

Webtechnologien im Überblick

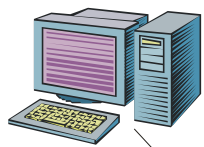
Wassili Kazakos (FZI)

- Fragestellung: „Wie bekomme ich meine Inhalte ins Web?“
- Teil 1: Der klassische Weg
 - ▶ Grundgedanken / Historische Entwicklung
 - ▶ Techniken
 - Client- und Server-seitige Verarbeitung
 - HTML, HTTP, JSP
- Teil 2: Der XML Weg
 - ▶ Grundgedanken / Einführung in XML
 - ▶ Techniken
 - Das XML-Verarbeitungsmodell
 - XML, SOAP, XSLT

- Fragestellung: „Wie bekomme ich meine Inhalte ins Web?“
- Zwei weitere Vorlesungen im Block
 - ▶ Verarbeitung von XML Daten
 - XPath
 - XSLT
 - ▶ Datenbanken und XML
 - XML-Anbindung an Oracle, SQLServer
 - XML-Anfragesprachen

1988 HTML, HTTP, Java, Java Script, Active X, ODBC, JDBC,

1998 XML, XPath, XSLT, SOAP, XML/DB, WebServices



Applet



WAP



HTTP

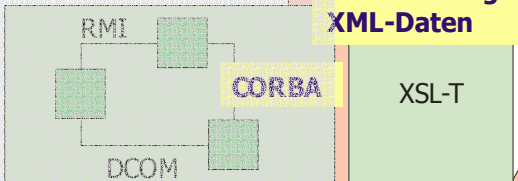
