Internet and the Web

Code: internet-intro

Author and version

- Daniel K. Schneider
- Email: Daniel.Schneider@unige.ch

- Version: 0.2 (modified 16/1/11 by DKS)

Prerequisites

- Some knowledge about HTTP
- Some knowledge of XML (if possible)

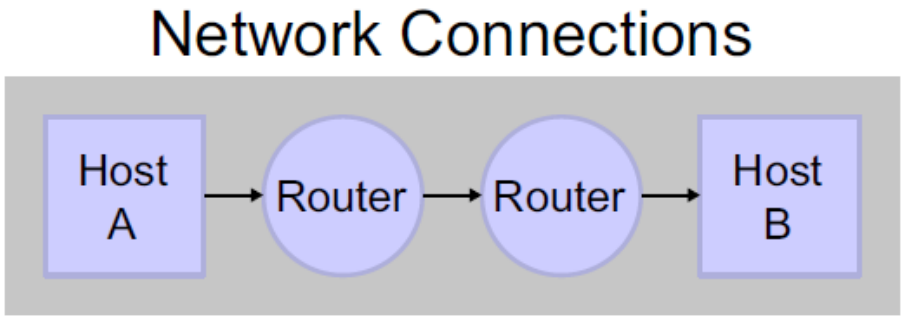
Objectives

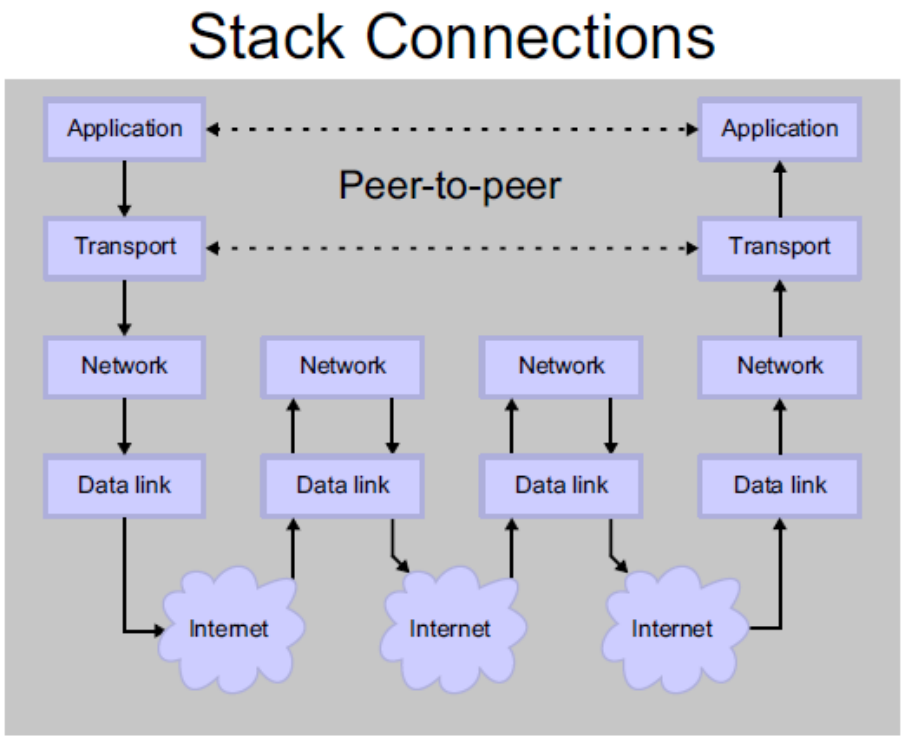
- Understand that Internet is a complex layert construct of protocols and services
- Disclaimer: ***DRAFT***

1. Table of Contents

2. What do we mean by Internet ?

- Internet refers to a complex stack of technologies (starting with physical layers up to application layers such as the world-wide web). An application talking to another application must use a transport protocol that relies on other lower level networking technology





2.1 Networking history

1960

Licklider (1960) wrote "Man-Computer Symbiosis": “ Man-computer symbiosis is an expected development in cooperative interaction between men and electronic computers. It will involve very close coupling between the human and the electronic members of the partnership. The main aims are 1) to let computers facilitate formulative thinking as they now facilitate the solution of formulated problems, and 2) to enable men and computers to cooperate in making decisions and controlling complex situations without inflexible dependence on predetermined programs.”

1962 Packet switching

The basis of all modern computer networks) A message can be broken down into packets like:

sender - receiver - message

1969 Arpanet and telnet

In December 1969, the first version of Arpanet (Internet) went online. It connected four computers from four universities (UCLA, Stanford Research Institute, UCSB, and the University of Utah). The project leader was Bob Kahn from BBN (Cambridge,MA).

Telnet (TELecommunication NETwork): Remote connection to another computer over a internet-like network.

1971 File transfer protocol FTP

Early version of FTP, revised in 1980 and 1985. Still popular (but consider using SFTP or SCP instead, since FTP is inherently insecure).

1972 Email

Ray Tomlinson (BBN) created the first e-mail program.

1973 Birth of Transmission Control Protocol (TCP):

According to Vint Cerf's FAQ: “ During 1973, we developed the concepts underlying the Internet and prepared a preliminary paper in September of that year that we presented to the International Network Working Group (INWG). In December 1974 the first full draft of TCP was produced.”

1978 TCP/IP

TCP/IP, the main technical pillar of Internet emerged in mid-late 1978 in nearly final form and was finalized in 1991. “The Internet protocol suite is the set of communications protocols that implements the protocol stack on which the Internet and many commercial networks run. It is part of the TCP/IP protocol suite, which is named after two of the most important protocols in it: the Transmission Control Protocol (TCP) and the Internet Protocol (IP), which were also the first two networking protocols defined.

1989 Archie

Peter Deutsch et al. (McGill University in Montreal) created Archie, an index machine for public ftp sites. Something like the grandfather of Google. This service allowed people to find software and texts.

1991 Gopher

The University of Minnesota developer gopher named after a mascot but also means "go fer". Gopher was a user-friendly server that allowed administrators to build menus to access local or remote files and services (e.g. phone directories, library interfaces).

1992 The Web

im Berners-Lee et al. invented the World Wide Web with its two main components: HTTP and HTML.

1998 XML

Soon after, the first XML contents applications (e.g. XHTML and SVG) and XML-based networking application standards emerged (e.g. SOAP and XML-RPC).

2.2 The network protocol stack

A. The five-layer TCP/IP model

url: **http://en.wikipedia.org/wiki/TCP/IP_model**

The TCP/IP model or Internet reference model is a layered abstract description for communications and computer network protocol design. Created in the 1970s by DARPA for use in developing the Internet's protocols. Still the most widely adopted view.

The five-layer TCP/IP model
5. Application layer
DHCP · DNS · FTP · Gopher · HTTP · IMAP4 · IRC · NNTP · XMPP · POP3 · SIP · SMTP · SNMP · SSH · TELNET · RPC · RTCP · RTSP · TLS · SDP · SOAP · GTP · STUN · NTP · (more)
4. Transport layer
TCP · UDP · DCCP · SCTP · RTP · RSVP · IGMP · (more)
3. Network/Internet layer
IP (IPv4 · IPv6) · OSPF · IS-IS · BGP · IPsec · ARP · RARP · RIP · ICMP · ICMPv6 · (more)
2. Data link layer
802.11 · 802.16 · Wi-Fi · WiMAX · ATM · DTM · Token ring · Ethernet · FDDI · Frame Relay · GPRS · EVDO · HSPA · HDLC · PPP · PPTP · L2TP · ISDN · (more)
1. Physical layer
Ethernet physical layer · Modems · PLC · SONET/SDH · G.709 · Optical fiber · Coaxial cable · Twisted pair · (more)

B. The 7-layer OSI model

1979. OSI has two major components: an abstract model of networking (the Basic Reference Model, or seven-layer model) and a set of concrete protocols.

