Web-Technologien in Beschleuniger Kontrollen

Kay Rehlich, DESY MVP

- Am Beispiel eLogBook
- Die Technologien:
  HTML, XML, XSL, XPath, DOM, Applet, Servlet, JSP, TagLib
- Wie spielt das alles zusammen?
- Welche Web-Technologien sind für Kontrollsysteme nutzbar?

Motivation

TTF Logbooks

1998 .. 2001

... Remote operations ?

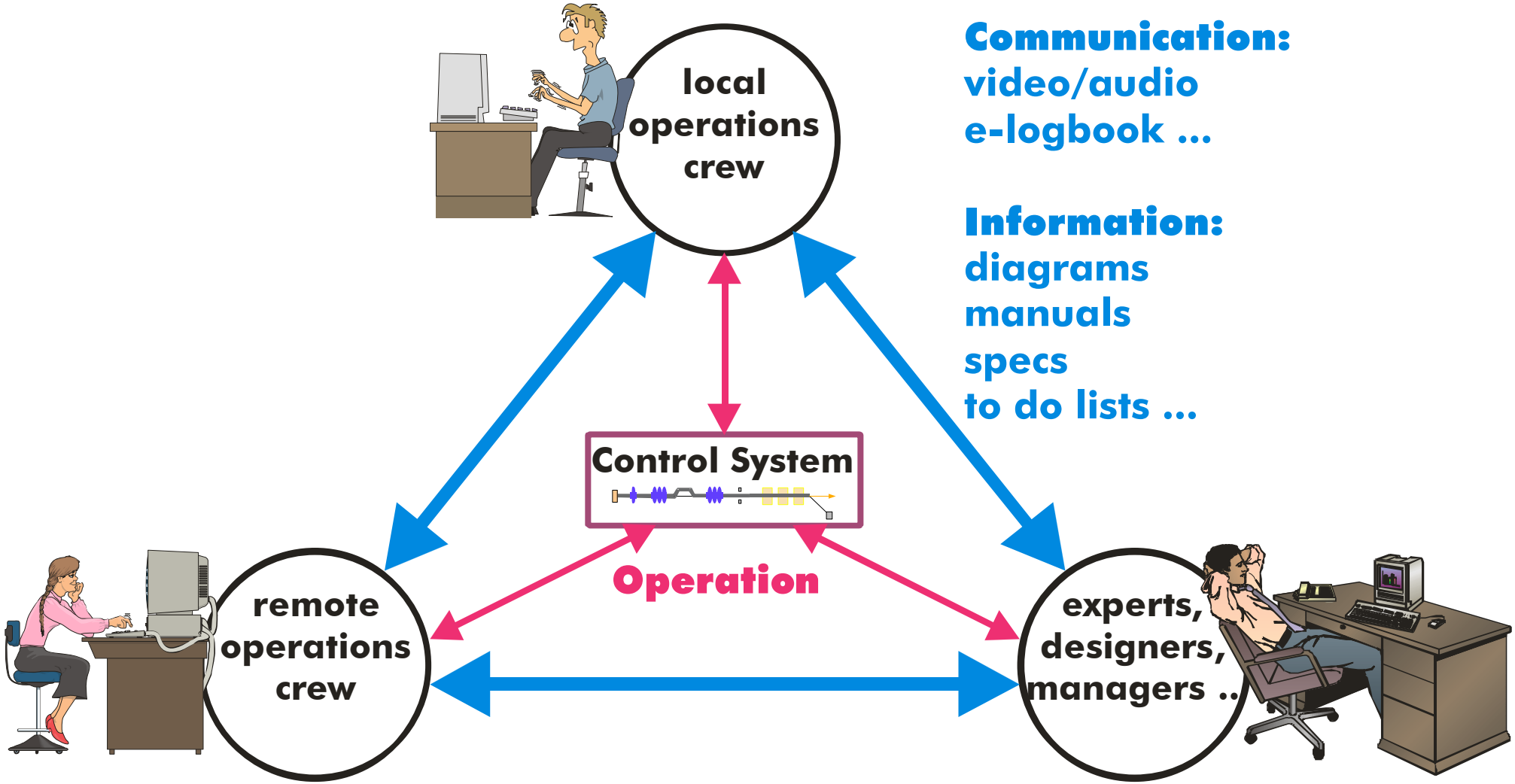
Search ?

Motivation cont.

- direct input from all kind of programs (PS print)
- better readable than hand writing
- involves more people in the machine operations
- allows experts to give comments, hints or warnings
- search function
- common place for all measurement results
- e-LogBook is a must for remote operations
- gain experience with new web technologies