Web-Technologien im Überblick

www.klick-and-bau.com

CGI JSP ASP Servlet XSL-FO

Komponentenframeworks

Verarbeitung von XML-Daten



SQL

Verteilte Transaktionen

Datenaustausch und -zugriff mit XML

Architekturen und Systeme zur Informationsintegration

JDBC

CICS

XML

XML Schema
XQuery

OEM

OEM

XML Schema
XQuery

Semantische Integration

RDBMS

HOST

RDBMS

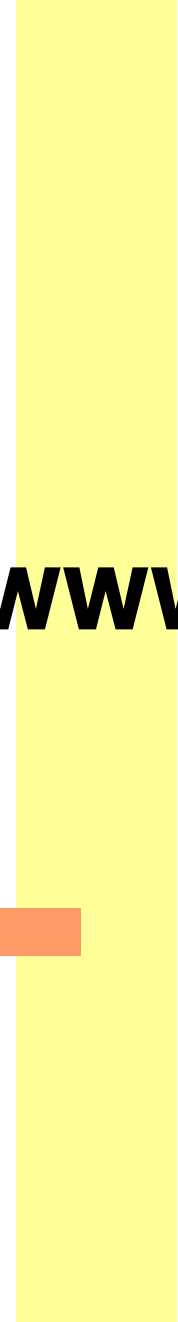
DBMS

RDBMS

OODBMS

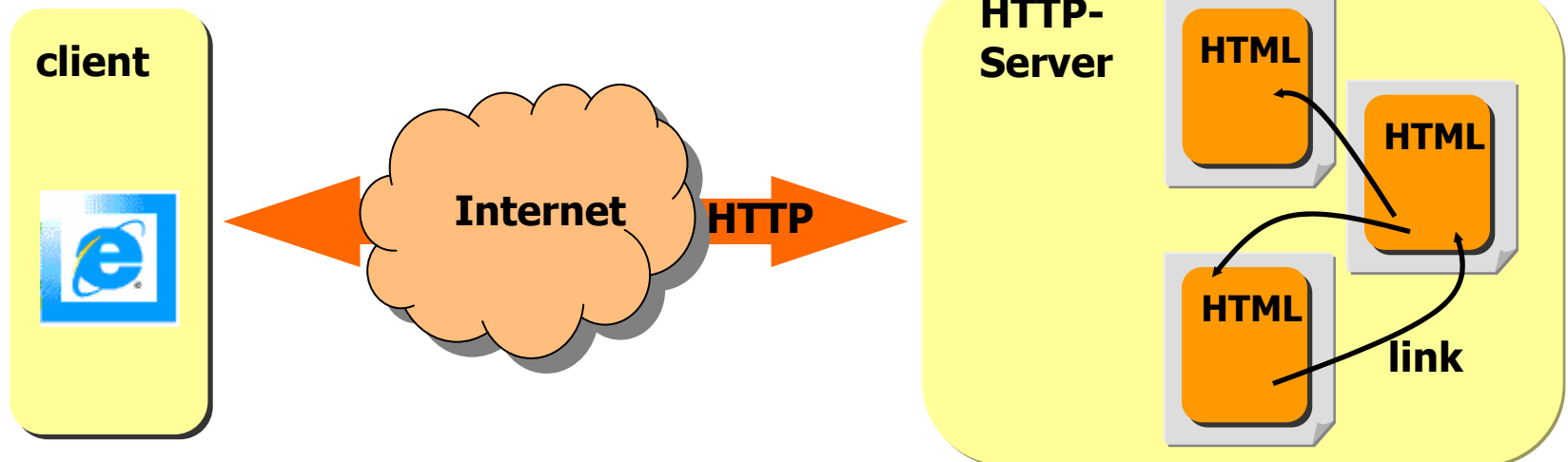
XML-DBMS

Technischer Aufbau des WWW

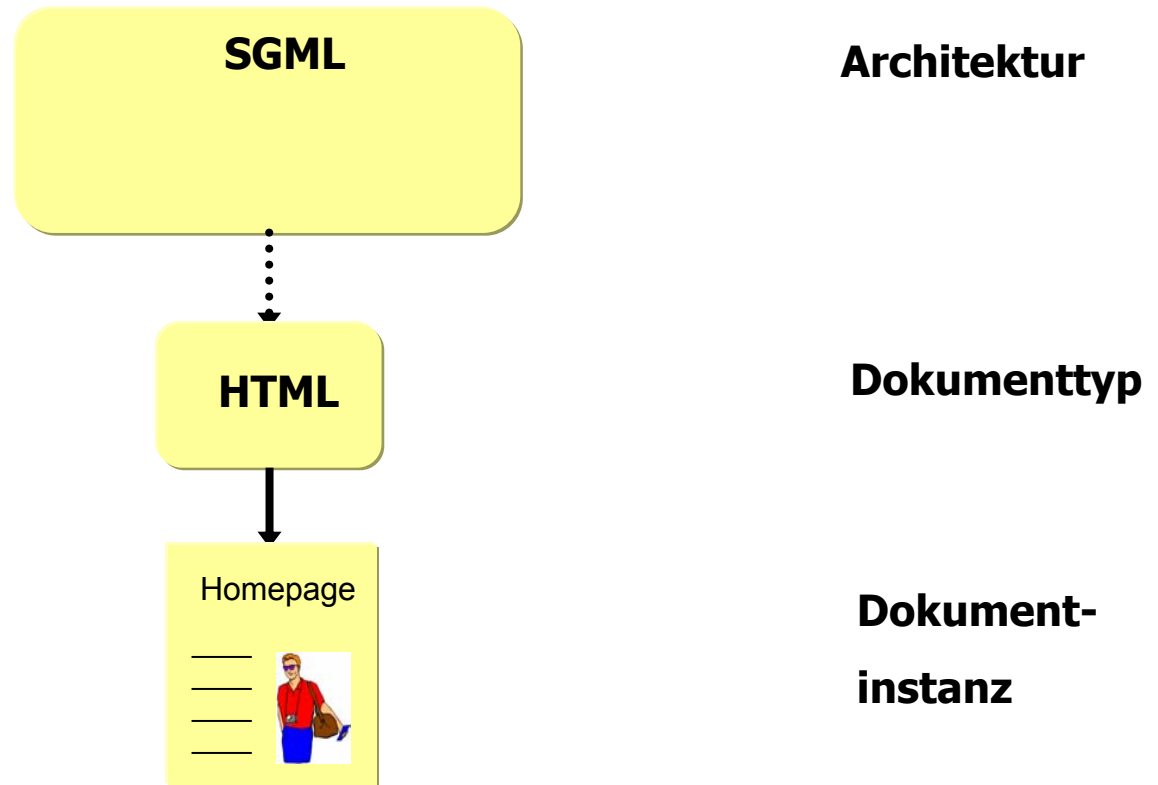


Technischer Aufbau des WWW

- Internet-basierte Client/Server-Architektur
 - ▶ Browser (Client) zur graphischen Darstellung
 - ▶ HTTP-Server zur Übertragung der HTML-Seiten
- HTTP: Protokoll zur Übertragung der Seiten
- HTML: Sprache zur Beschreibung der Seiten
- URL: für unidirektionale Links