OSI Model			
	Data unit	Layer	Function
Host layers	Data	7. Application	Network process to application
		6. Presentation	Data representation and encryption
		5. Session	Interhost communication
	Segment	4. Transport	End-to-end connections and reliability (TCP)
Media layers	Packet/Datagram	3. Network	Path determination and logical addressing (IP)
	Frame	2. Data link	Physical addressing (MAC & LLC)
	Bit	1. Physical	Media, signal and binary transmission

Session

- controls the dialogues/connections (sessions) between computer

Presentation

- syntax and semantics, e.g. Abstract Syntax Notation One (ASN.1) or XML

Application

- things like FTP, HTTP etc. (i.e. services used by application programs)

2.3 Application layers and services

- Usually rely on transport procols like TCP and UDP
- Common servers have specific ports assigned to them (HTTP has port 80; FTP has port 21; etc.)

Important basic Internet services

- File Transfer e.g. FTP (TCP/IP Protocol)
- Mail Transfer.g. SMTP/POP3/IMAP (TCP/IP protocols)
- WWW, e.g. HTTP (TCP/IP protocol), an extension of the telnet protocol

Web services (more recent services mostly built on top of HTTP)

- There are a variety of specifications associated with web services. These specifications are in varying degrees of maturity and are maintained or supported by various standards bodies and entities. Specifications may complement, overlap, and compete with each other. (Wikipedia).

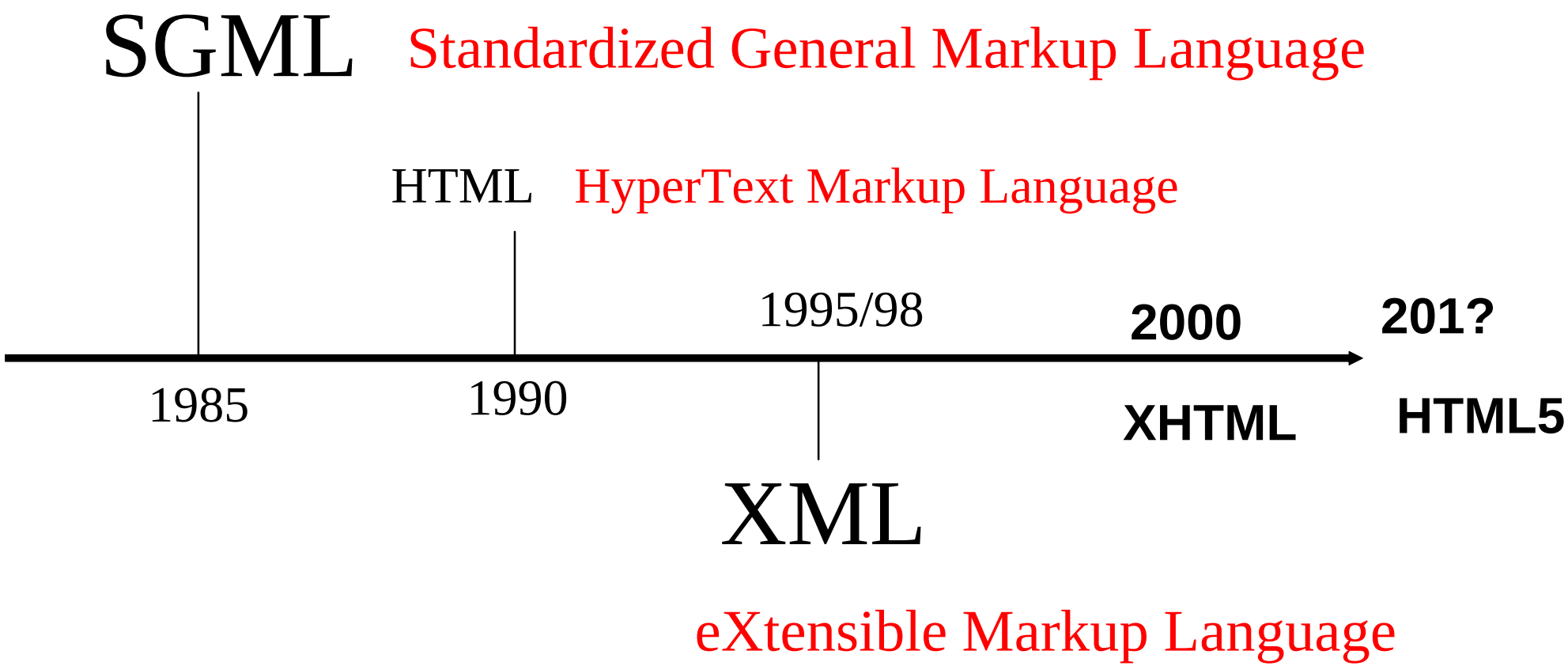
2.4 Note on end user applications

End user applications ***use*** the services of the application layer, but are not part of the application layer itself.

- File Transfer applications use e.g. FTP (TCP/IP Protocol)
- Mail Transfer clients use e.g. SMTP/POP3/IMAP (TCP/IP protocols)
- Web browsers using HTTP (TCP/IP protocol)

3. The Web I: Markup languages for content

3.1 History



- T. Bray, J. Paoli, and C. M. Sperberg-McQueen (Eds.), Extensible Markup Language (XML) 1.0, W3C Recommendation 10- February-1998, **http://www.w3.org/TR/1998/REC-xml-19980210/**

