

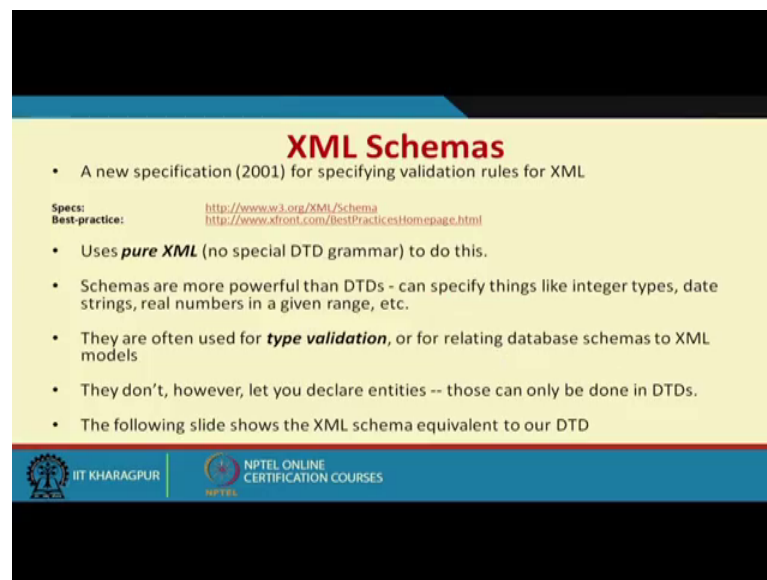
**Cloud Computing**  
**Prof. Soumya Kanti Ghosh**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 09**  
**XML Basics (Contd.)**

Hello. So, we will continue our discussion on cloud computing. So, if you remember the last lecture you are talking about XML basics, we are talking about this how XML discussing how XML allows us to interoperate between, between different interacting or cooperating system in a or in a distributed system. And how it can it can become a one of the major backbone for realizing cloud computing specially in such type of cloud software as a service type of cloud, right.

So, we were discussing about DTD and XML schema XSD. So, today we will continue that that discussion and on this basics of XML, alright.

(Refer Slide Time: 01:11)



**XML Schemas**

- A new specification (2001) for specifying validation rules for XML

Specs: <http://www.w3.org/XML/Schema>  
Best-practice: <http://www.xmlfront.com/BestPracticesHomepage.html>

- Uses **pure XML** (no special DTD grammar) to do this.
- Schemas are more powerful than DTDs - can specify things like integer types, date strings, real numbers in a given range, etc.
- They are often used for **type validation**, or for relating database schemas to XML models
- They don't, however, let you declare entities -- those can only be done in DTDs.
- The following slide shows the XML schema equivalent to our DTD

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, as if you remember that we are discussing about XML schema which is primarily defines the structure of the XML. So, XML of a used data like say I have a XML of a student data of a particular batch of student of IIT, Kharagpur, it will be a huge volume of data now whenever I want this data say our placement section or our account section

for the bank or other things. So, they want to they want to say a some of these data with the external agencies, right.

So, one way of looking at it that say are a part of the schema look at the schema and say are the part of the things with the with the external agencies, or I want to on the other way in your organization takes the data from the other organization. So, the first of all it there is a need of that schema to be to be agreed upon that what should be there. So, the XML schema place a vital role in exchange of information helping in interoperability between different repositories.

So, a new specification schema is a new specification which came up in 2001 a to specify validation rule of XML, right. So, what we have seen well formed any XML parser per 6 per check whether that it is syntactically correct that is well formed. And secondly, the schema it checks with the schema if available to say that whether it is valid with respect to the schema. So, what we say it is a valid XML document, alright. So, so there are w 3 some spec and best practice as you can see. So, it use pure XML to do this XML schema that are retained in XML, schemas are more powerful and DTDs can specify things like integer type date real thing etcetera and other type of parameters.

They are often use for type validation alright, or for relating data base schemas to XML models all right. So, it is I can if I know the schema than I create the database and then pump this XML data to this database or other way around, alright. They do not however, let you declare entities; those can be done only in DTD.

(Refer Slide Time: 03:46)

### XML Schema version of our DTD (Portion)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified">
  <xs:element name="accountID" type="xs:string"/>
  <xs:element name="acknowledgeReceipt" type="xs:string"/>
  <xs:complexType name="amountType">
    <xs:simpleContent>
      <xs:restriction base="xs:string">
        <xs:attribute name="currency" use="required">
          <xs:simpleType>
            <xs:restriction base="xs:NMTOKEN">
              <xs:enumeration value="USD"/>
              . . . (some stuff omitted) . . .
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
      </xs:restriction>
    </xs:simpleContent>
  </xs:complexType>
  <xs:complexType name="fromType">
    <xs:sequence>
      <xs:element name="amount" type="amountType"/>
      <xs:element ref="transitID" minOccurs="0"/>
      <xs:element ref="accountID"/>
      <xs:element ref="acknowledgeReceipt"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

simple.xsd

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, it cannot define this in DTDs. The followings will see that how schema is equivalent to our DTDs. Like the last DTD we have if have if you remember in the XML basics that first class. So, that same thing is now represented as a schema.