Einordnung von SGML und HTML



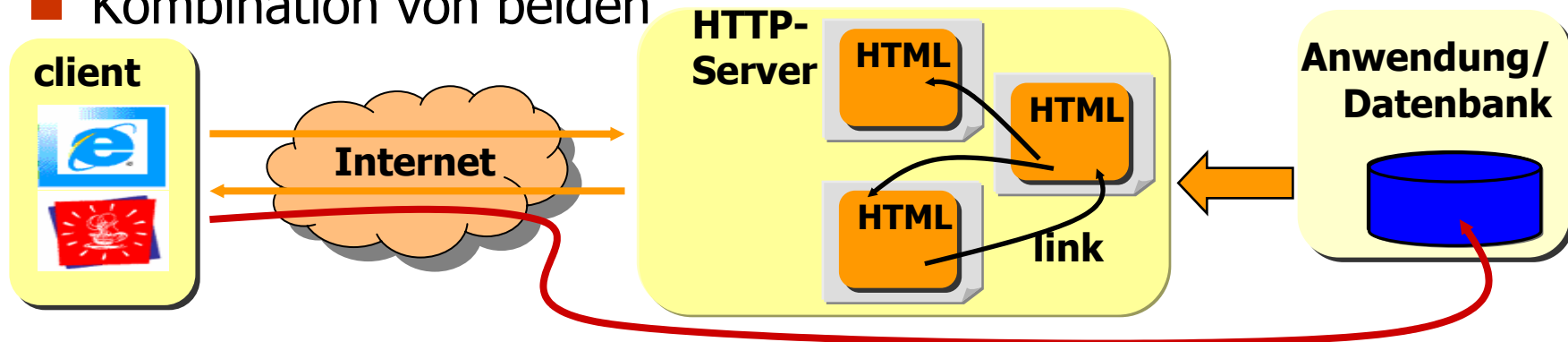
HTTP: Das Protokoll des WWW



- HTTP - HyperText Transfer Protocol
 - ▶ Dient der Übertragung von Hypermedia-Dokumenten zwischen einem Server und einem Client
 - ▶ Anwendungsschicht-Protokoll für statische Informationen
 - get: Anfordern einer Datei vom Server
 - put: Ablegen einer Datei auf dem Server
 - post: Übermitteln von Parametern & Anforderung einer Datei
 - ▶ Zustandslos

Zugriff auf Inhalte über WWW

- Feststellung
 - ▶ Inhalte liegen selten als HTML vor
 - ▶ Inhalte werden in Datenbanken vorgehalten und/oder durch Anwendungen generiert
- Ansätze zur Darstellung der Inhalte (zwei Extreme)
 - ▶ Server-seitig wird HTML generiert
 - ▶ Client-seitige wird auf die Inhalte zugegriffen (clients/server)
- Kombination von beiden