➡ **e-LogBook is an important GAN tool**

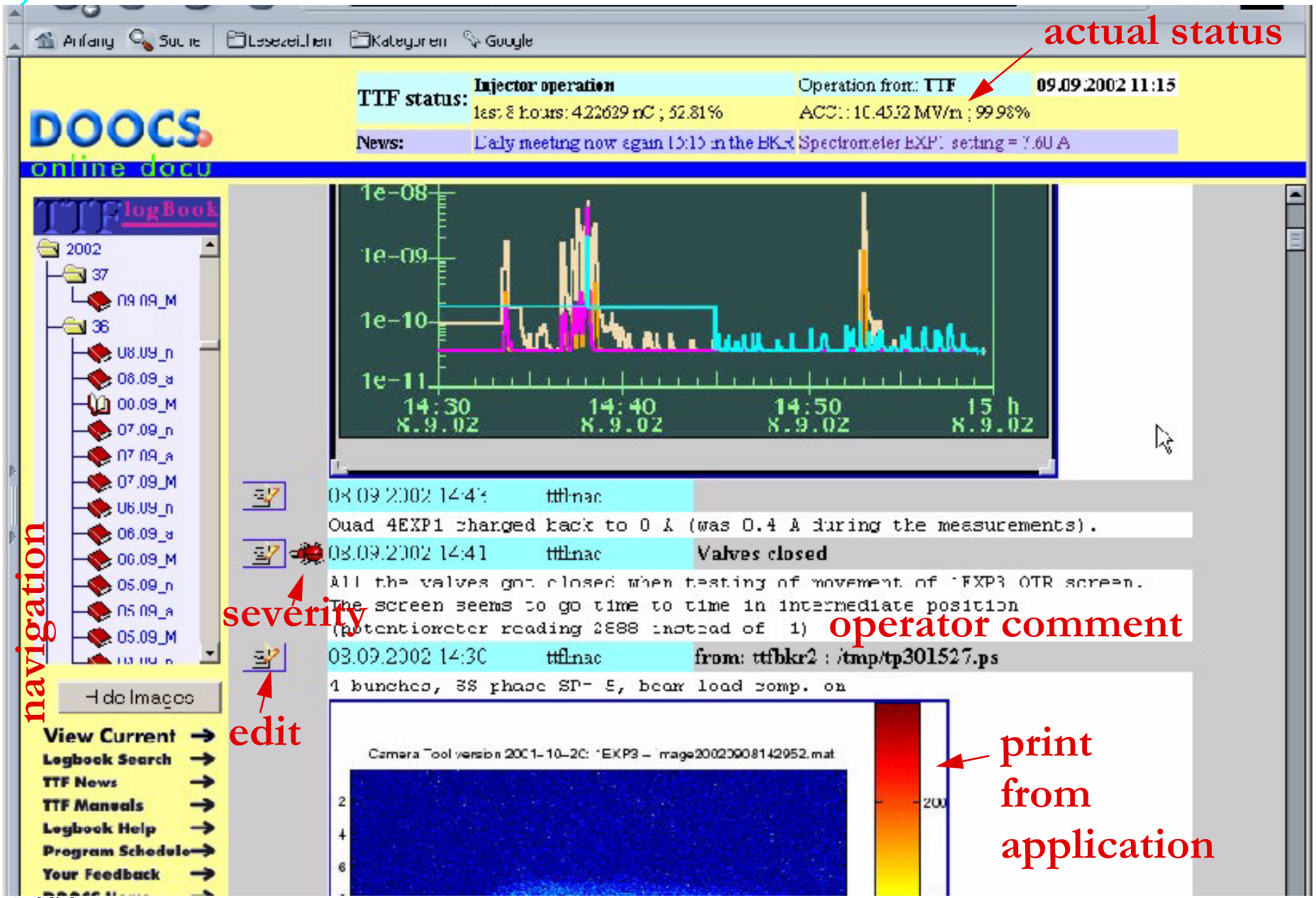
Requirements for Remote Operations



Communication:
video/audio
e-logbook ...

Information:
diagrams
manuals
specs
to do lists ...

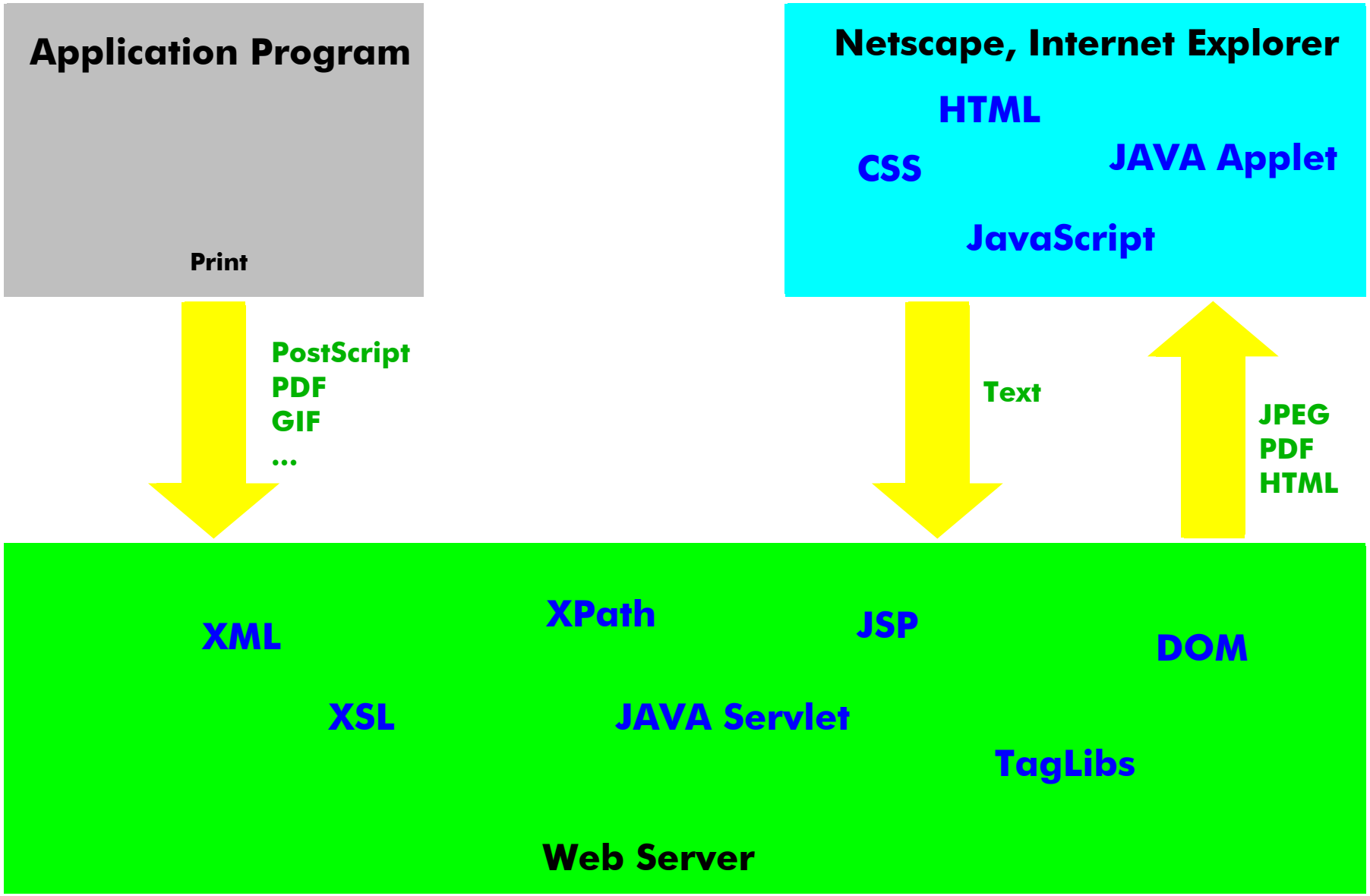
The eLogBook



Why Web-Technologies?