So, what we see it is more of XML type document and in doing. So, I can handle this schema the same way I handle a XML data, alright.

(Refer Slide Time: 04:09)

### XML Namespaces

- Mechanism for identifying different "spaces" for XML names
  - That is, *element* or *attribute* names
- This is a way of identifying different *language dialects*, consisting of names that have specific semantic (and processing) meanings.
- Thus <key/> in one language (might mean a security key) can be distinguished from <key/> in another language (a database key)
- Mechanism uses a special **xmlns** attribute to define the namespace. The namespace is given as a **URL string**
  - But the URL does not reference anything in particular (there may be nothing there)

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES



So, that that is a beauty of the thing; another aspect that the important aspect of XML is XML name space, right; it is a very important aspect as XML allows user defined data. So, you need to be careful on that which type of definition is there and which namespace we are looking at. If you remember last class we are discussing that if I define a entity call table or element call table then I need to specify whether this is a table which type of table. Like I can have a table as we have we have discussed that furniture or the table can be from a say table of a; what processor or spread said type of things.

So, in order to distinguish I have to say table of furniture type then it has some definition, table of say what processing has some other definition alright. So that means, where I will define this I will define in the name space. So, the name space tell me name space basically specifies that which what element is there what sort of dealing etcetera etcetera there. So, it is a mechanism for identifying different spaces for XML names alright that is element or attributes names. This is a way of identifying different language dialects consisting of names that have specific semantics and processing meanings all right. The key is one language mean a specific security key the as we are talking about table the distinguished from, the key in another language with maybe a database key A, right.

So, key maybe some sort of a security key and in some other thing the key what we are looking for is more of a database related keys. So, this 2 keys are one from database schema a database name space and another for this security name space need to be defined somewhere. So, it is a may use uses a special XMLNS XML name stands for XML name space attribute to define this name space right. The namespace is given as a URL string alright, but the URL does not reference does not reference anything in particular they maybe nothing is there. So, it may not refer to something.

(Refer Slide Time: 06:39)


**Mixing language dialects together**

**Namespaces let you do this relatively easily:**


```
<?xml version="1.0" encoding="utf-8" ?>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:mt="http://www.w3.org/1998/mathml" >
  <head>
    <title> Title of XHTML Document </title>
  </head><body>
    <div class="myDiv">
      <h1> Heading of Page </h1>
      <mt:mathml>
        <mt:title> ... MathML markup ...
      </mt:mathml>
      <p> more html stuff goes here </p>
    </div>
  </body>
</html>
```

Default 'space' is *xhtml*


*mt*: prefix indicates 'space' mathml (a different language)



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

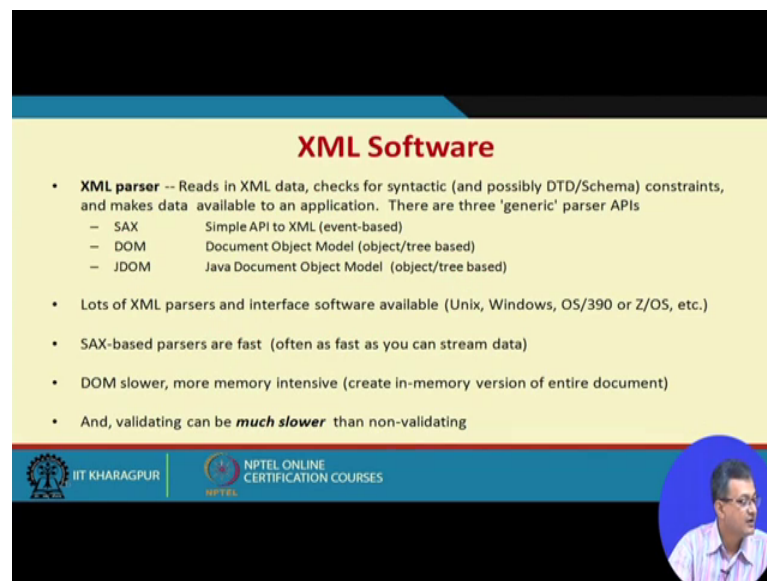


So, namespace so, mixing language dialogues together. So, namespace let you do this relatively easier like I can have something from the something from that like, I want both the keys all right. From the database keys then I use the security key for unlocking the database, and then user database key for accessing the both the keys I am using somewhere someone right. And both are in the XML So, this XML namespace may allow me to do that like I can say that db dot key mean something or somewhere something dot some they that sec dot security dot something it means a some other key.