Server-seitige Ansätze

Der „klassische“ Weg



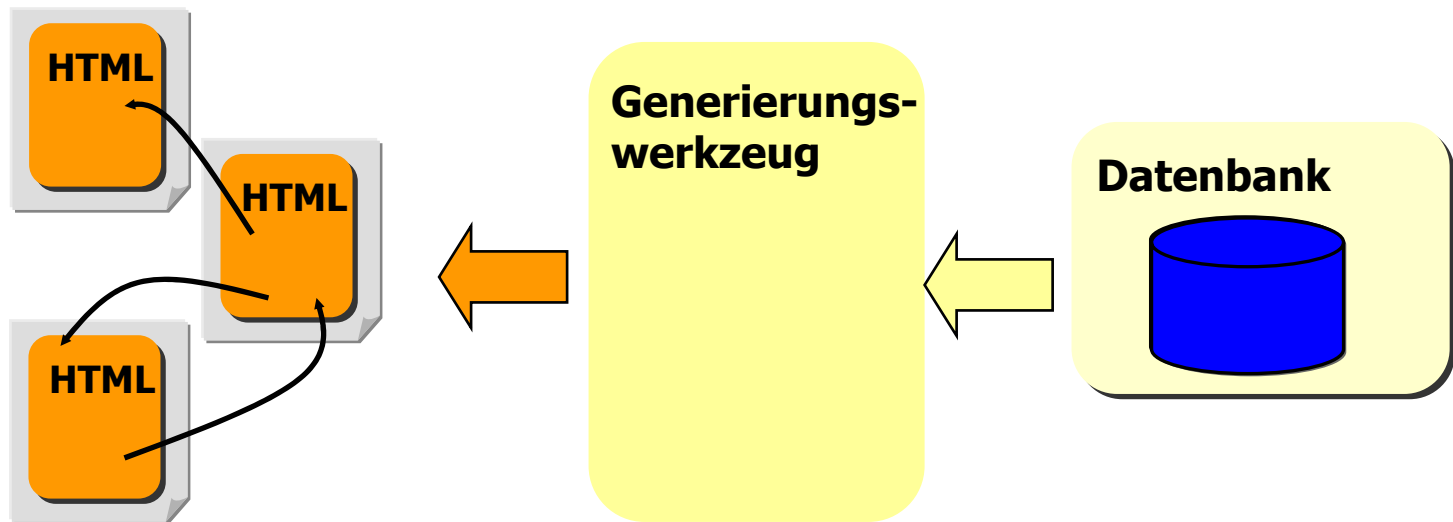
Serverseitige Ansätze - Übersicht



- **Generierung statischer HTML-Seiten**
- **Common Gateway Interface CGI und CGI-Skripte**
- Schema Basierte Generierung dynamischer Seiten
- **Web Server APIs**
- Server Side Includes (SSI)
- **Server-seitige Skripte**
 - ▶ Beispiel: Java Server Pages (JSP)

Generierung statischer HTML-Seiten

- Periodische Extraktion von Daten aus einem DBMS, Vorbereitung von statischen HTML Seiten
- Navigation mittels off-line generierter Links
- Deskriptiver Zugriff mittels Suchmaschinen und Stringvergleich auf den Seiten



■ Vorteile

- ▶ Kein Overhead bei Nutzung: nur HTML-Seiten
- ▶ Keine DB-Zugriffe
- ▶ Automatische Erfassung durch Suchmaschinen