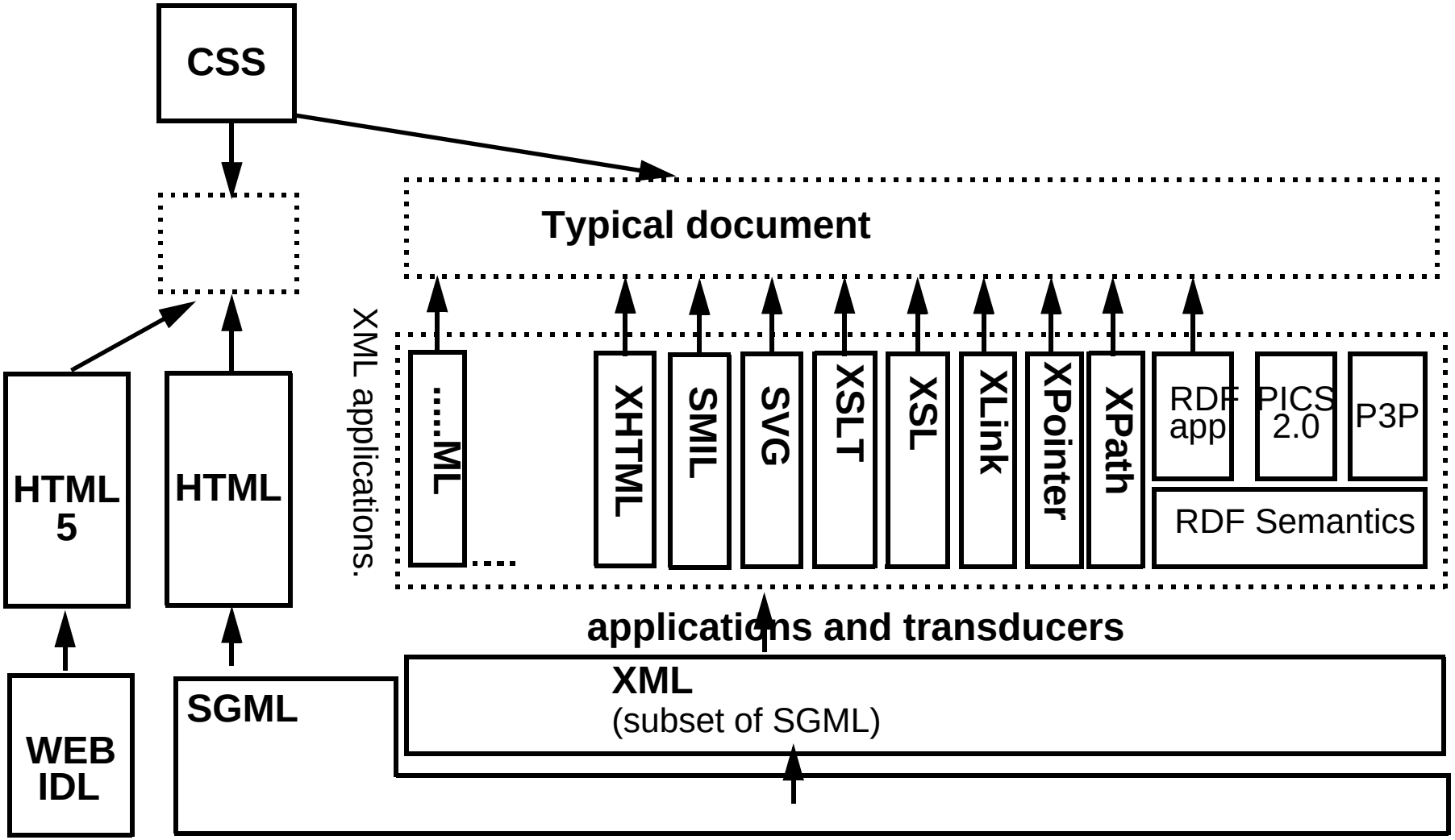
3.2 Two ways to look at XML

(1) XML as formalism to define vocabularies (also called applications)	(2) XML as a set of languages for defining:

Example DTD :
`<!ELEMENT page`
`(title, content, comment?)>`
`<!ELEMENT title (#PCDATA)>`
`<!ELEMENT content (#PCDATA)>`
`<!ELEMENT comment (#PCDATA)>`
 	
Example of an XML document:
`<page>`
`<title>Hello XML friend</title>`
`<content>`
`Here is some content :)`
`</content>`
`<comment>`
`Written by DKS/Tecfa,`
`</comment>`
`</page>`
	• Contents, Graphics, Style, Transformation and query languages, Data exchange protocols between machines, Programming languages-learning contents, Metadata, Administrative data
	
	... almost anything you can think of
	
	XML-languages can be categorized into:
	(1) XML ***accessories***, e.g. XML Schema
	• Extend the capabilities specified in XML
	• Intended for wide, general use
	(2) XML ***transducers***: e.g. XSLT
	• Convert XML input data into output
	• Associated with a processing model
	(3) XML ***applications***, e.g XHTML
	• Define grammars, constraints for a class of XML data
	• Intended for a specific application area

3.3 HTML and XML documents on the Web

- There is various XML markup on the Web (and increasingly so)
- Some documents may contain multiple vocabularies (e.g. HTML + SVG + MathML)
- Various transducers and accessories (XSLT, Xlink, XPointer, XPath, ...) may intervene



3.4 Content Markup with HTML and/or XHTML

Example 3-1: HTML Page

url: http://tecfa.unige.ch/guides/js/ex/coap/preq-html-page.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
   <title>Simple HTML Page</title>
    <meta http-equiv="Content-Type" content="text/html;charset=utf-8" >
  </head>

  <body>
    <h1>Simple HTML Page</h1>
    <p>Hello</p>

    <hr>
  </body>
</html>
```

Document type declaration

Encoding declaration

Body (page contents)

Notes:

- Navigators are very forgiving, you may omit the doctype, the encoding declarations, or even the head. But your page will not validate (not acceptable in most professional settings).

Example 3-2: XHTML Page

url: http://tecfa.unige.ch/guides/js/ex/coap/preq-xhtml-page.html

- All most recent HTML versions are XHTML.
- XHTML is an XML application (i.e. it uses the XML formalism).
- XHTML is more powerful than HTML, but simpler, stricter and somewhat less forgiving ....
- All tags are lower case, all tags must be closed (think “boxes” within “boxes”) !!!
- <html> must include xmlns, i.e. a namespace declaration
- The XML declaration on top is not absolutely mandatory, but the DOCTYPE is !

```
<?xml version = "1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns = "http://www.w3.org/1999/xhtml">
 <head>
   <title>Object Model</title>
  <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
 </head>
 <body>
   <h1>Welcome to our Web page!</h1>
   <p>Enjoy ! </p>
 </body>
</html>
```

XML and Document type declaration

Namespace declaration

Encoding declaration

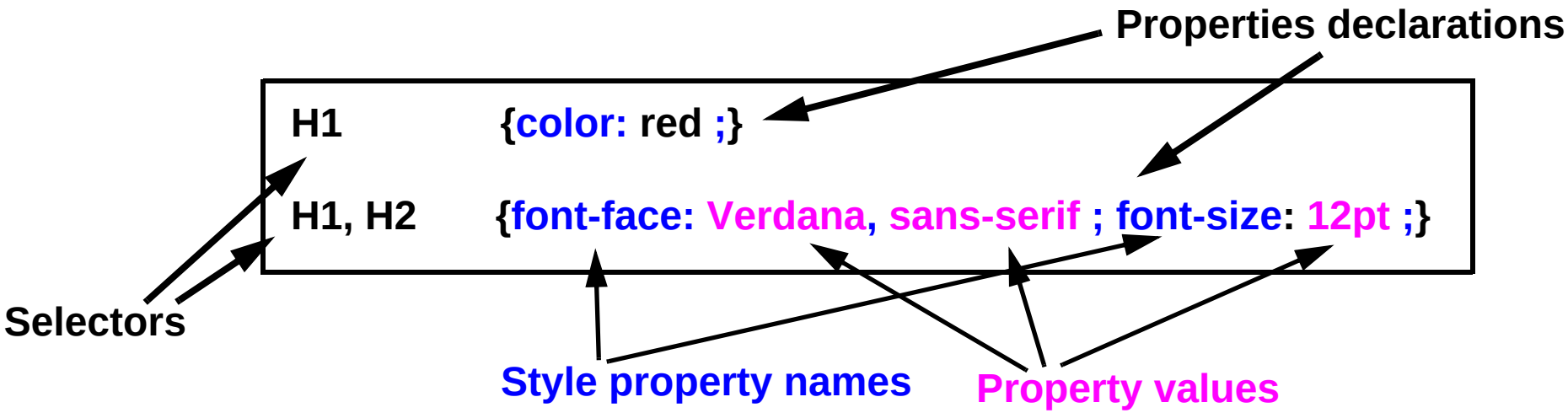
Body (page contents)

3.5 Styling with CSS

- Style sheet = set of ***rules*** that describe how to render XML or (X)HTML elements
- Each rule has two parts:
  - The ***selector***: defines to which elements a rule applies
  - The ***declaration***: defines rendering, i.e. defines values for style properties

Example 3-3: A simple HTML example