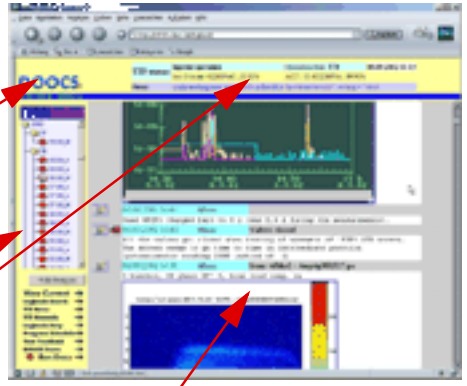
- Every body knows how to use Netscape or Internet Explorer
- Runs everywhere
- Can be integrated into different environments
  eLogBook is installed:
  ESRF, INFN (Milano), SLAC (SPPS, PEP II, Main Linac)
  Zeuthen (PITZ), Hamburg (TTF, HERA, H1, Groups)
- Link to other information sources

e-LogBook Technologies



HTML: Hyper Text Markup Language

```
<html>
  <head>
    <title>TTFlogbook</title>
  </head>
  <body>
    <frameset cols="145,*">
      <frameset rows="88,*">
        <frame src="logo.html" name="logo_frame">
        <frame src="left.jsp" name="left_frame">
      </frameset>
      <frameset rows="88,*">
        <frame src="top.jsp" name="top_frame">
        <frame src="/elog/servlet/XMLlist?file="xx/act_dir
          &xsl=/elogbook/xsl/elog.xsl" name="list_frame">
      </frameset>
    </frameset>
  </body>
</html>
```



JavaScript and Cascading Style Sheets (CSS)

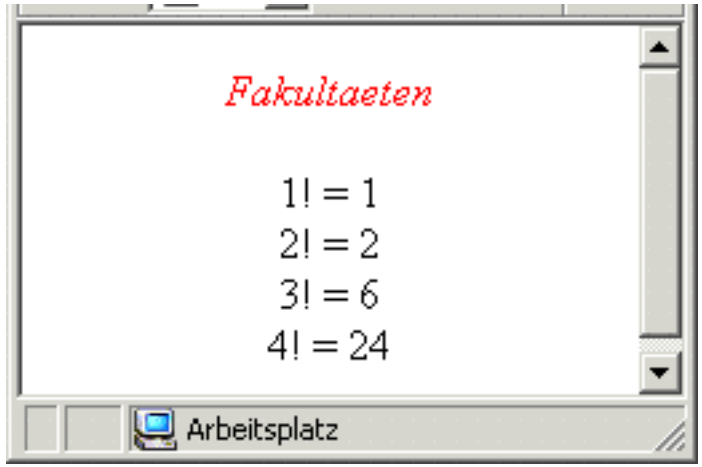
```
<html>
<body>
 <style type="text/css">
   body { text-align: center }
   p    { font-style: italic; color: red }
 </style>

 <script language="JavaScript">
   document.write("<p>Fakultaeten</p>");
   for (i=1, fak=1; i<5; i++, fak*=i) {
     document.write(i + "! = " + fak + "<br>");
   }
 </script>

</body>
</html>
```

CSS

JavaScript

JavaScript:

- is NOT JAVA
- is object oriented
- is an interpretor



XML: eXtensible Markup Language

```
<?xml version="1.0" encoding="iso-8859-1"?>
<list>
 <entry>
   <text>This is a text filed for the logbook!</text>
   <title>This is a new logbook</title>
   <isodate>2003-02-07</isodate>
   <time>19:24</time>
   <severity>NONE</severity>
   <category>USERLOG</category>
   <keywords>LinacStatus</keywords>
   <author>Kay Rehlich</author>
   <file>/sppselog/data/2003/08/02.16_a/new_file.jpeg</file>
   <link>/sppselog/data/2003/08/02.16_a/new_file.ps</link>
..
 </entry>
..
</list>
```

Start Tag: one logbook entry

End Tag

XSL: eXtensible Stylsheet Language

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/
1999/XSL/Transform">
  <xsl:template match="list">
    <html>
      <body BGCOLOR="#cccccc">
        <TABLE>
          <xsl:apply-templates select="entry">
            <xsl:sort order="descending" select="isodate"/>
          </xsl:apply-templates>
        </TABLE>
      </body>
    </html>
  </xsl:template>
```

Main Block:

- creates HTML page
- creates a table
- sorts 'entry' by 'isodate'
- calls 'entry' template for each 'entry' in 'list'