■ Nachteile

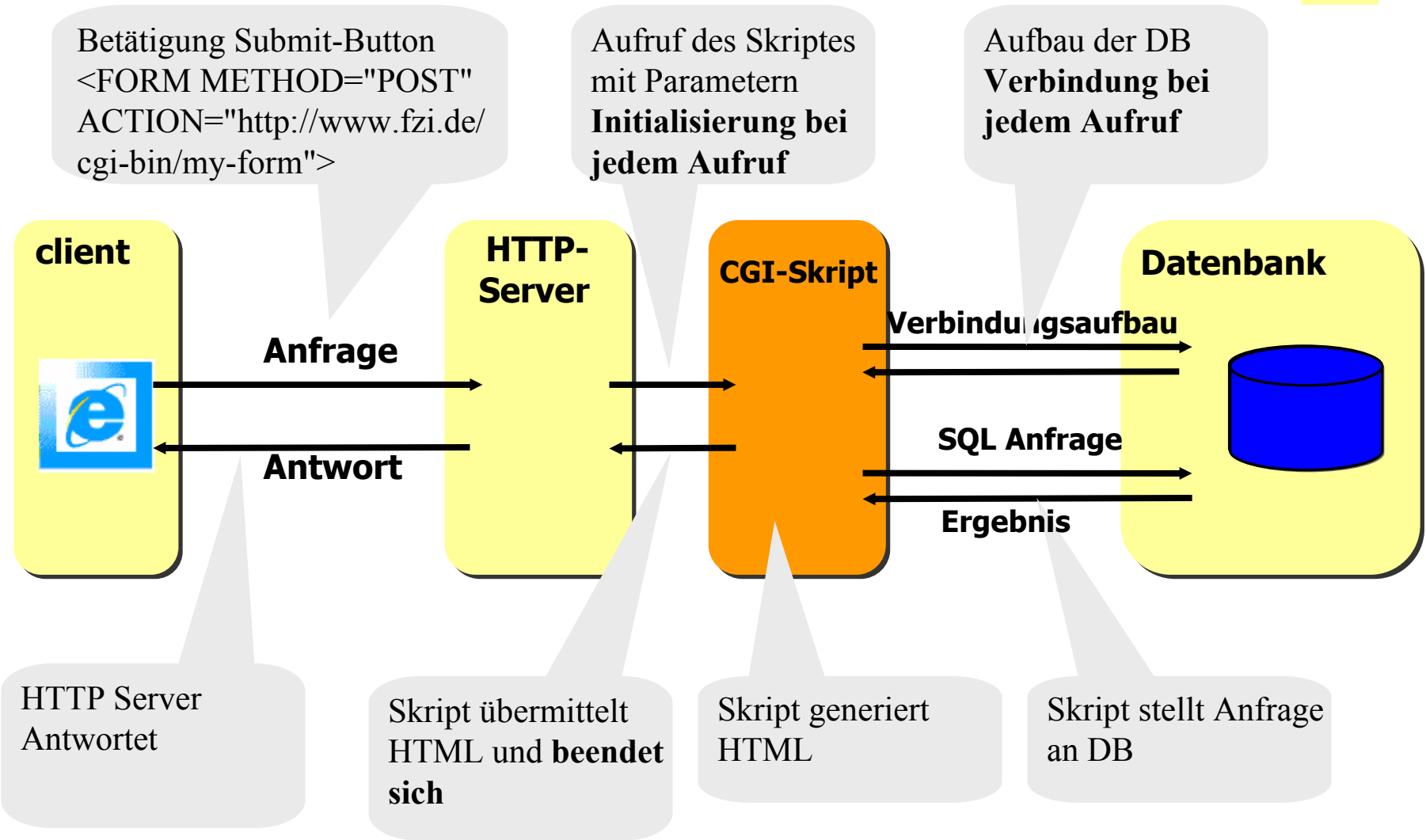
- ▶ Im Vergleich zu Datenbanksystemen begrenzte Suchfunktionalität
- ▶ Aktualisierungs- und Konsistenzproblem
- ▶ Keine Anwendungsfunktionalität

■ Einsatz interessant bei sich selten ändernden Daten und vielen Zugriffen

Common Gateway Interface - CGI

- Definierte Schnittstelle für den Zugriff auf Server-seitige Anwendungen
- Ermöglicht HTTP-Servern den Zugriff auf externe, lokale Anwendungen
- Beispiel:
 - ▶ <http://www.klick-and-bau.com/cgi/suche?term=tisch&preis=1>
 - ▶ Aufruf des Programms "suche" mit den parametern "term" und "preis"
- Implementiert in beliebiger Programmier- oder Skriptsprache (CGI-Skripte)
- Datenbankintegration durch entsprechende Anwendungen

CGI-Skripte



■ Vorteile

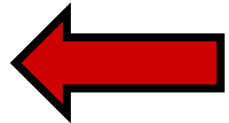
- ▶ Beliebige Programme können integriert werden
- ▶ Sicherheit durch eigenen Prozess
- ▶ Volle Suchfunktionalität des darunterliegenden DBMS

■ Nachteile

- ▶ Ein Prozess pro Anfrage
- ▶ Keine Speicherung des Zustands
- ▶ Für jede DB-Anfrage Verbindung aufbauen und trennen
- ▶ Keine Trennung von Präsentation und Anwendungslogik