```
H1 { color: red }
P  { font-face: Verdana, sans-serif ; font-size: 12pt}
H1, H2, H3 { color : blue }
H1.ChapterTOC, H2.PeriodeTOC, H2.ExerciceTOC, H2.SectionTOC  {
        display: block;text-indent: 30pt;
        text-align: left; font-size: 14.000000pt;
        font-weight: Bold; font-family: "Times";
    }
```



3.6 JavaScript

A. Origin

- JavaScript was developed by Netscape (under the name LiveScript)
- Microsoft adopted it under the name JScript
- The name “JavaScript” reflects certain syntactic similarities with JAVA (but JavaScript and Java are **very** different !)

B. Principal use of JavaScript

- Interactive forms
- Interactive applications with HTML forms (par ex. tests or quizzes)
- Verification of HTML forms before sending them off for “server-side” treatment

Interactive pages (DHTML/AJAX)

- Richer HTML pages (ex. “highlighting”, menus, etc.)
- Applications that look like desktop applications (e.g. Google applications)
- Animations

User-centered contents

- Generation of HTML pages according to user profile

Plugin detection, etc.

- JavaScript is a programming language that runs inside your navigator

C. An example page with JS inside

- JavaScript functions are usually defined in the .head and within "**script**" tags
- JavaScript functions can be called in various ways from the body of the document
  - JavaScript instructions can be inserted within "**script**" tags (as in the head)
  - JavaScript also can be found within "inline" event handler definitions

```
<html xmlns = "http://www.w3.org/1999/xhtml">
   <head>
      <title>Object Model</title>
      <script type = "text/javascript">
         function start()
         {
            var p = document.getElementById("pText");
            alert( "Page contents : " + p.innerHTML );
            p.innerHTML = "Thanks for coming.";
            alert( "Page contents changed to : " + p.innerHTML );
            p.innerHTML = "Cool, isn't it ?";
         }
      </script>
   </head>

   <body onload = "start()">
      <p id = "pText">Welcome to our Web page!</p>
   </body>
</html>
```

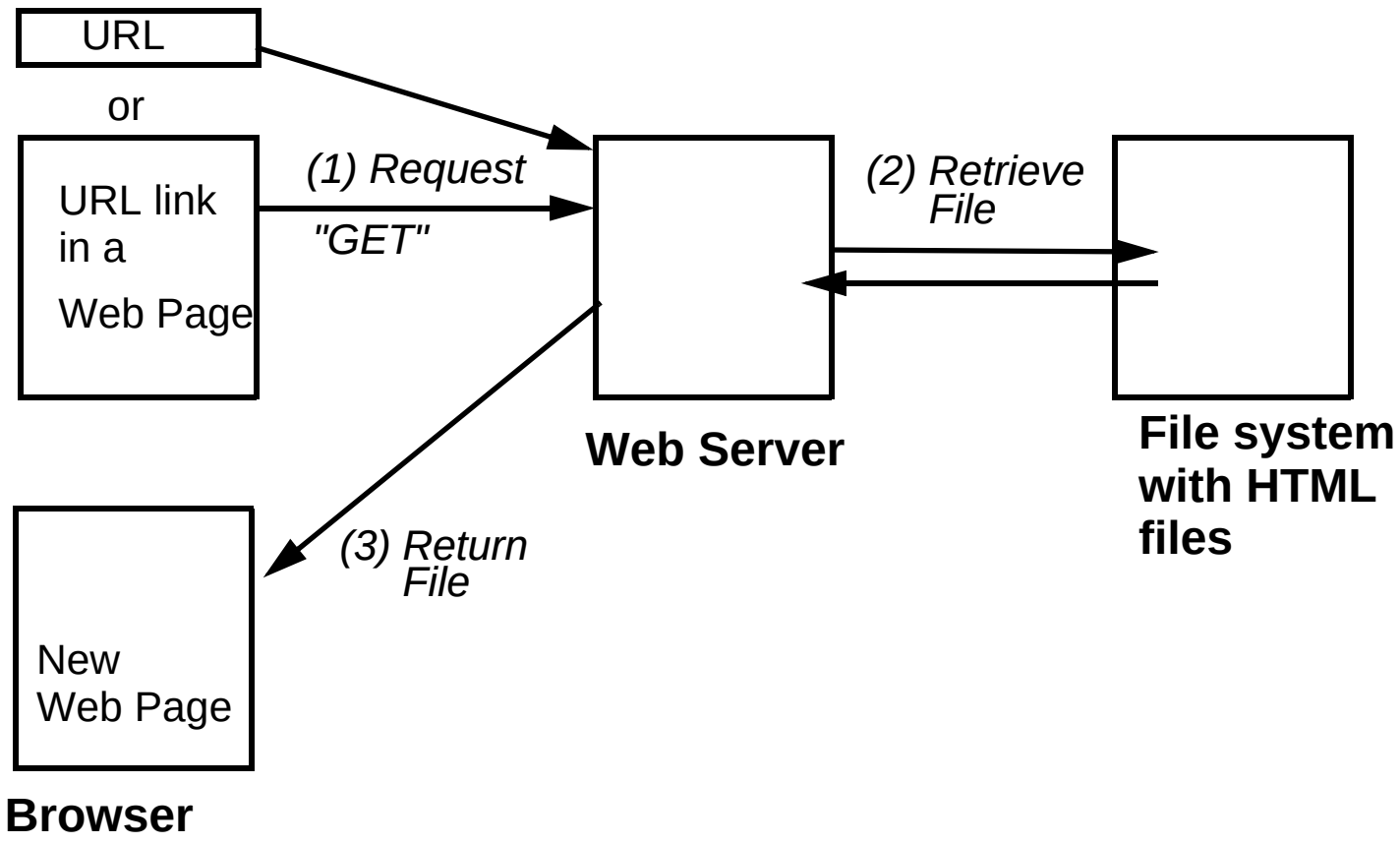
Definition of the event handling function "start"

Inline event and call of event handling function

- Note: This is just a FYI-type of slide about JavaScript ....

4. The Web II - Hypertext Transfer Protocol (HTTP)

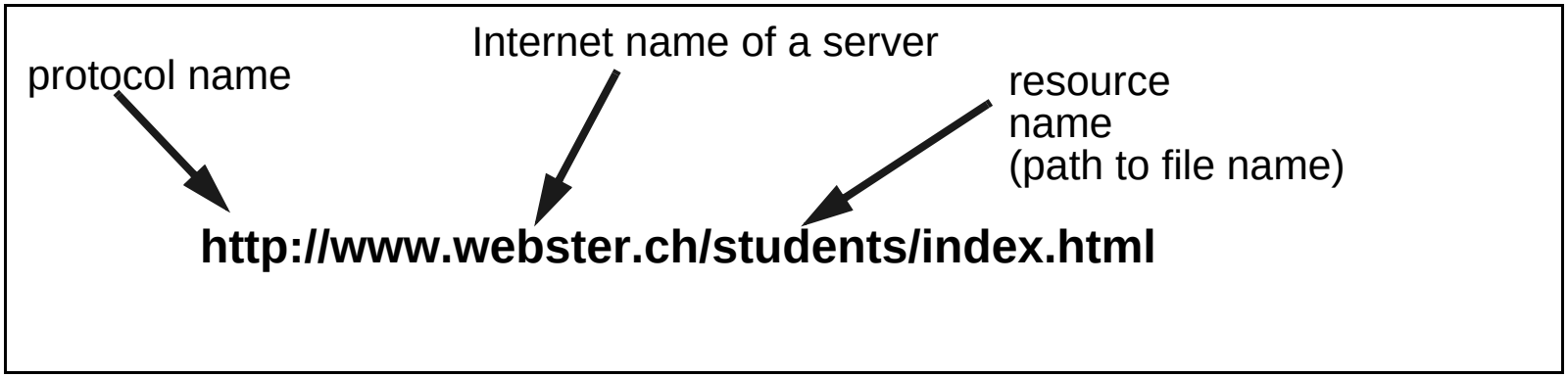
4.1 HTTP - Simple web pages



4.2 Defining resource addresses with URLs

- URL = Universal Resource Locator = unique addresses for resources.

  `Syntax: URL = protocol://address/resource name`



  Other protocols exist, e.g. ftp ...

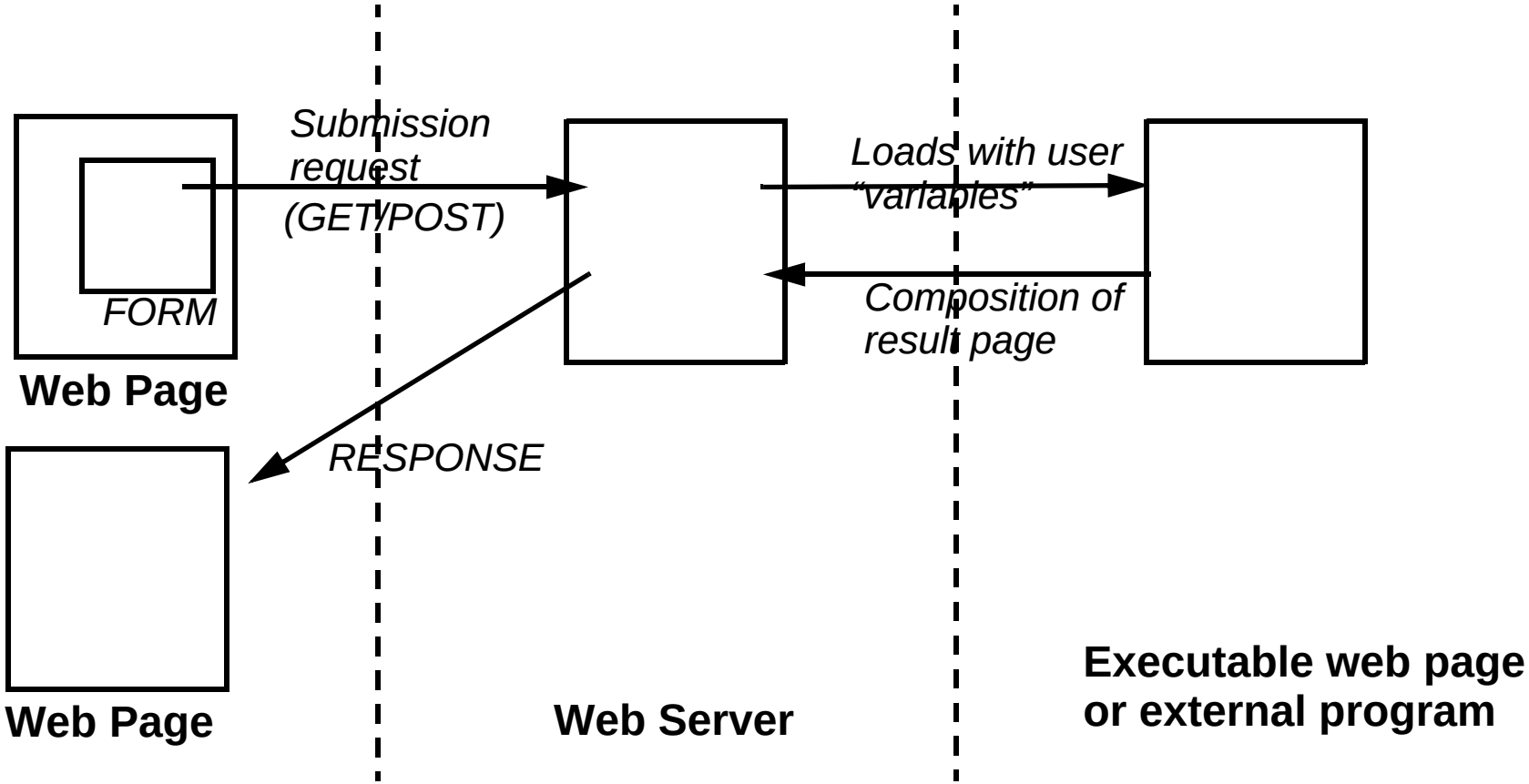
- Uniform Resource Names (URNs) are intended to serve as persistent, location-independent, universal resource identifiers (URIs). Examples:

  `urn:issn:1082-9873`
  `urn:doi:10.1000/1`

- URNs vs. URLs - Example of a URI in both URL and URI form (from RFC 3986):