So, I basically specify the type of key is there. So that means, here if you look at these are the name space which are a default space right. Which define by the w 3 c whereas, this name space empty is a math related name space right. That is also a predefined, but I say that the empty is a math related name space. Now whenever I do anything with this particular math related then I say empty double colon mathml and title and so on and so forth; that means, I say that I am referring these math name space.

So, empty prefix indicates the space mathml a different language mathml maybe a totally different language and I basically able to use those elements of that particular name space in my document.

(Refer Slide Time: 08:18)



**XML Software**

- **XML parser** -- Reads in XML data, checks for syntactic (and possibly DTD/Schema) constraints, and makes data available to an application. There are three 'generic' parser APIs
  - SAX Simple API to XML (event-based)
  - DOM Document Object Model (object/tree based)
  - JDOM Java Document Object Model (object/tree based)
- Lots of XML parsers and interface software available (Unix, Windows, OS/390 or Z/OS, etc.)
- SAX-based parsers are fast (often as fast as you can stream data)
- DOM slower, more memory intensive (create in-memory version of entire document)
- And, validating can be *much slower* than non-validating

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

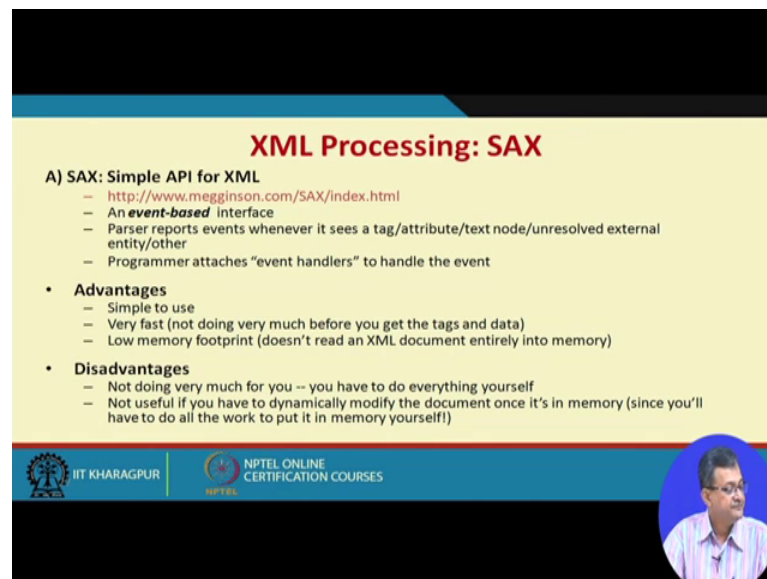
So, this so, name space as we sees plays a important role and you should be careful, and the name space definition are should be there. So, in other words so, whenever we do like we have seen that mathml that what we do is more of looking at that particular name space and how to use it and type of things.

Another important aspects already we have discussed some of these is XML software, right. So, as XML document is one is representation textual mode whereas, I have a I need to process it and type of thing one a first of all I validate well form or not syntactic recurring this is one part. Secondly, need to process it for different application etcetera. So, this XML software plays a important role.

So, XML parser reads in XML data checks for syntactic possibly with DTD and schema constraint, and makes all data available to an application. There are 3 generic XML APIs or parser APIs like one is a SAX parser simple API to XML event based it is a very popular widely used parser and available mostly in all platform. Another is DOM parser, document object model or object tree based parser another is JDOM parser java DOM parser right. So, lots of XML parsers and interface software available in different language different operating systems, SAX based parsers are fast often as fast as you can stream data and these are very low bellowed parser and this pretty fast though the functionalities less.

Whereas DOM is slower more memory intensive create in memory version of the entire document and type of things all right. And validating can be much slower than non validating parser, because validating parser need to validate whether it is conformed to the schema before doing any processes.

(Refer Slide Time: 10:33)



**XML Processing: SAX**

A) SAX: Simple API for XML

- <http://www.megginson.com/SAX/index.html>
- An **event-based** interface
- Parser reports events whenever it sees a tag/attribute/text node/unresolved external entity/other
- Programmer attaches "event handlers" to handle the event


• **Advantages**

- Simple to use
- Very fast (not doing very much before you get the tags and data)
- Low memory footprint (doesn't read an XML document entirely into memory)

• **Disadvantages**

- Not doing very much for you -- you have to do everything yourself
- Not useful if you have to dynamically modify the document once it's in memory (since you'll have to do all the work to put it in memory yourself!)

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES



So, these are some of the quick look at the different parsers like if the SAX parser is the available there, it say event based interface parser reports event whenever it sees a tag attribute text node unresolved external entities and other.

