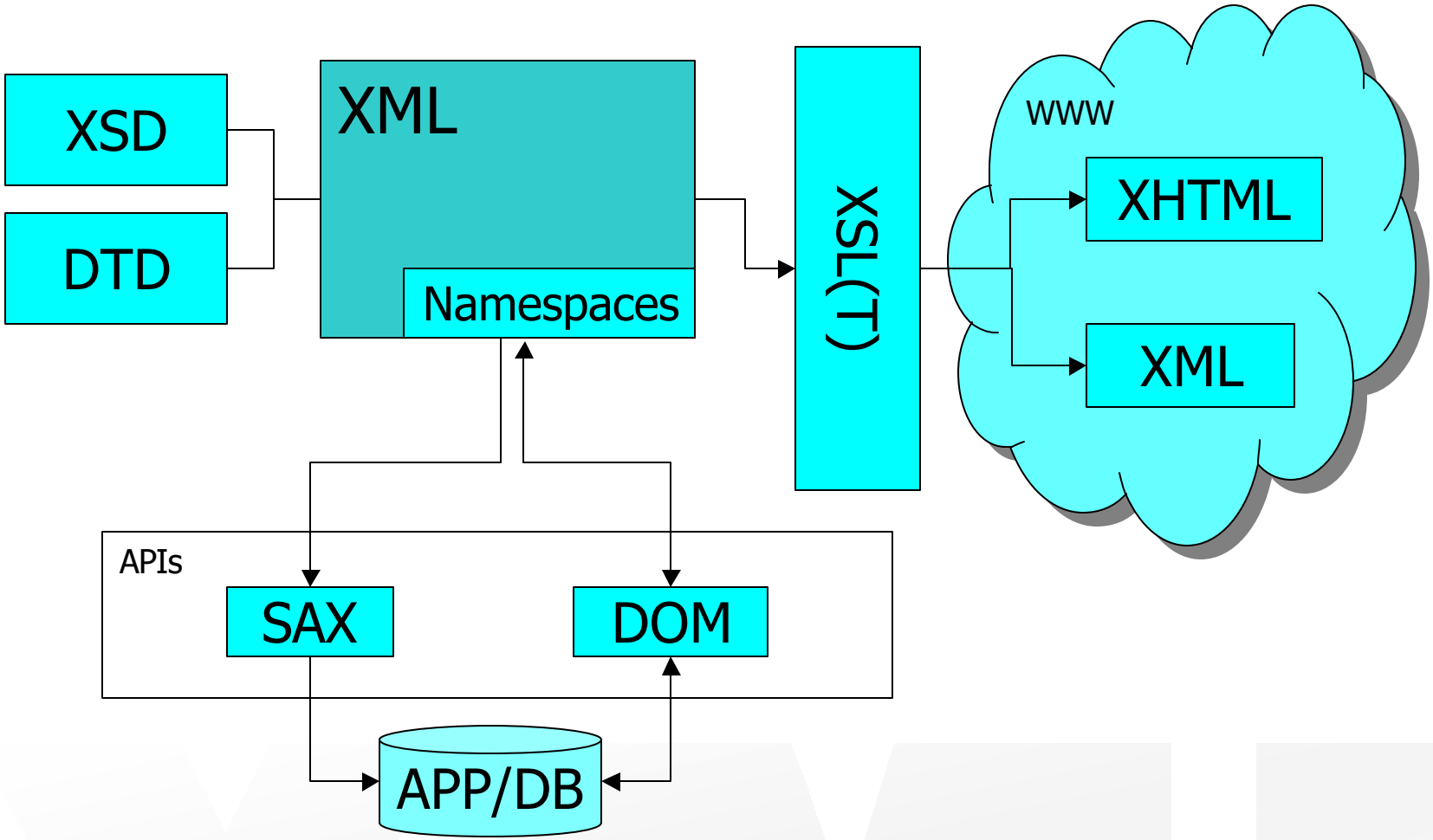
```
<a>
    <b>
        <x>…</x>
        <y>…</y>
    </b>
</a>
```