```
  <xsl:template match="entry">
    <TR>
      <TD BGCOLOR="#99ffff">
        <xsl:value-of select="date"/>
        <xsl:value-of select="substring(time,1,5)"/>
      </TD>
        <xsl:apply-templates select="author"/>
        <xsl:apply-templates select="title"/>
    </TR>
  </xsl:template>
...
</xsl:stylesheet>
```

Template Block for 'entry':

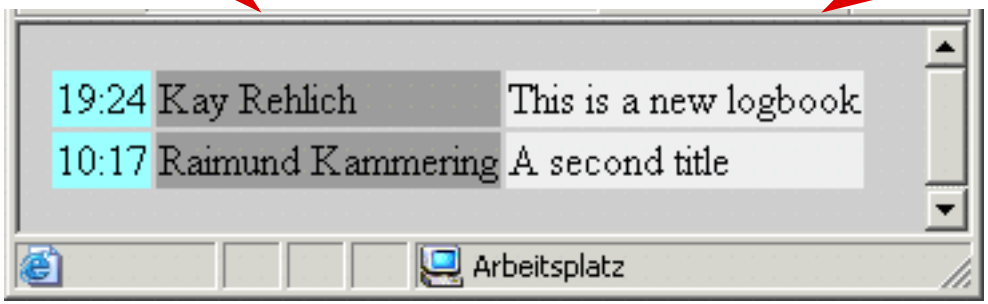
- creates a table row
- creates a table cell
- inserts date and time
- calls further templates

XML + XSL = HTML

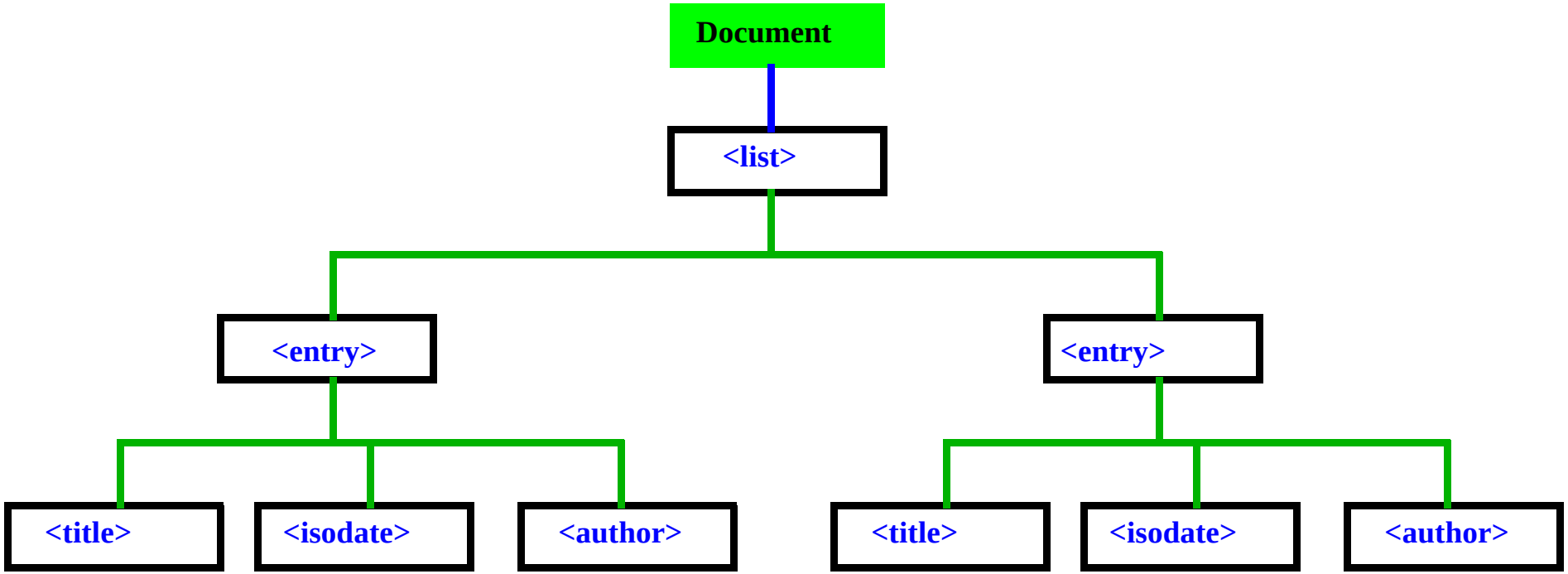
```
<?xml version="1.0"?>
<list>
 <entry>
     <title>This is a new logbook</title>
     <isodate>2003-02-07</isodate>
     <time>19:24</time>
     <author>Kay Rehlich</author>
 </entry>

 <entry>
     <title>A second title</title>
     <isodate>2003-01-09</isodate>
     <time>10:17</time>
     <author>Raimund Kammering</author>
 </entry>

<list>
```

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="list">
    <html>
       <body BGCOLOR="#cccccc">
         <TABLE>
           <xsl:apply-templates select="entry">
                <xsl:sort order="descending" select="isodate"/>
            </xsl:apply-templates>
         </TABLE>
      </body>
    </html>
  </xsl:template>

  <xsl:template match="entry">
    <TR>
       <TD BGCOLOR="#99ffff">
                <xsl:value-of select="date"/>
                <xsl:value-of select="substring(time,1,5)"/>
                <xsl:apply-templates select="author"/>
                <xsl:apply-templates select="title"/>
      </TD>
    </TR>
  </xsl:template>
...
</xsl:stylesheet>
```