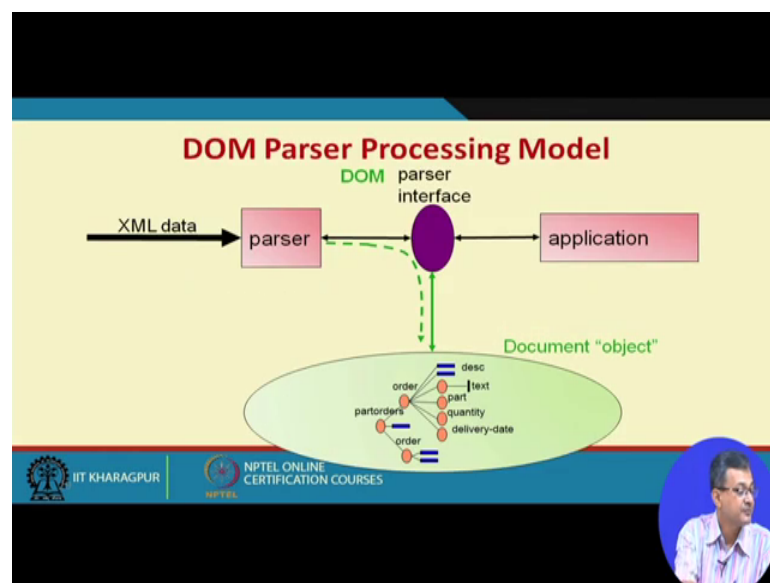
Programmers attach event handlers to handle the events, right. Advantages it is simple to use, very fast not doing much before you get the tax and data, right. So, it is a it is pretty fast low memory footprints. So, it is low payload disadvantages not doing much of the processing. So, you need to do some of the processing at your end not useful if you have dynamically modify document world series in memory. So, it is not useful if you dynamically modify the documents it is memory since you will have to do the all the work and put memory of yourself anyway. So, what it is there it is useful when you have a vanilla type xm processing requirement which process and goes on with low memory footprint and ready fast and overall pretty simple to use.

Whereas the document object model or the DOM parser it is a object based interface parser generates in memory tree, in memory tree or the in memory XML tree

corresponding to the XML document DOM interface identifies method for accessing and modifying the tree, right. So, that is the advantage very useful for dynamic modification access to the tree. So, if there is a dynamic modification access to the tree dynamically changing then it is extremely useful, useful for query that is looking for data and depends on the tree structure. So, querying because if it is a tree structure which plays a role it is useful. It is the same interface for many programming language c plus java that is as such the interface wise interpret interoperate with different languages, disadvantage can be slow, right.

Needs to produce the tree and may need lots of memory DOM programming interface is little bit not that state forward. So, it needs to be little complex and needs more what we say professional handling of the data here handling of this programming.

(Refer Slide Time: 13:08)



So, in case of a DOM parser the parser this parser interface it goes to this structure of the tree which is in (Refer Time: 13:18) maintaining the thing and based on that (Refer Time: 13:21) application. The JDOM or the java DOM it is java based object oriented.



(Refer Slide Time: 13:24)

**C) JDOM: Java Document Object Model**

- <http://www.jdom.org>
- A Java-specific **object-oriented** interface
- Parser generates an in-memory tree corresponding to the document
- JDOM interface has methods for accessing and modifying the tree

• **Advantages**

- Very useful for dynamic modification of the tree
- Useful for querying (i.e. looking for data) that depends on the tree structure
- Much nicer Object Oriented programming interface than DOM

• **Disadvantages**

- Can be slow (make that tree...), and can take up lots of memory
- New, and not entirely cooked (but close)
- Only works with Java, and not (yet) part of Core Java standard

**XML Processing: JDOM**

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, it is similar to DOM with the; it is from java. Parser generates in memory tree corresponding the document. JDOM interface has methods for accessing and modifying the tree, right. So, where accessing and modifying the tree, advantages very useful dynamic modification of the tree useful for querying that depends on the tree structured. So, if your query is dependent on the tree structure it is pretty first and useful. Much nicer object oriented programming interface than DOM. So, it is much means better programming interface is there.

These advantage can be slow a new and not entirely cooked not. So, new these days, but there are you need some expertise to work on it and it works on java that is that maybe disadvantage or it may be thought of what we say that they you need to know java to work on it may not be a big issue.

(Refer Slide Time: 14:30)

**C) dom4j: XML framework for Java** **XML Processing: dom4j**

- <http://www.dom4j.org>
- Java framework for reading, writing, navigating and editing XML.
- Provides access to SAX, DOM, JDOM interfaces, and other XML utilities (XSLT, JAXP, ...)
- Can do “mixed” SAX/DOM parsing -- use SAX to one point in a document, then turn rest into a DOM tree.

• **Advantages**

- Lots of goodies, all rolled into one easy-to-use Java package
- Can do “mixed” SAX/DOM parsing -- use SAX to one point in a document, then turn rest into a DOM tree
- Apache open source license means free use (and IBM likes it!)

• **Disadvantages**

- Java only; may be concerns over open source nature (but IBM uses it, so it can't be that bad!)

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

There are few more like a dom4j which is java framework for reading writing navigation and editing XML right. Provides access to SAX, DOM, JDOM interface and other XML entities like XSLT and other type of interface, can do mixed SAX, DOM parsing all right. And it has some of the some advantages like good of all lots of goodies and all rolled into one things in java package can make SAX and DOM parser apache open source license means free use And so and so forth.