DTD

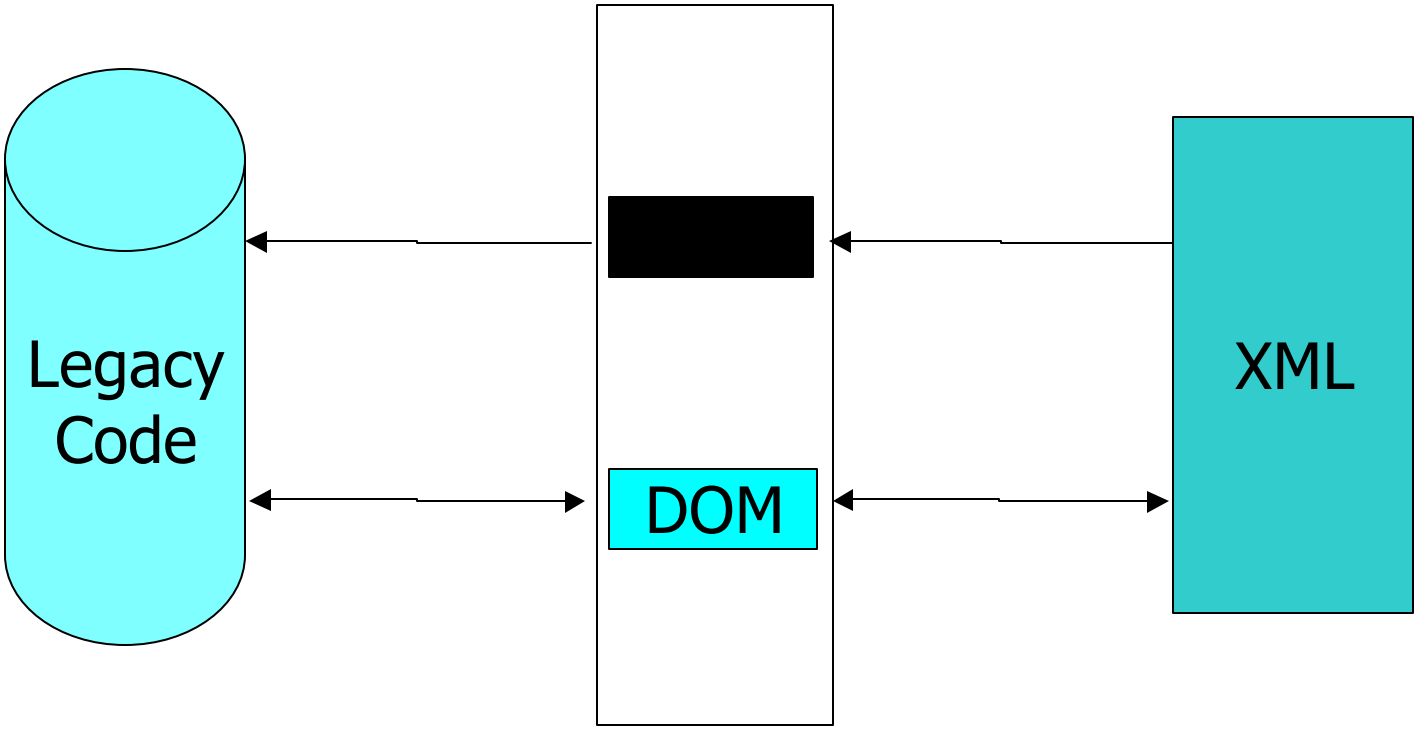
XSD

XML

Manipulation

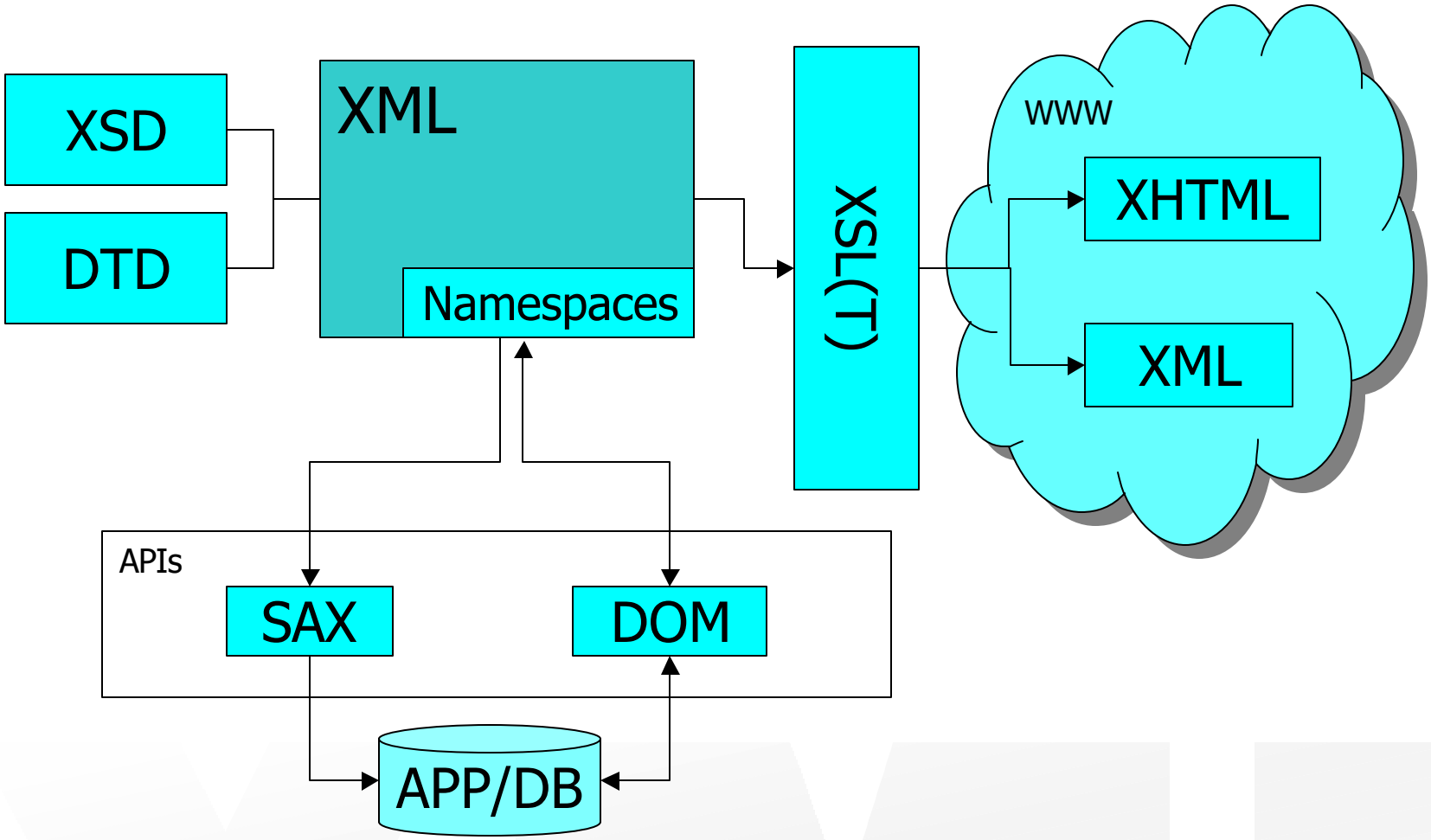


Manipulation

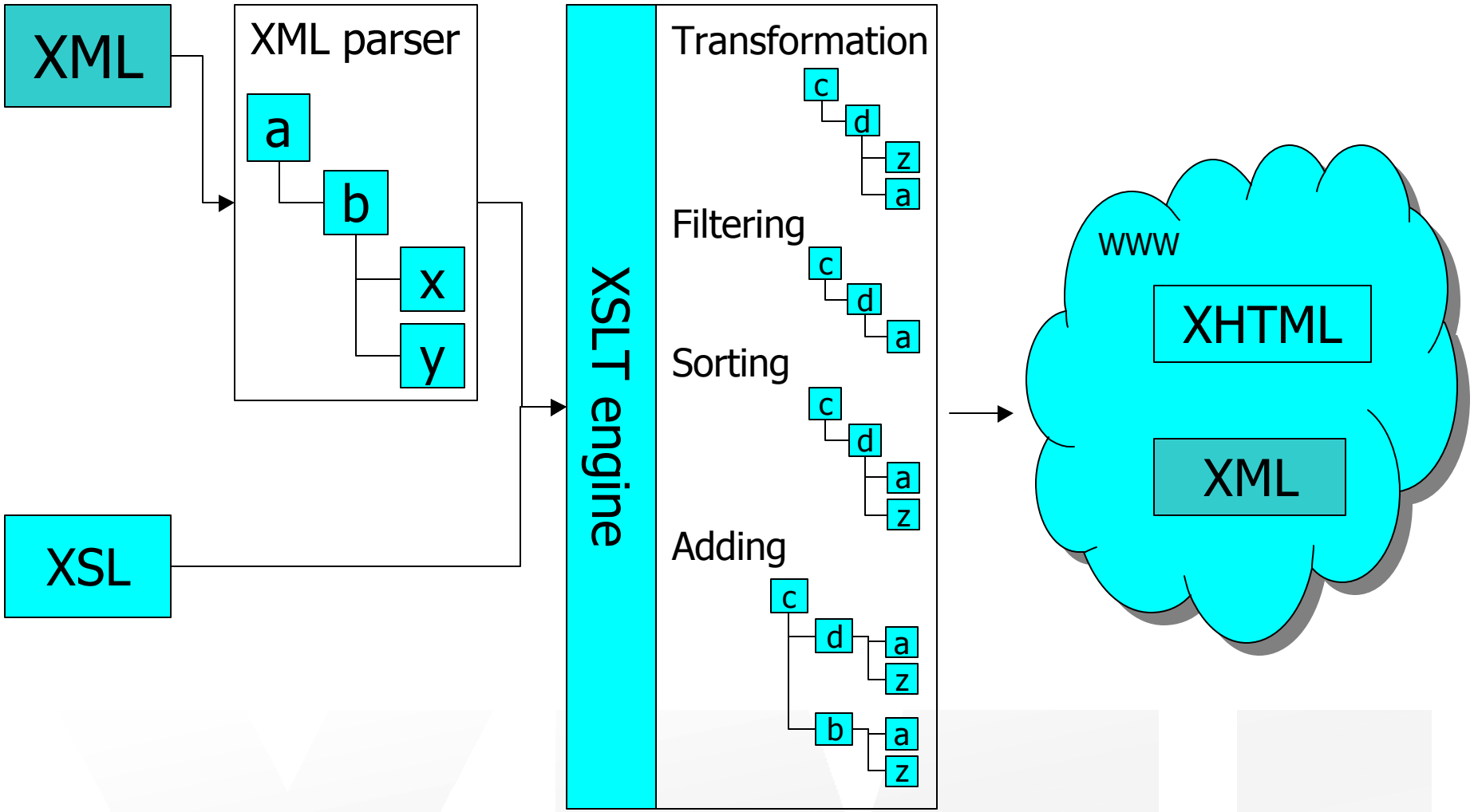