XML + **XSL**

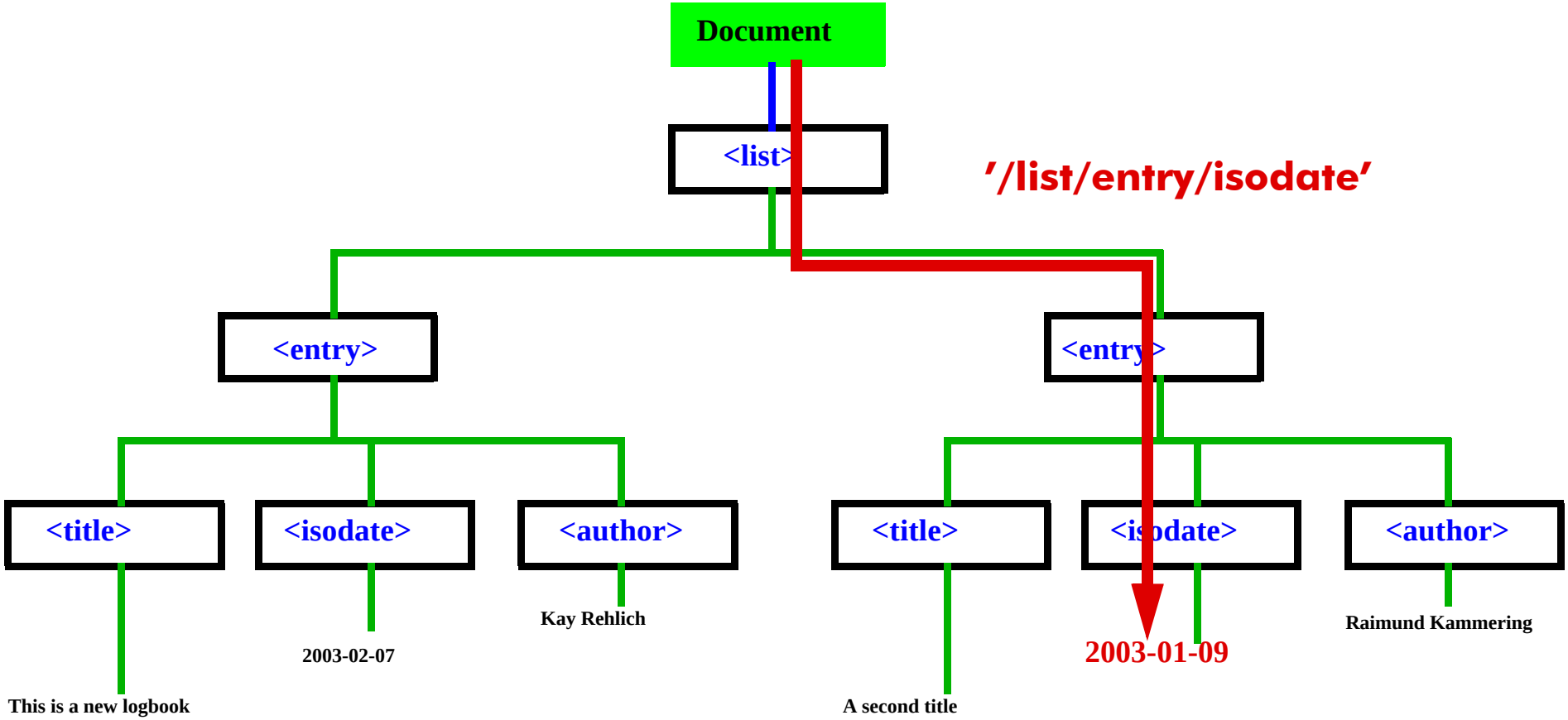


DOM: Document Object Model



- dynamically access and update the content, structure and style of documents
- provides a standard set of objects for representing HTML and XML documents
- platform- and language-neutral interface

XPath: to access the data in a DOM tree

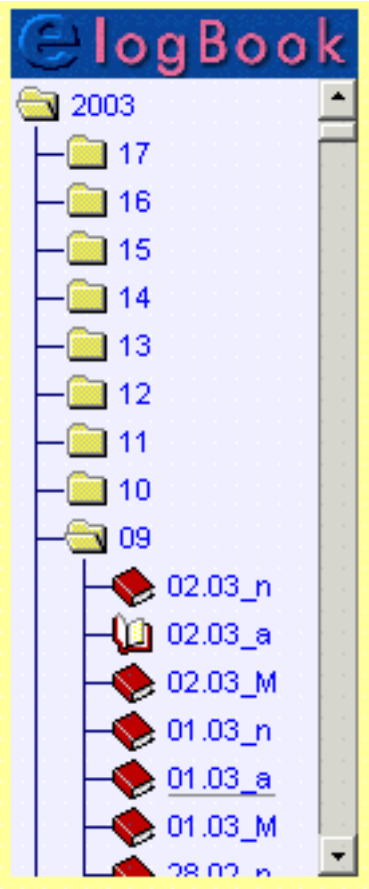


XPath:

- to navigate in DOM trees
- complex pattern expressions (e.g.: doc.SelectNodes('/list/entry[isodate=2003-01-09]'))
- functions (e.g. n = doc.count('/list/entry/*'))

Applet: Client JAVA Code

```
<td width="300" valign="top" align="left">
<applet code="tree.class" codebase="/applets/tree"
archive="tree.jar" width="130" height="250" >
        <param name="background" value="#eeeeff">
        <param name="textcolour" value="#0000ff">
        <param name="conncolour" value="#000077">
        <param name="linkcolour" value="#888888">
        <param name="SORT" value="DESCENDING">
        <param name="MATCH" value="EXACT">
        <param name="expanded" value="FIRST">
        <param name="ICON0" value="Folder95C.gif">
        <param name="ICON1" value="Folder95O.gif">
        <param name="ICON2" value="BookC.gif">
   <param name="ICON3" value="BookO.gif">
   <param name="NODEFILE"
     value="/applets/tree/ttflogbook.txt">
</applet>
</td>
```



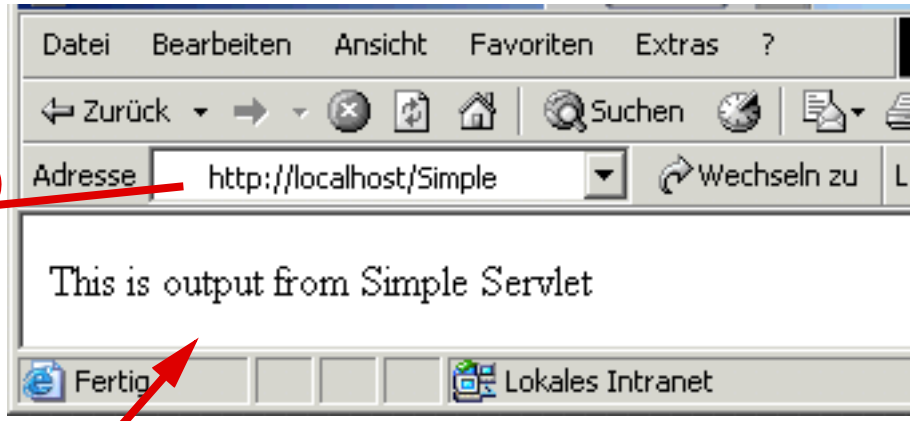
JAVA Applet:

- executes JAVA code on the client
- can display within a e.g. table or in a separate window

Servlet: Server JAVA Code

```
/*@(#)SimpleServlet.java*/
import javax.servlet.*;
import javax.servlet.http.*;

public class Simple extends HttpServlet
{

  public void doGet (
    HttpServletRequestrequest,
    HttpServletResponseresponse
     ) {
        PrintWriter out = response.getWriter();
        session_ = request.getSession(true);

        response.setContentType("text/html");

        out.println("<HTML><BODY>");
        out.println("<H1>This is output from Simple Servlet</H1>" );
        out.println("</BODY></HTML>");
        out.close();
     }
  // public void doPost (...)
}
```

calls Simple.doGet()



Servlet:

- JAVA code runs on the Web server
- independent of client browser version
- has a session object (user session)
- eLogBook: read XML files and do XSL transformation