■ CGI als Schnittstelle immer noch geeignet und genutzt

■ CGI-Skripte nur für kleine Anwendungen geeignet



API-basierte Ansätze

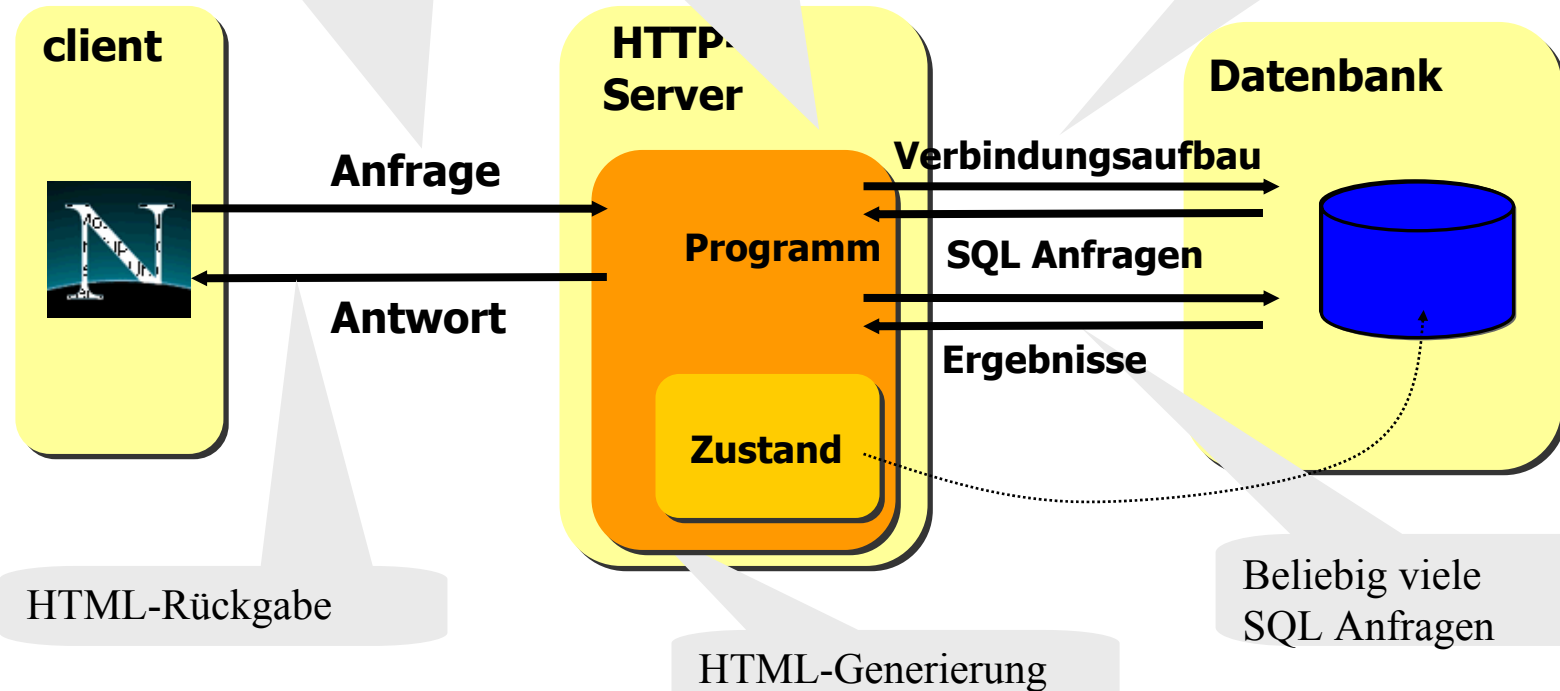
- Entwickelt um Nachteile der CGI-Skripte zu überwinden
 - ▶ Die CGI-Schnittstelle bleibt erhalten
- Erweiterungen werden in den Adressraum des Servers geladen
 - ▶ Müssen nur einmal geladen werden
 - ▶ Werden in Threads statt Prozessen ausgeführt
- Bekanntesten Vertreter
 - ▶ NSAPI (Netscape)
 - ▶ ISAPI (Microsoft)
 - ▶ Java Servlets (Sun)

API-basierte Ansätze

Betätigung Submit-Button
<FORM METHOD="POST"
ACTION="http://www.fzi.de/
cgi-bin/my-form">

Aufruf des Programms,
Parameterübergabe
**Initialisierung bei ersten
Aufruf (einmalig)**

Verbindungsaufbau zur
DB beim ersten Aufruf
(einmalig)