SAX DOM Legacy Code

Display

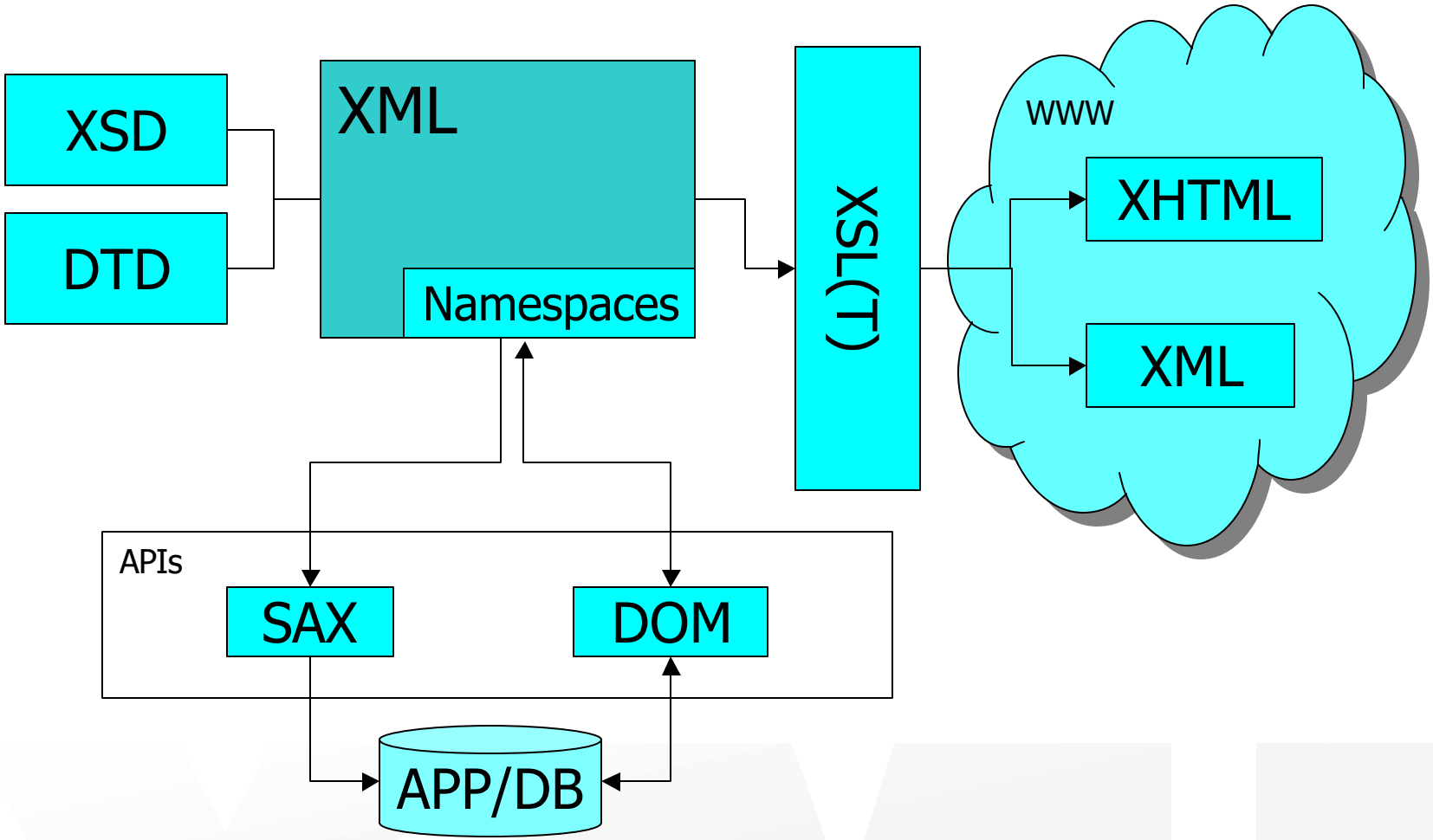


Display



XML XSL(T) → XML XHTML

XML - The Core



Editors: Simple text editor or XMLSpy

XML - Components

```
<?xml version="1.0" encoding="UTF-16"?>
<emailMessage>
    <To>Samuel</To>
    <From>Adam</From>
    <Subject>Party at 5:00</Subject>
    <Message lang="EN">
        Wanna go?
    </Message>
    <Attachment/>
    <!-- No comment-->
    <?MyPrinter R=128; G=0; B=128?>
    <![CDATA[This is not parsed!]]>
</emailMessage>
```