```
http://example.com:8042/over/there?name=ferret#nose
 \_/   \______________/\_________/ \_________/ \__/
  |           |            |            |        |
 scheme     authority      path        query   fragment
  |   _____________________|__
 / \ /                        \
 urn:example:animal:ferret:nose
```

4.3 HTML Forms and server-side scripts



Form example (see next slide for HTML code):

Form example (truncated HTML code example)

```
<form method="post" name="searchForm"
      action="http://connections.webster.edu/cp/..../fileshare/Search.jsp">
<input name="groupID" value="24039" type="hidden">
<table align="center" border="0" cellpadding="0" cellspacing="5">
<tbody><tr>
   <td align="right"><span class="text10">Search for:  </span></td>
   <td align="right"><input name="searchName" value="Y" class="textform"
alt="Select to search file names" checked="checked" type="checkbox"></td>
   <td><span class="text10">File Names</span></td>
   <td align="right"><input name="searchFolder" value="Y" class="textform"
alt="Select to search folder names" checked="checked" type="checkbox"></td>
   <td><span class="text10">Folder Names  </span></td>
   <td align="right"><input name="searchContent" value="Y" class="textform"
alt="Select to search file content" checked="checked" type="checkbox"></td>
   <td><span class="text10">File Content</span></td>
</tr>
<tr>
   <td align="right"><span class="text10">that contain:</span></td>
  <td colspan="6"><span class="text10"><input class="textform" name="searchText"
size="60" maxlength="100" value="" type="text"></span></td>
   <td><input src="/cps/images/btns/search.gif" name="Search" alt="Search"
onclick="return searchRequest()" border="0" hspace="5" type="image"
vspace="5">></td>
</tr>
</tbody></table>
</form>
```

4.4 The HTTP Protocol - messages structure

- Hypertext Transfer Protocol (HTTP) is a communications protocol used to transfer or convey information on the World Wide Web. (Wikipedia)

  url: **http://en.wikipedia.org/wiki/Http**

- Sample message sent from browser to server