(Refer Slide Time: 15:17)

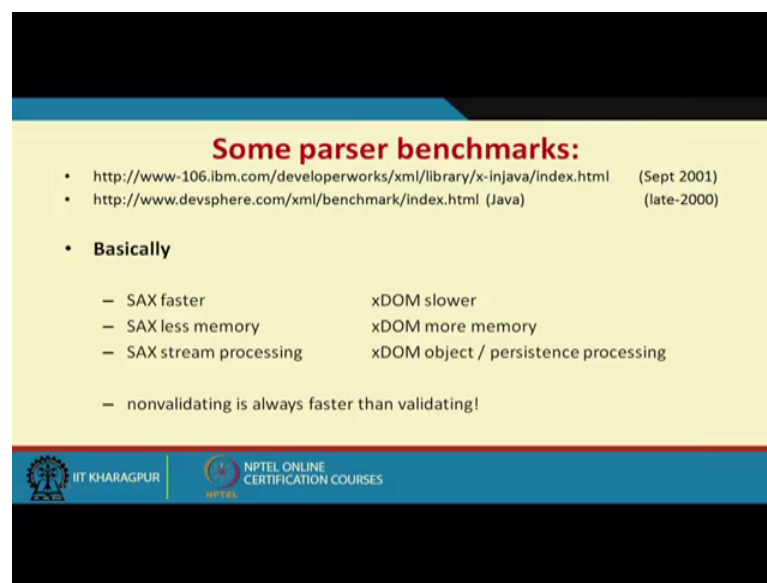
**Some XML Parsers (OS/390's)**

- Xerces (C++; Apache Open Source)  
<http://xml.apache.org/xerces-c/index.html>
- XML toolkit (Java and C++; Commercial license)  
<http://www-1.ibm.com/servers/eserver/zseries/software/xml/>  
I believe the Java version uses XML4j, IBM's Java Parser. The latest version is always found at:  
<http://www.alphaworks.ibm.com>
- XML for C++ (IBM; based on Xerces; Commercial license)  
<http://www.alphaworks.ibm.com/tech/xml4c>
- XMLBooster (parsers for COBOL, C++ ...; Commercial license; don't know much about it; OS/390? [dunno])  
<http://www.xmlbooster.com/>  
Has free trial download; can see if it is any good :-)
- XML4Cobol (don't know much about it, any COBOL85 is fine)  
<http://www.xml4cobol.com>
- [www.xmlsoftware.com/parsers/](http://www.xmlsoftware.com/parsers/) -- Good generic list of parsers

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

There are disadvantages it is again java only and maybe concerns for open source nature of things like that, some of the things, but nevertheless it is a (Refer Time: 15:18) there are some other XML parser like excess and XML toolkit XML for c plus and etcetera, etcetera. So, rather there are a good number of XML parser to not only well formed and validating also try to do more complex operations in our in basically in when we are doing interoperating in some distributed system like cloud.

(Refer Slide Time: 15:50)



**Some parser benchmarks:**

- <http://www-106.ibm.com/developerworks/xml/library/x-injava/index.html> (Sept 2001)
- <http://www.devsphere.com/xml/benchmark/index.html> (Java) (late-2000)

• **Basically**

– SAX faster	xDOM slower
– SAX less memory	xDOM more memory
– SAX stream processing	xDOM object / persistence processing
– nonvalidating is always faster than validating!	

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, there are some of the benchmark like how the parser will be there whether regarding speed memory and stream processing there are few parser benchmarks.

(Refer Slide Time: 16:08)

**XML Processing: XSLT**

**D) XSLT eXtensible Stylesheet Language -- Transformations**

- <http://www.w3.org/TR/xslt>
- An XML language for processing XML
- Does tree transformations -- takes XML and an XSLT style sheet as input, and produces a new XML document with a different structure

• **Advantages**

- Very useful for tree transformations -- much easier than DOM or SAX for this purpose
- Can be used to query a document (XSLT pulls out the part you want)

• **Disadvantages**

- Can be slow for large documents or stylesheets
- Can be difficult to debug stylesheets (poor error detection; much better if you use schemas)