XML - Prolog

```
<?xml version="1.0" encoding="UTF-16"?>
<emailMessage>
    <To>Samuel</To>
    <From>Adam</From>
    <Subject>Party at 5:00</Subject>
    <Message lang="EN">
        Wanna go?
    </Message>
    <Attachment/>
    <!-- No comment-->
    <?MyPrinter R=128; G=0; B=128?>
    <![CDATA[This is not parsed!]]>
</emailMessage>
```

Prolog.

Provides version against which to parse.

Optional character set.

XML - Document Element

```
<?xml version="1.0" encoding="UTF-16"?>
<emailMessage>
    <To>Samuel</To>
    <From>Adam</From>
    <Subject>Party at 5:00</Subject>
    <Message lang="EN">
        Wanna go?
    </Message>
    <Attachment/>
    <!-- No comment-->
    <?MyPrinter R=128; G=0; B=128?>
    <![CDATA[This is not parsed!]]>
</emailMessage>
```

Document Element.

Container for the whole document.

XML - Empty Element

```
<?xml version="1.0" encoding="UTF-16"?>
<emailMessage>
    <To>Samuel</To>
    <From>Adam</From>
    <Subject>Party at 5:00</Subject>
    <Message lang="EN">
        Wanna go?
    </Message>
    <Attachment/>
    <!-- No comment-->
    <?MyPrinter R=128; G=0; B=128?>
    <![CDATA[This is not parsed!]]>
</emailMessage>
```

Empty Element.

Allowed to have attributes.

XML - Other Elements

```
<?xml version="1.0" encoding="UTF-16"?>
<emailMessage>
    <To>Samuel</To>
    <From>Adam</From>
    <Subject>Party at 5:00</Subject>
    <Message lang="EN">
        Wanna go?
    </MeSsAgE>
    <Attachment/>
    <!-- No comment-->
    <?MyPrinter R=128; G=0; B=128?>
    <![CDATA[This is not parsed!]]>
</emailMessage>
```

Other Elements.

Are case-sensitive !

Must be nested correctly.

Have attributes (quoted !)

Follow naming conventions.

Start with: Letter / + / , / :

Followed by: Letter / Digit / _ / : / , / .

XML - Unparsed and parsed data

```
<?xml version="1.0" encoding="UTF-16"?>
<emailMessage>
    <To>Samuel</To>
    <From>Adam</From>
    <Subject>Party at 5:00</Subject>
    <Message lang="EN">
        Wanna go?
    </Message>
    <Attachment/>
    <!-- No comment-->
    <?MyPrinter R=128; G=0; B=128?>
    <![CDATA[This is not parsed!]]>
</emailMessage>
```

Unparsed data.

Ignored while the document is run through the parser.

Common technique, when passing data from a legacy system.

Reserved letters: > / < / & / ` / "

Parsed data.

All other code is parsed data.

XML - Processing Instruction

```
<?xml version="1.0" encoding="UTF-16"?>
<emailMessage>
    <To>Samuel</To>
    <From>Adam</From>
    <Subject>Party at 5:00</Subject>
    <Message lang="EN">
        Wanna go?
    </Message>
    <Attachment/>
    <!-- No comment-->
    <?MyPrinter R=128; G=0; B=128?>
    <![CDATA[This is not parsed!]]>
</emailMessage>
```

Processing Instruction.

Used to pass instruction to an application.

i.e.