```
GET /index.html HTTP/1.1
Host: www.example.com
```

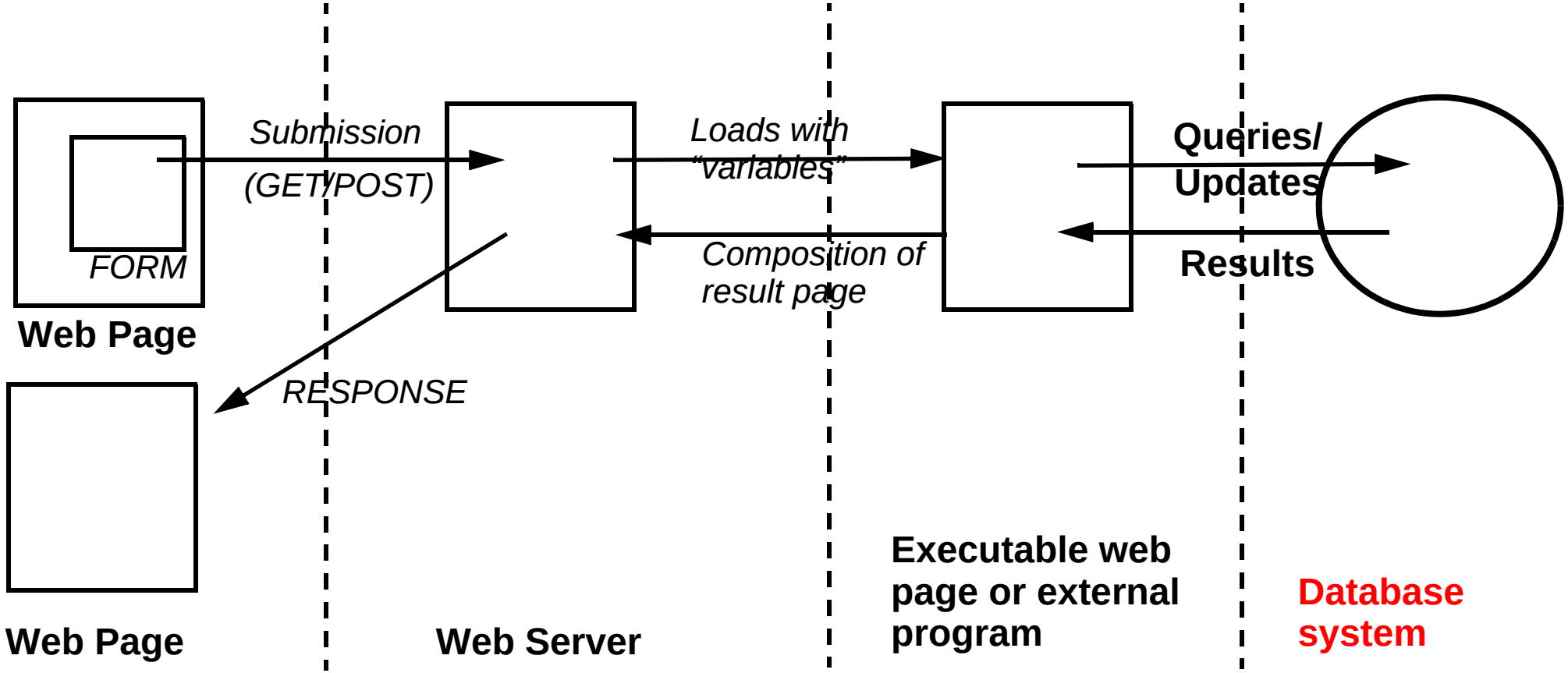
- Sample message sent from server to browser in response