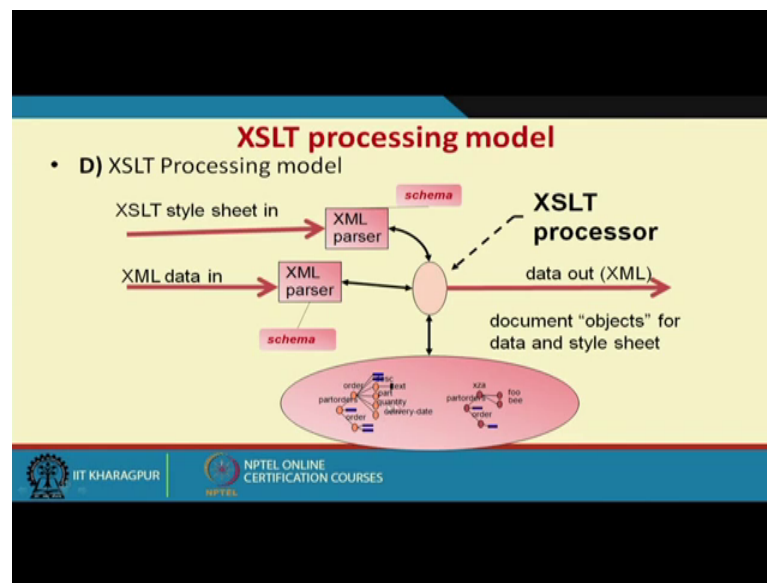
IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

XML processing another important aspect is XSLT right. So, extensible style sheet language, right. So, how XML to be represented is given by this style sheet, right. So, in some other sense I can say that XML along with the styling allows me to represent it or displayed right. So, sometimes XML plus XSLT is something equivalent to HTML type of things like not that all power wise, but it is characteristics wise. So, exestensee XSLT extensible style sheet language is primarily used for transformation type of things right. Or it is more of a transformation type of things. XML language for processing XML data does tree transformation takes XML and XSLT style sheet as input, and produce a new XML document with a different structure and type of things.

So, as it is XSLT. So, I can basically do some sort of a filtering operations on the XSLT right. So, it takes the XML data, does some sort of process it XSLT and generate a another XML data on the other hand and produces a new XML document with different structure and so on and so forth. So, advantages of course, very large very useful for tree transformations; so, if I want to transform the tree or say I have a tree XML tree, and for my application I require want to use a sub tree of it all right. So, how can I do it? So, I can have this filtering using this XSLT and then filter these things to the other part.

So, disadvantage can be slow for large document or style sheets right. It can be disadvantages and can we slow can be difficult to debug style sheet poor error detection, etcetera.

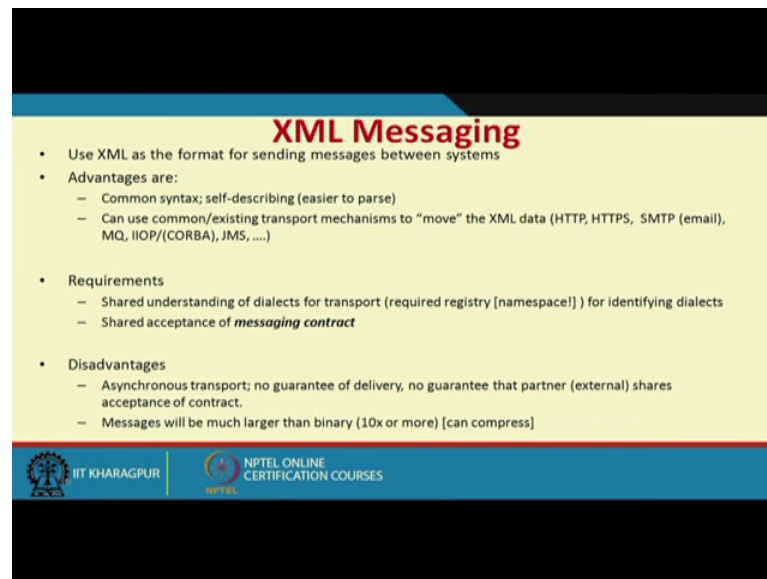
(Refer Slide Time: 18:16)



So, that another is that not so versatile in error detection. So now, if you look at our big picture So, XML data is coming right. So, we have now a style sheet with a it requires again another XML parser. And then it goes to this processor processing unit. And we have this different tree.

So, I have a tree here and I can have a part of the tree out here all right. So, taking these as input the data out is a XML which is which is based on the XSLT has been filtered for the; and generated new XML, XML messaging use XML as the format for sending messages between the systems. Advantages common syntax self describing easier to parse can use common existing transport mechanism to move the XML data like https etcetera.

(Refer Slide Time: 18:59)