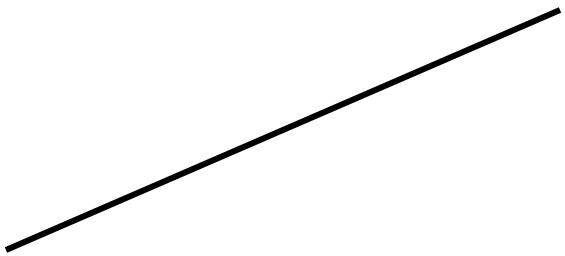
Formatting information

Presentation information

Scripting information

XML - Comments

```
<?xml version="1.0" encoding="UTF-16"?>
<emailMessage>
    <To>Samuel</To>
    <From>Adam</From>
    <Subject>Party at 5:00</Subject>
    <Message lang="EN">
        Wanna go?
    </Message>
    <Attachment/>
    <!-- No comment-->
    <?MyPrinter R=128; G=0; B=128?>
    <![CDATA[This is not parsed!]]>
</emailMessage>
```



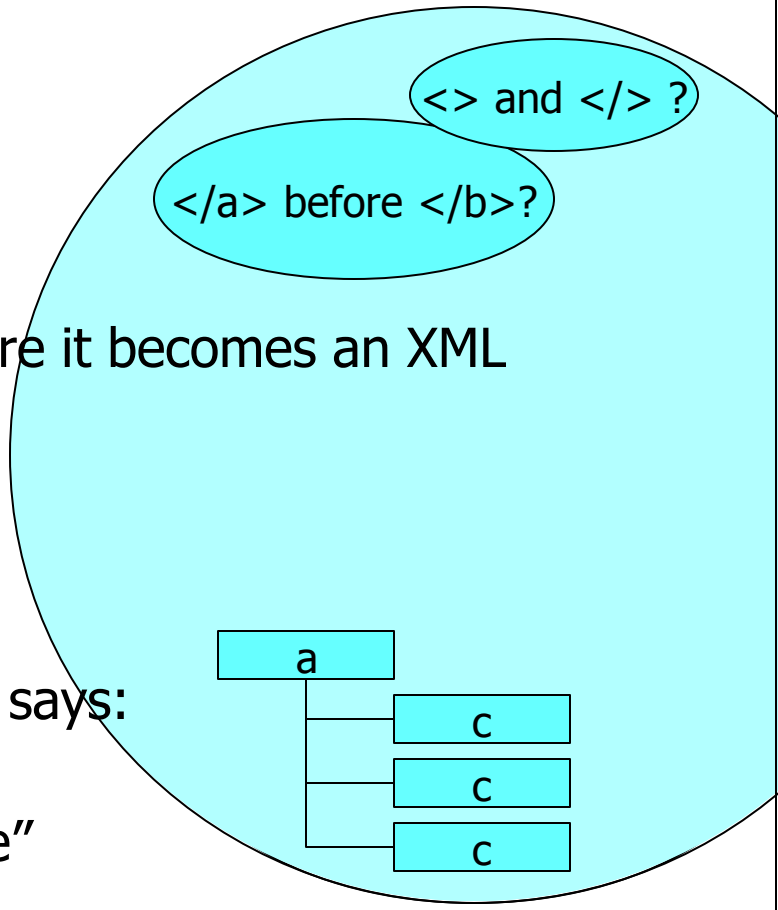
Comments.

May or maybe not parsed into the processing application.

May appear anywhere in XML document

XML - Well-Formed and Valid

- Well-Formed:
  - Syntax is correct
    i.e. no nested tags, tags match case
  - A document has to be well formed before it becomes an XML document and can be used any further.
- Valid:
  - Semantic is correct
    i.e. when compared against a DTD that says:
    "Name must be a child of address !",
    "Message can have a language attribute"
  - May be valid against one DTD and not valid against another one!



XML - Well-Formed and Valid

- How do you do it?
- Next session:
  Information Modeling with XSD and DTD