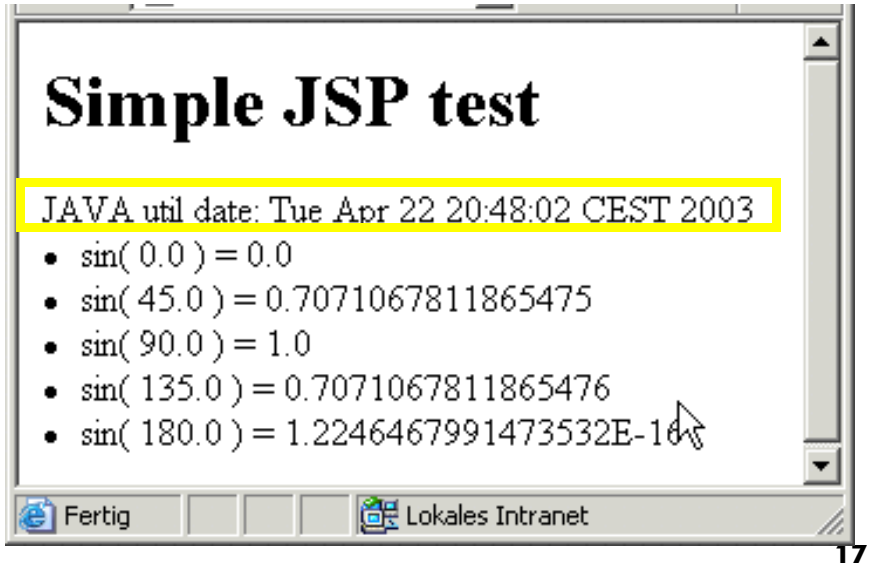
JSP (JAVA Server Page) Example

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">

<%@ page info="a JSP example" %>

<HTML> <BODY>
<h1>Simple JSP test</h1>

JAVA util date: <%= new java.util.Date() %>
<%
   for (double i=0.0; i<=180.0; i=i+45.0) {
%>
   <li>sin( <%=i %> ) = <%=Math.sin(i*Math.PI/180.0) %></li>
<%
   }
%>

</BODY>
</HTML>
```

is a JSP page

insert the local value into the response

mix JAVA and HTML



TagLibs: Common Functions in JSP

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">

<%@ page info="Template logbook" %>
<%@ taglib uri="http://java.sun.com/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jstl/xml"  prefix="x" %>

<html>

<c:import url="conf.xml" var="conf_xml" />

<x:parse xml="${conf_xml}" var="conf" />

   <head>
      <title><x:out select="$conf/logbook/name" /></title>
   </head>
   <body>
      <a href="http://<x:out select="$conf/logbook/host"/><x:out
select="$conf/logbook/rootdir"/>/logo.html">A Link</a>
   </body>
</html>
```