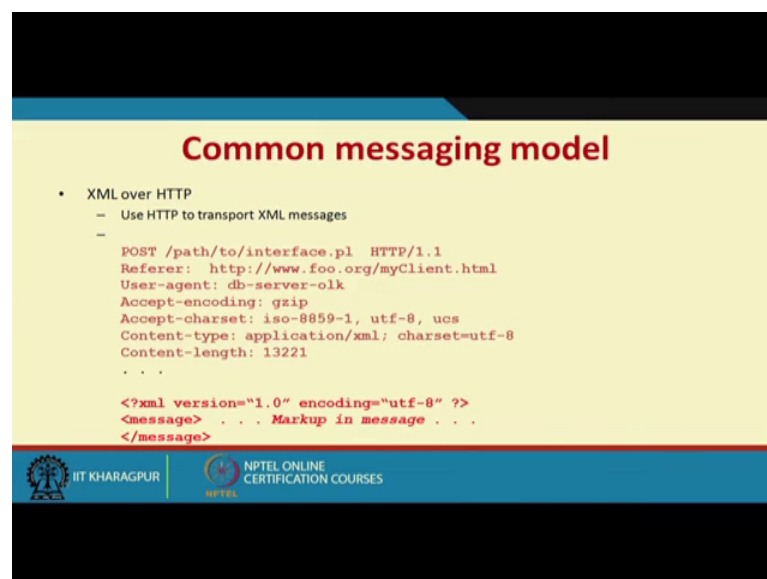
### XML Messaging

- Use XML as the format for sending messages between systems
- Advantages are:
  - Common syntax; self-describing (easier to parse)
  - Can use common/existing transport mechanisms to “move” the XML data (HTTP, HTTPS, SMTP (email), MQ, IIOP/(CORBA), JMS, ....)
- Requirements
  - Shared understanding of dialects for transport (required registry [namespace!]) for identifying dialects
  - Shared acceptance of *messaging contract*
- Disadvantages
  - Asynchronous transport; no guarantee of delivery, no guarantee that partner (external) shares acceptance of contract.
  - Messages will be much larger than binary (10x or more) [can compress]

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, XML needs a carrier protocol to transfer the data. So, I can we can use any transport data transport mechanism like transport we should not mix up this transport with our transport layer protocols. So, it is a transport mechanism means un-lining the network and by which you can do that http, https, smtp and so on and so forth. And there are some requirement and requirements and disadvantages of the same this type of XML messaging.

(Refer Slide Time: 19:55)



### Common messaging model

- XML over HTTP
  - Use HTTP to transport XML messages
  - ```
POST /path/to/interface.pl HTTP/1.1
Referer: http://www.foo.org/myClient.html
User-agent: db-server-olk
Accept-encoding: gzip
Accept-charset: iso-8859-1, utf-8, ucs
Content-type: application/xml; charset=utf-8
Content-length: 13221
. . .

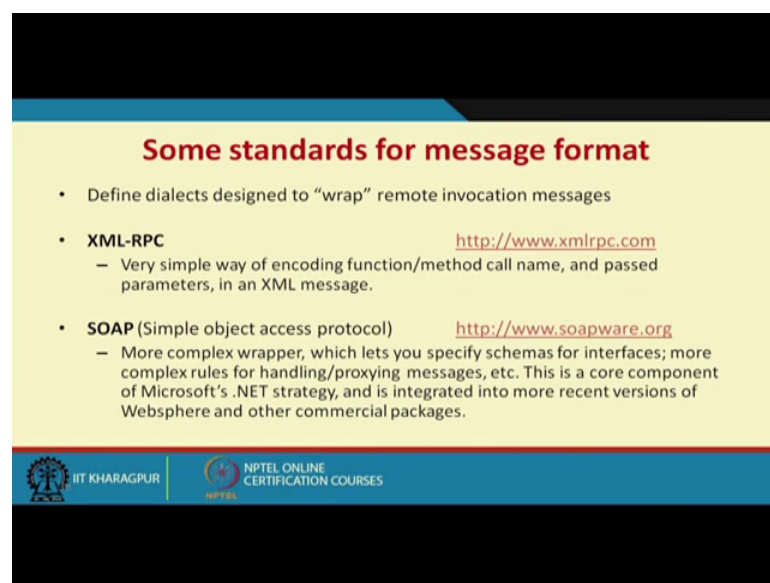
<?xml version="1.0" encoding="utf-8" ?>
<message> . . . Markup in message . . .
</message>
```

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

And there are common messaging models. So, a using a http a http to protocol for XML messages, in other since this XML messages are if you see this is a post at said that this portion of the thing is a standard protocol http protocol mechanisms.

Whereas, it is the XML message or the XML document is embedded into the things. So, that it is a envelope which is the http protocol itself and then it is embedded into the stuff.

(Refer Slide Time: 20:32)