```
HTTP/1.1 200 OK
Date: Mon, 23 May 2005 22:38:34 GMT
Server: Apache/1.3.27 (Unix)  (Red-Hat/Linux)
Last-Modified: Wed, 08 Jan 2003 23:11:55 GMT
Etag: "3f80f-1b6-3e1cb03b"
Accept-Ranges: bytes
Content-Length: 438
Connection: close
Content-Type: text/html; charset=UTF-8
```

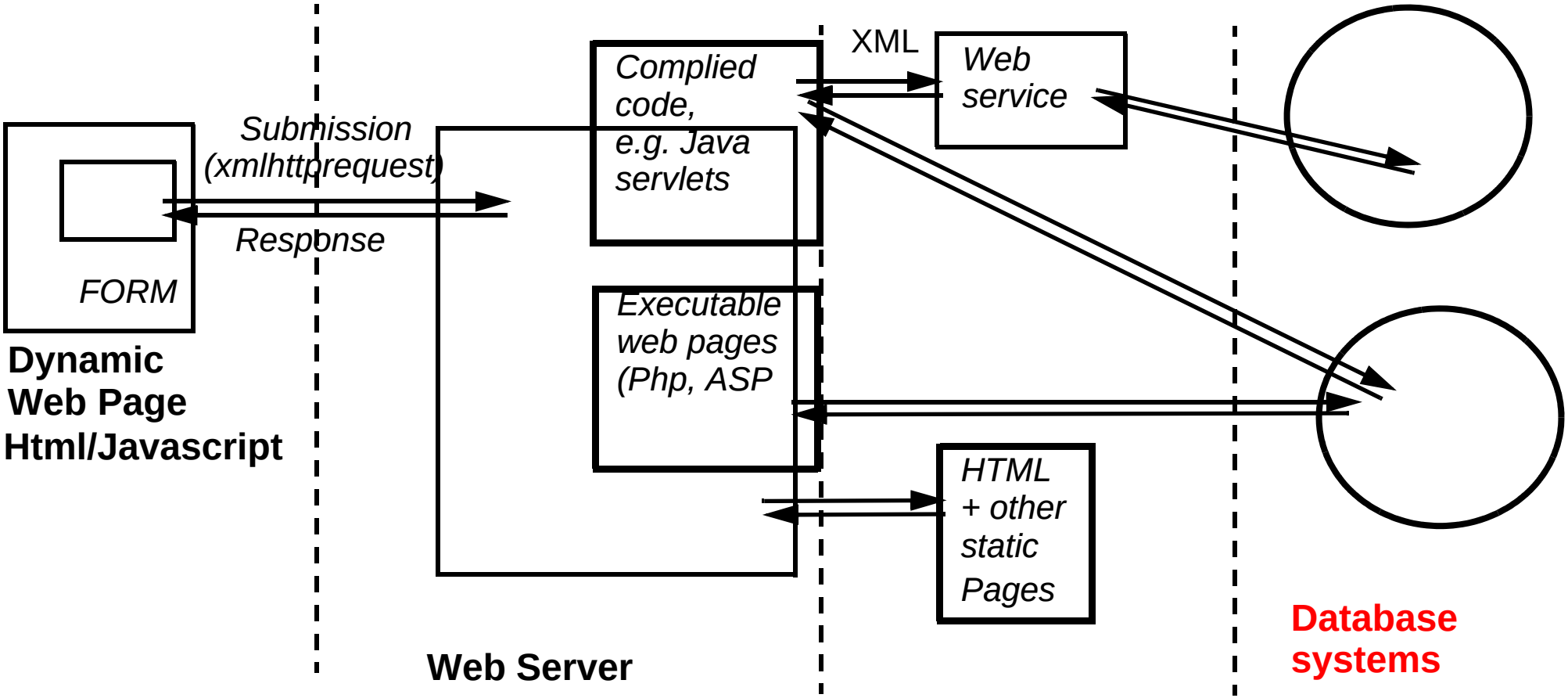
4.5 Web applications

- Web applications have a client side (e.g. simple HTML, HTML + JavaScript or another format like Flash) and a server side, e.g. a script + a database

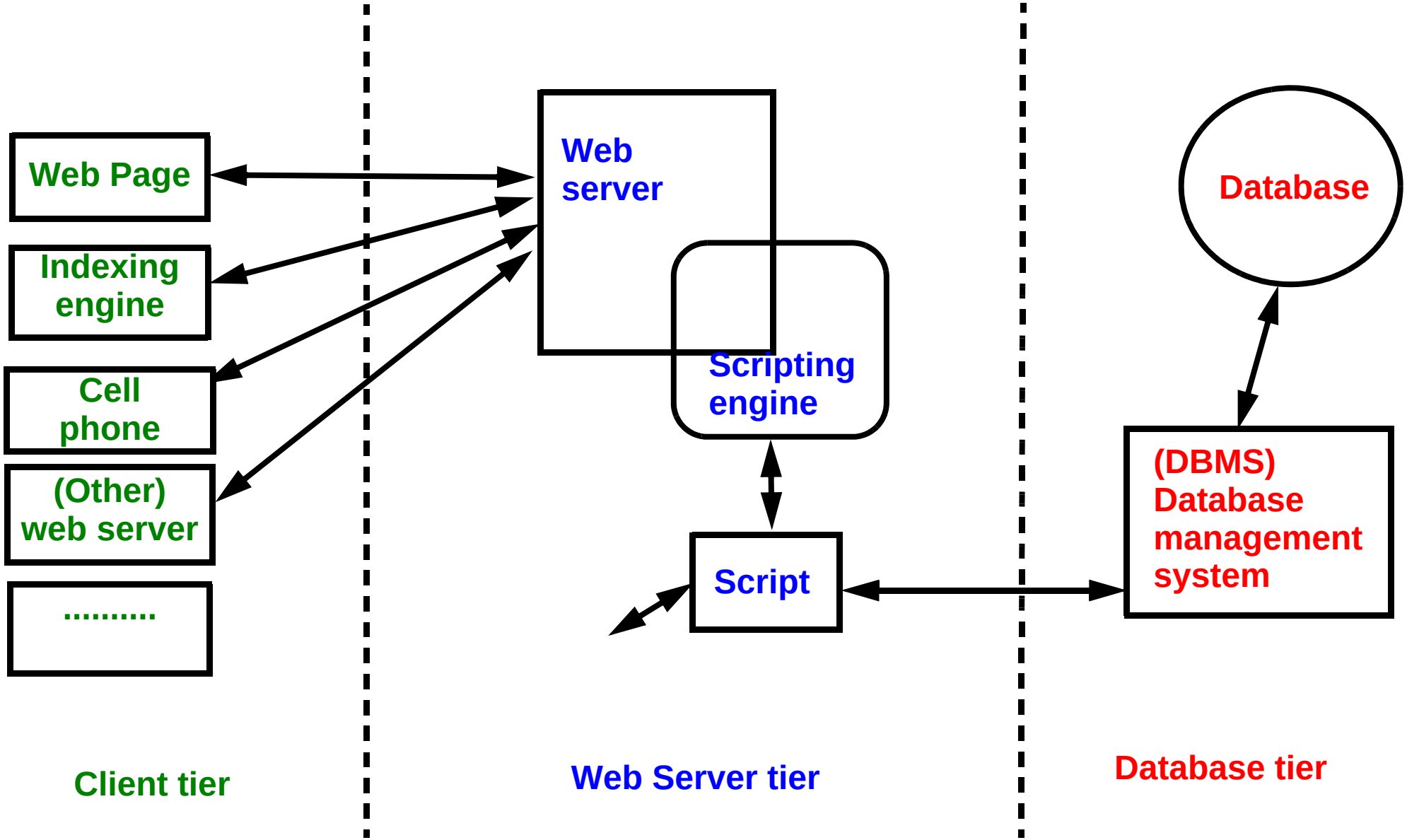
4.6 HTML Forms, server-side scripts and databases



4.7 A modern more complex setup ....



5. The typical three-tier architecture model



6. Types of databases and technologies

A global definition

"In computer science, a database is a **structured collection of records or data** that is stored in a computer system so that a computer program or person **using a query language** can consult it to answer queries. The records retrieved in answer to queries are information that can be used to make decisions. The computer program used to manage and query a database is known as a database management system (DBMS)." (Wikipedia, retrieved 22:30, 12 September 2007 (MEST)).

6.1 Types

1. Simple/flat (e.g. a table with user name, password and name, e.g. excel sheets). Rows are entries and colums define various data types that can be entered.
2. Hierarchical (e.g. LDAP authentication/address serveurs, some XML datastructures, Windows registry): Objects are inserted within other objects. Each object can have both properties and child objects.
3. Relational (mainstream today, in particular SQL databases): Data entries describe properties of an object like in the simple model, but can be linked to identifiers of other data entries.
4. Object and object-relational databases (not covered in this course)
5. Native XML datastores

6.2 Database connectivity

Direct connections

- APIs that allow a client (e.g. a database tool or a scripting language to connect directly to a database).
- Examples
  - Open Database Connectivity (ODBC) specification offers a procedural API for using SQL queries to access data. An implementation of ODBC will contain one or more applications, a core ODBC library, and one or more "database drivers". The core independant core library is an "interpreter" between applications and database drivers, whereas the database drivers contain the DBMS-specific details.
  - Java Database Connectivity (JDBC) is an API for the Java programming language that defines how a client may access a database. It provides methods for querying and updating data in a database. JDBC is oriented towards relational databases. JDBC may interact with OBDC drivers.

  url: **http://en.wikipedia.org/wiki/Java_Database_Connectivity**
  - ADO.NET is a set of software components that can be used by programmers to access data and data services. It is a part of the base class library that is included with the Microsoft .NET Framework
  - Proprietary, e.g. the MySQL libraries in PHP. Note: Most professional PHP programmers use a database independant library such as ADODB.
  - XML:DB API (XAPI) is designed to enable a common access mechanism to XML databases. The API enables the construction of applications to store, retrieve, modify and query data that is stored in an XML database. These facilities are intended to enable the construction of applications for any XML database that claims conformance with the XML:DB API. The API can be considered generally equivalent to technologies such as ODBC, JDBC or Perl DBI.

Through web services

- An application will send requests to a webservice that in turn will then send database queries.

7. Web services

The W3C defines a Web service as a software system designed to support interoperable Machine to Machine interaction over a network. Web services are frequently just Web APIs that can be accessed over a network (e.g. Internet), and executed on a remote system hosting the requested services.

7.1 Overview

The Web service protocol stack is an evolving set of application layer protocols used to define, discover, and implement Web services. The core protocol stack has 4 layers:

- Transport: E.g. HTTP, SMTP, FTP, and newer protocols.
- XML messaging: E.g. Simple Object Access Protocol (SOAP) or XML-RPC. These define messages containing a service request and a response. SOAP and XML-RPC etc. are independent of any particular transport and implementation technology.
- Service description: E.g. Web Services Description Language (WSDL) - describes what a service does in a machine readable way
- Service discovery: E.g. Universal Discovery, Description, Integration (UDDI) - a service to publish available services

url: **http://en.wikipedia.org/wiki/Web_service**

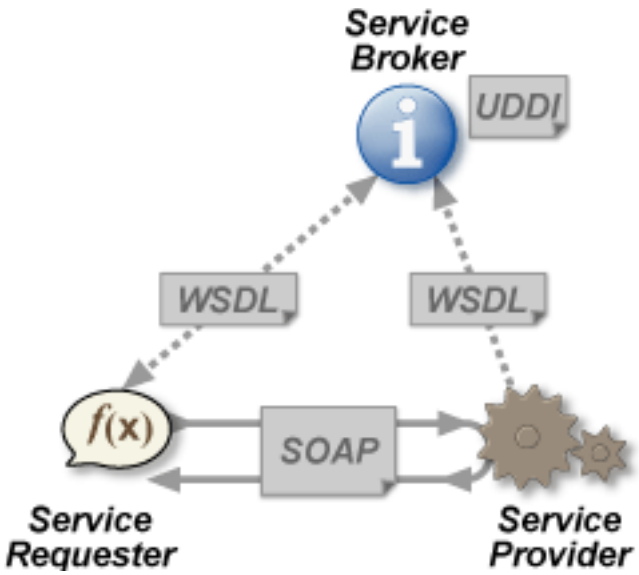
url: **http://en.wikipedia.org/wiki/List_of_Web_service_specifications**

url: **http://en.wikipedia.org/wiki/List_of_web_service_protocols**

url: **http://en.wikipedia.org/wiki/List_of_Web_service_Frameworks**

7.2 SOAP, UDDI and WSDL

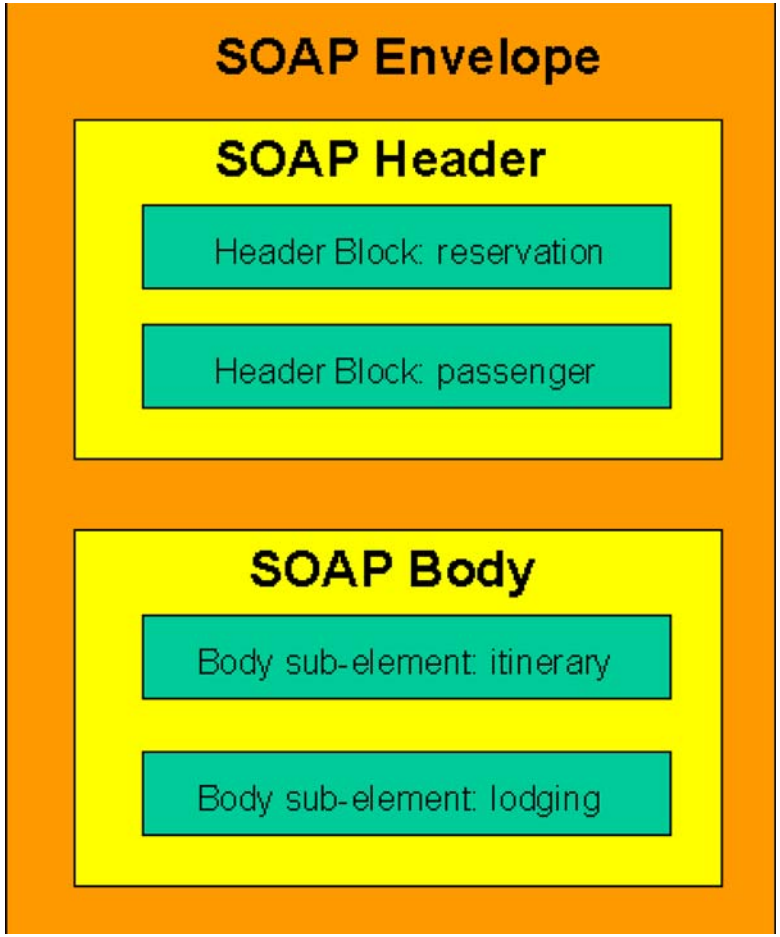
- SOAP - An XML-based, extensible message envelope format with "bindings" to underlying protocols. The primary protocols are HTTP and HTTPS, although bindings for others, including SMTP and XMPP, have been written.
  - SOAP originally stood for Simple Object Access Protocol, and lately also Service Oriented Architecture Protocol. SOAP became a W3C standard in 1993. Before SOAP existed other protocols such as XML-RPC (still popular) and CORBA or DCOM. Services like CORBA are less popular since they can't go past most firewalls and since they are more complicated.
- Web Services Description Language (WSDL) - An XML format that allows service interfaces to be described along with the details of their bindings to specific protocols. Typically used to generate server and client code, and for configuration.
- Universal Description Discovery and Integration (UDDI) - A protocol for publishing and discovering metadata about Web services that enables applications to find them, either at design time or runtime.



7.3 SOAP

url: **http://en.wikipedia.org/wiki/SOAP**

- SOAP mostly uses either HTTP or HTTPS as transport protocol (so HTTP which is an application layer actually turns into a transport layer ...).
- With SOAP one can either send messages or remote procedure calls to another application
- Structure of a SOAP message to make a passenger reservation.



Example code of the passenger reservation