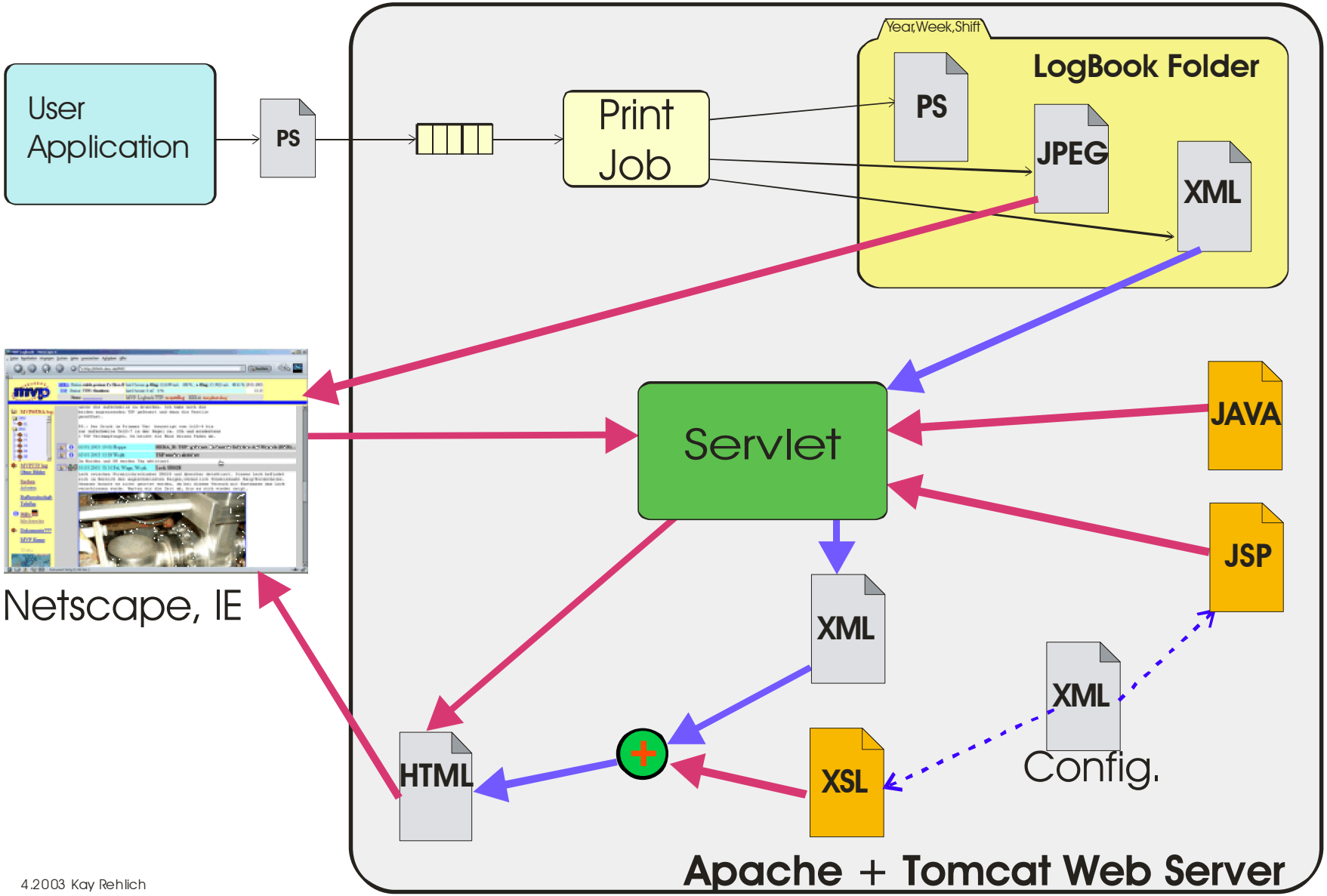
two TAGlibs

parse the 'conf.xml' file

insert the XPath expression

Putting all together: eLogBook Architecture

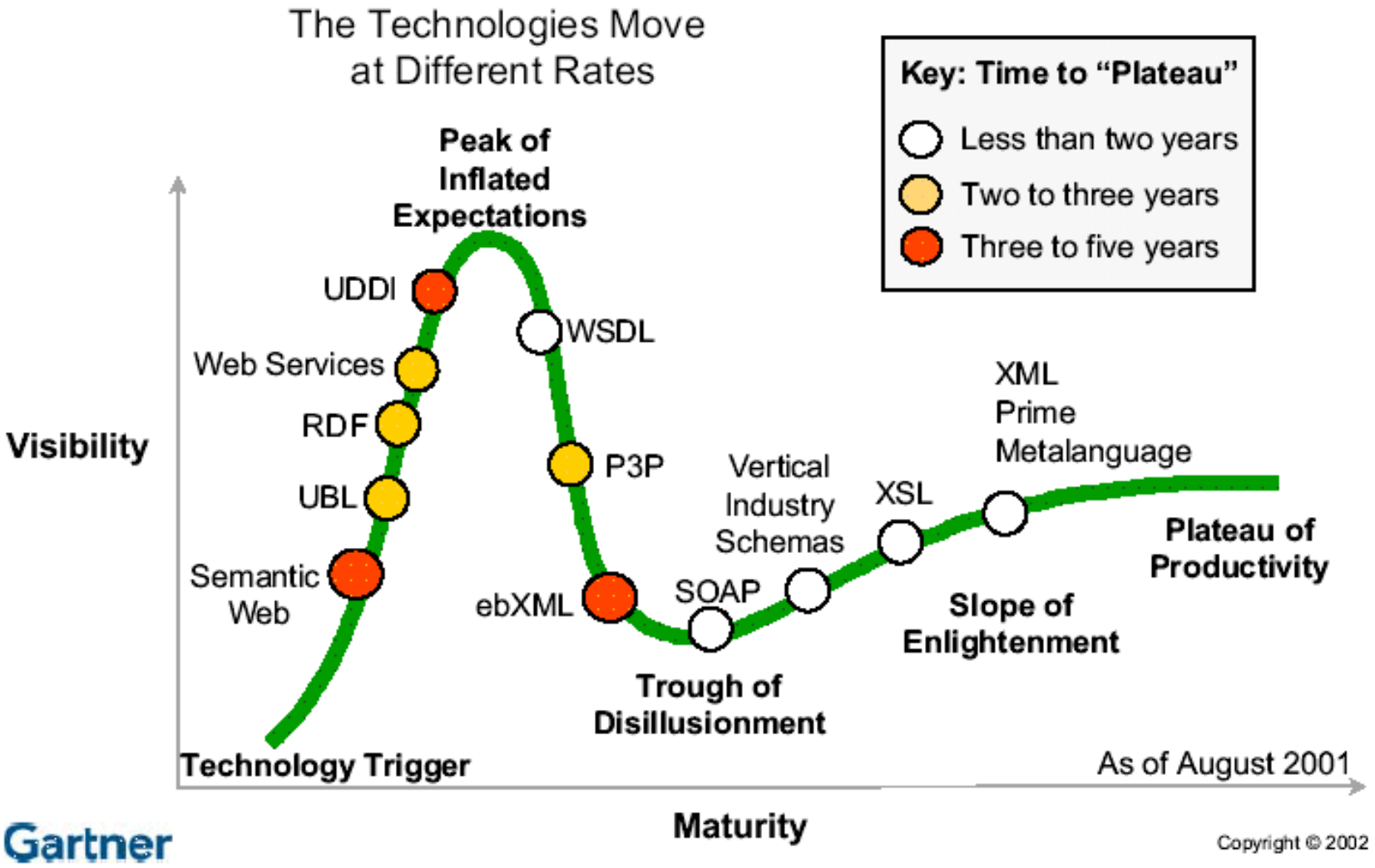
eLogBook Data Flow



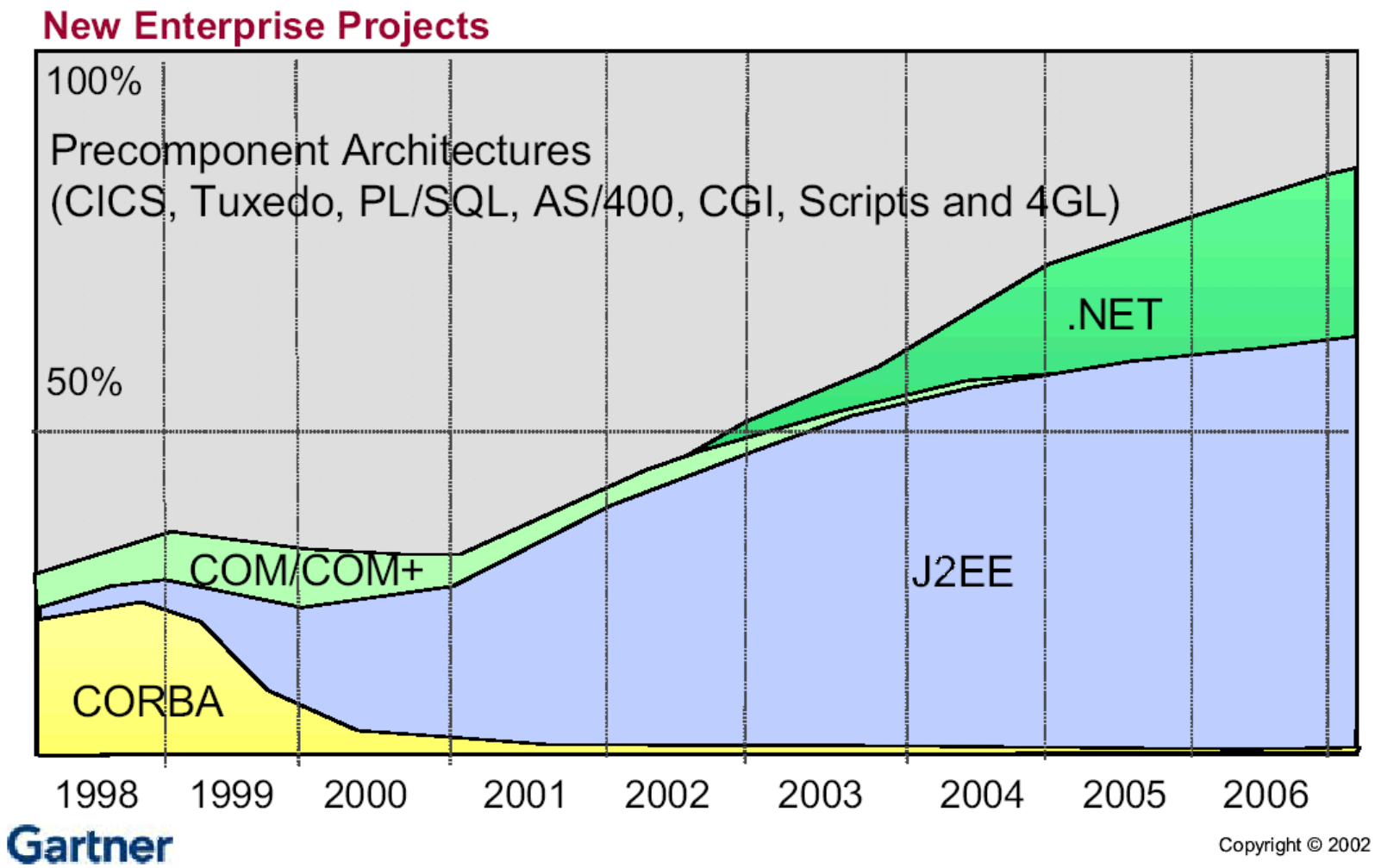
More Web Technologies

- SAX (Simple Api for XML): event driven parser for XML
- DTD (Document Type Definition): to validate documents
- SOAP (Simple Object Access Protocol): XML RPC communication
- WSDL (Web Service Description Language)
  UDDI (Universal Description, Discovery, and Integration)
- JMS (JAVA Messaging Service)
- SVG (Simple Vector Graphics)
- ...

The XML Hype Cycle: 4Q02



Enterprise Projects Adopt New Architectures



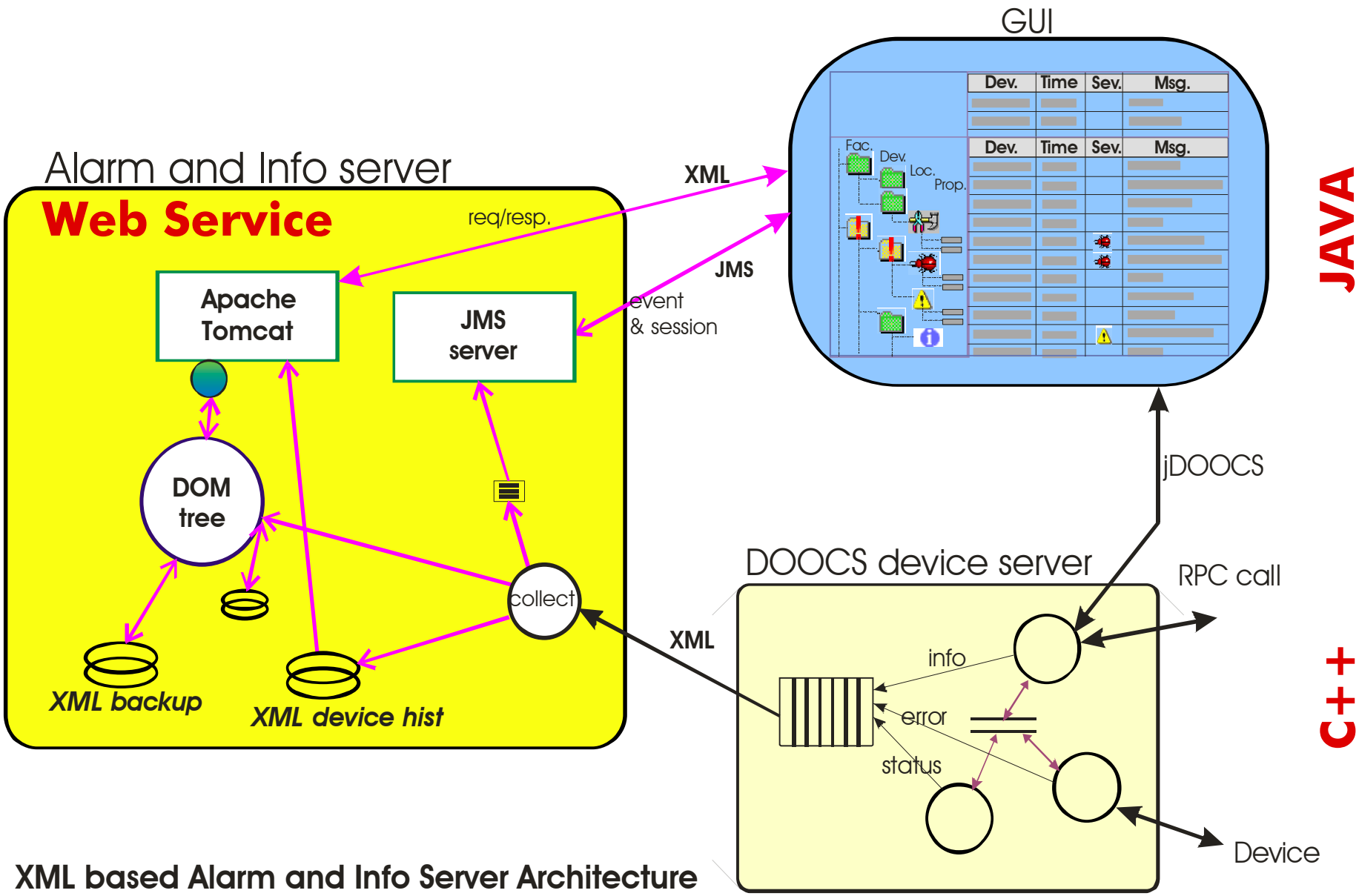
Web Technologies in Control Systems? Outlook

- JAVA: write once - run everywhere (biggest server .. mobile phone)
- JAVA: needs a lot of memory ==> slow
- XML: THE standard of info exchange (configuration, docu ..)
- XML: many tools available
- Web browser: every body knows how to use it
- Web development still very complex
- SVG too slow for graphic display, not available on all browsers

➡ Integrate technologies (e.g. XML, JAVA)
and bring Web and control systems together

➡ Web will not replace Control Systems

Outlook: Alarm and Info Server



XML based Alarm and Info Server Architecture

<more_info>tesla.desy.de/doocs</more_info>

<this>tesla.desy.de/doocs/elogbook/talk0305.pdf</this>

<the_end>Thank you!</the_end>