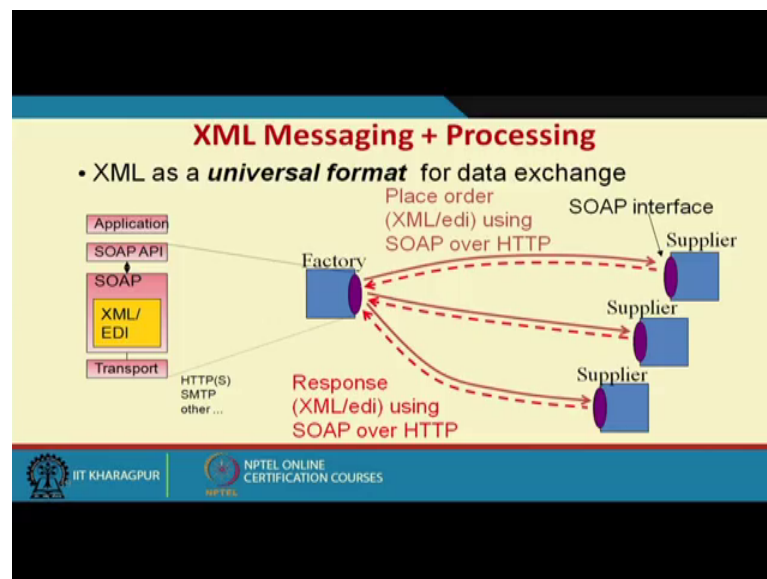
**Some standards for message format**

- Define dialects designed to “wrap” remote invocation messages
- **XML-RPC** <http://www.xmlrpc.com>
  - Very simple way of encoding function/method call name, and passed parameters, in an XML message.
- **SOAP** (Simple object access protocol) <http://www.soapware.org>
  - More complex wrapper, which lets you specify schemas for interfaces; more complex rules for handling/proxying messages, etc. This is a core component of Microsoft’s .NET strategy, and is integrated into more recent versions of Websphere and other commercial packages.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, there are some of the standards for message format like XML RPC. So, a very simple way of encoding function method calls call name and passed in the things. So, it is one of the message format is soap simple object access protocol primarily used in web services in a service oriented architecture. So, more complex wrapper which let us you specify schemas for interface more complex rules for handling and proxying messages and so and so forth.

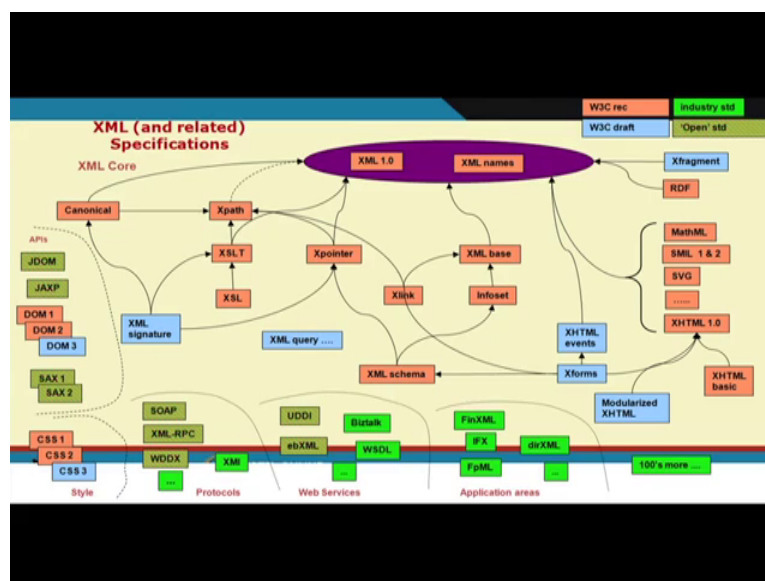
(Refer Slide Time: 21:13)



So, that can be done through this type of soap message format. And if you see again the XML messaging plus processing XML is a universal format for data exchange right. So, I have this application with soap API, and from here I can basically 2 different type of supplier. So, interact with different type of supplier though, at the backbone the overall it is the same XML applications which interacts with different other applications over http. So, it is a soap message which is which piggyback on http and carried over as a as a payload of this http and deliver at the things the other entities extracts and deliver to that particular applications which is running.



(Refer Slide Time: 22:12)



So, if we look at this family of XML all right. So, if you look at this is this middle portion is the cores of the things like we have a XML, XSLT, XSL, X path, X pointer and so forth we have not covered or we have not discussed on all the component it is first of all that amount of time is not there. And Secondly, we have we want to have a fill that how this allows me or what are the typical properties of XML is there. Those who are interested in a more looking at the XML can follow any standard book for event w 3 see a school tutorial and try to do some small projects of the things.

Nevertheless; so, we have this sort of APIs these are different styling, these are different protocols, web services application areas where the XML feed chain. And there can be different other type of things. So, what it tries to show the versatile nature of this XML and it is companion technologies. This XML and this community plays a vital role in realizing this distributed structure of systems right. So, I have distributed systems which are which are talking to each other and doing that is why I have So, messaging WSDL messaging WSDL UDDI which are primarily XML based.

And so, and we have different type of other namespaces like mathml and there are SVG for generating graphics and type of things. So, this shows this XML is a is a versatile in nature in allowing data transformation and if I want to represent this data then I require some styling either through XSLT or some XLD some style sheets which allows me to

represent the data and type of things I even; I can generate a HTML file which can be viewed in to the viewed at the other end, right.

So, we basically have what we had is a basic very basic introductory things or XML try to see that what are the major components of the XML, and how this component help us in realizing interoperability. If you look at the cloud or any distributed systems where the sort of interact interfaces interacting or interoperability needs are there XML is the de facto language to realize this. So, with this we will conclude today for our this basics of XML.

Thank you.