```
<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
 <env:Header>
  <m:reservation xmlns:m="http://travelcompany.example.org/reservation"
            env:role="http://www.w3.org/2003/05/soap-envelope/role/next"
        env:mustUnderstand="true">
   <m:reference>uuid:093a2da1-q345-739r-ba5d-pqff98fe8j7d</m:reference>
   <m:dateAndTime>2001-11-29T13:20:00.000-05:00</m:dateAndTime>
  </m:reservation>
  <n:passenger xmlns:n="http://mycompany.example.com/employees"  env:role="http://www.w3.org/2003/05/soap-envelope/role/next"
        env:mustUnderstand="true">
   <n:name>Åke Jógvan Øyvind</n:name>
  </n:passenger>
 </env:Header>
  <env:Body>
  <p:itinerary xmlns:p="http://travelcompany.example.org/reservation/travel">
   <p:departure>
    <p:departing>New York</p:departing>   <p:arriving>Los Angeles</p:arriving>
    <p:departureDate>2001-12-14</p:departureDate>  <p:departureTime>late afternoon</p:departureTime>
    <p:seatPreference>aisle</p:seatPreference>
   </p:departure>
   <p:return>
    <p:departing>Los Angeles</p:departing>  <p:arriving>New York</p:arriving>
    <p:departureDate>2001-12-20</p:departureDate> <p:departureTime>mid-morning</p:departureTime>
    <p:seatPreference/>
   </p:return>
  </p:itinerary>
  <q:lodging
   xmlns:q="http://travelcompany.example.org/reservation/hotels">
   <q:preference>none</q:preference>
  </q:lodging>
  </env:Body>
</env:Envelope>
```

7.4 XML-RPC

- XML-RPC is a remote procedure call protocol which uses XML to encode its calls and HTTP as a transport mechanism.

  url: **http://en.wikipedia.org/wiki/XML-RPC**
- XML-RPC was first created by Dave Winer of UserLand Software in 1998 with Microsoft. As new functionality was introduced, the standard evolved into what is now SOAP. Some people still prefer XML-RPC to SOAP because of its simplicity, minimalism, and ease of use.
- An example of a typical XML-RPC request would be:

```
<?xml version="1.0"?>
<methodCall>
  <methodName>examples.getStateName</methodName>
  <params>
    <param>
        <value><i4>40</i4></value>
    </param>
  </params>
</methodCall>
```

- An example of a typical XML-RPC response would be:

```
<?xml version="1.0"?>
<methodResponse>
  <params>
    <param>
        <value><string>South Dakota</string></value>
    </param>
  </params>
</methodResponse>
```

7.5 REST

url: **http://en.wikipedia.org/wiki/REST**

- Representational State Transfer (REST) is a style of software architecture for systems like the World Wide Web. The term originated in a 2000 doctoral dissertation about the web written by Roy Fielding. It is very popular, because simple.
- Basically, REST only refers to a collection of architectural principles. The term is also often used to describe any simple interface that transmits domain-specific data over HTTP without an additional messaging layer such as SOAP. These two meanings can conflict as well as overlap.

Important REST concepts:

- the existence of resources (sources of specific information), each of which can be referred to using a single global identifier (a URI).
- A constrained set of well defined operations, e.g. GET, POST, PUT, DELETE (defined by the HTTP protocol).
- A constrained set of content types

8. Things a web designer should know

8.1 Content

- Markup languages - XHTML, HTML
- Understand the role of XML and XML niche formats (SVG, MathML, VoiceML, etc.)
- Styling with CSS
- XML transducers like XSLT
- Handling external multimedia formats (Bitmaps, sounds, video)
- Interactive multimedia (Flash)

8.2 Web applications

- Create simple database applications
  - Create typed tables with web 2.0 services, enter data and connect these with your own webpages
  - generate very simple PHP/MySQL applications, e.g. with generator software
- Install and/or use XAMP (Apache/MySQL/PhP) systems:
  - Simple web server configuration
  - Understand how database systems like MySQL work
  - Understand what the PHP scripting engine can do (and its dangers)
- Configure LAMP/WAMP-based applications (various kinds of portals)
  - Create databases and/or learn how to connect to a database with a user name and a password
  - Give the right answers to installation scripts
  - Configure the applications (modules, skins, users, etc.)

- Repair data of web applications
  - E.g. remove SPAM with SQL queries
  - Extend applications with modules (know how to submit SQL code to a db management tool)
- Understand various roles of XML in modern architectures, e.g.
  - Transform database output from XML to HTML
  - Learn XQuery/XQuery update basics
- Understand the role of LDAP and similar authentication/address book applications

  ... At the end of the course you should have acquired the necessary foundations

8.3 User interface design and usability

- ... not addressed here