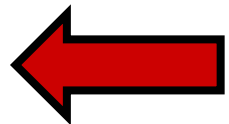


■ Vorteile

- ▶ Höhere Leistungsfähigkeit
 - Session-Verwaltung
 - Zustände, z.B. DB-Verbindung
- ▶ Weniger Ressourcenverbrauch

■ Nachteile

- ▶ Keine Standardisierung
- ▶ Keine Trennung von Präsentation und Anwendungslogik

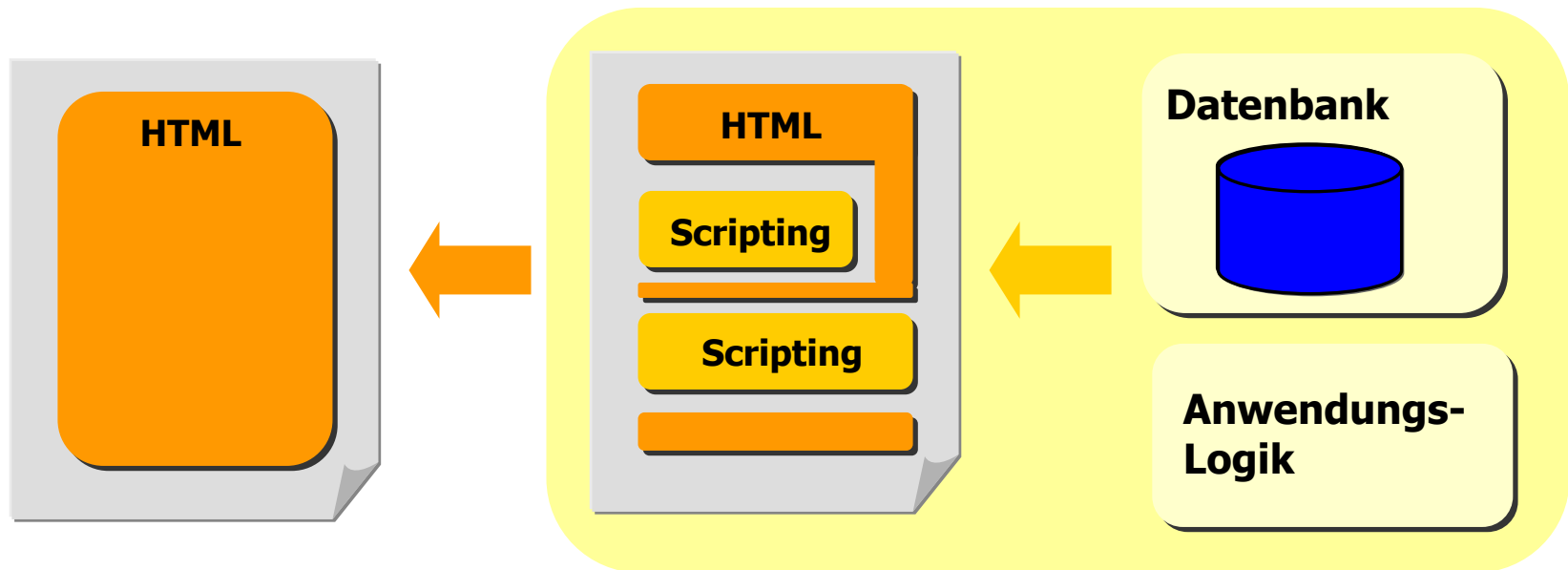


Server Side Includes

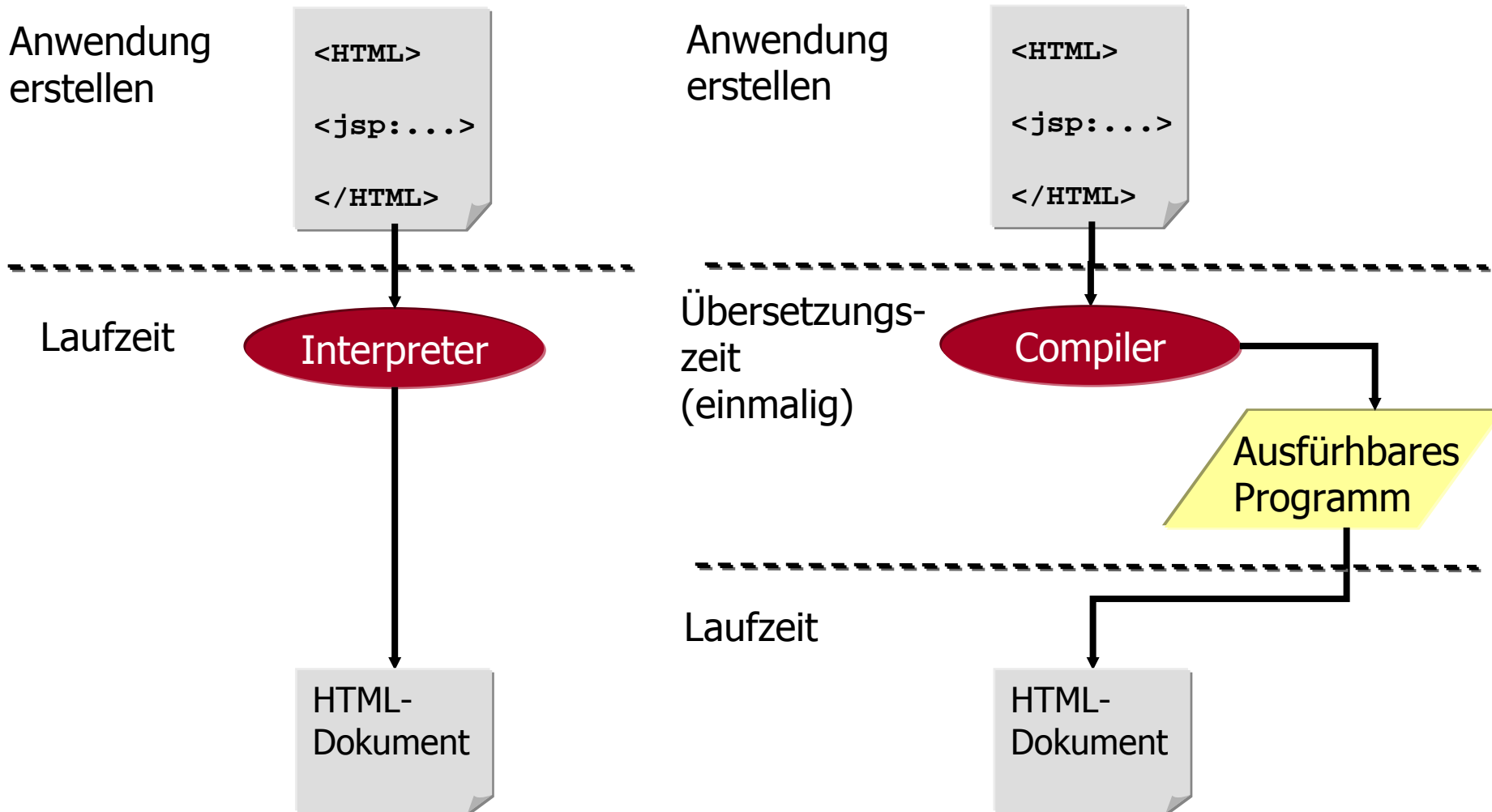
- Statt HTML-Generierung Erweiterung von HTML um Programmfragmente
- Bei Aufruf der HTML-Seite werden Server-seitig die Programmfragmente ausgeführt
 - ▶ Z.B. Zeitstempel, Versionsnummer etc.
- Vorteile
 - ▶ Erster Ansatz, Design von Anwendungslogik zu trennen
 - ▶ HTML Designer kann Scripting einfügen
- Nachteile
 - ▶ WebServer-abhängige proprietäre Erweiterung von HTML
 - ▶ Befehlssatz reicht für komplexere Anwendungen nicht
 - keine vollwertige Programmiersprache

Server-Side Skripting

- In HTML-Seiten werden zusätzliche HTML-generierende Quellen integriert
- Beliebig viele HTML-Quellen können aufgenommen werden

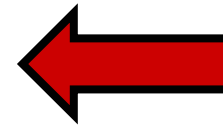


Ausführungsvarianten



Typische Vertreter

- PHP - Personal Home Page
- ASP - Active Server Pages (Microsoft IIS)
- Server Side JavaScript (Netscape)
- JSP - Java Server Pages



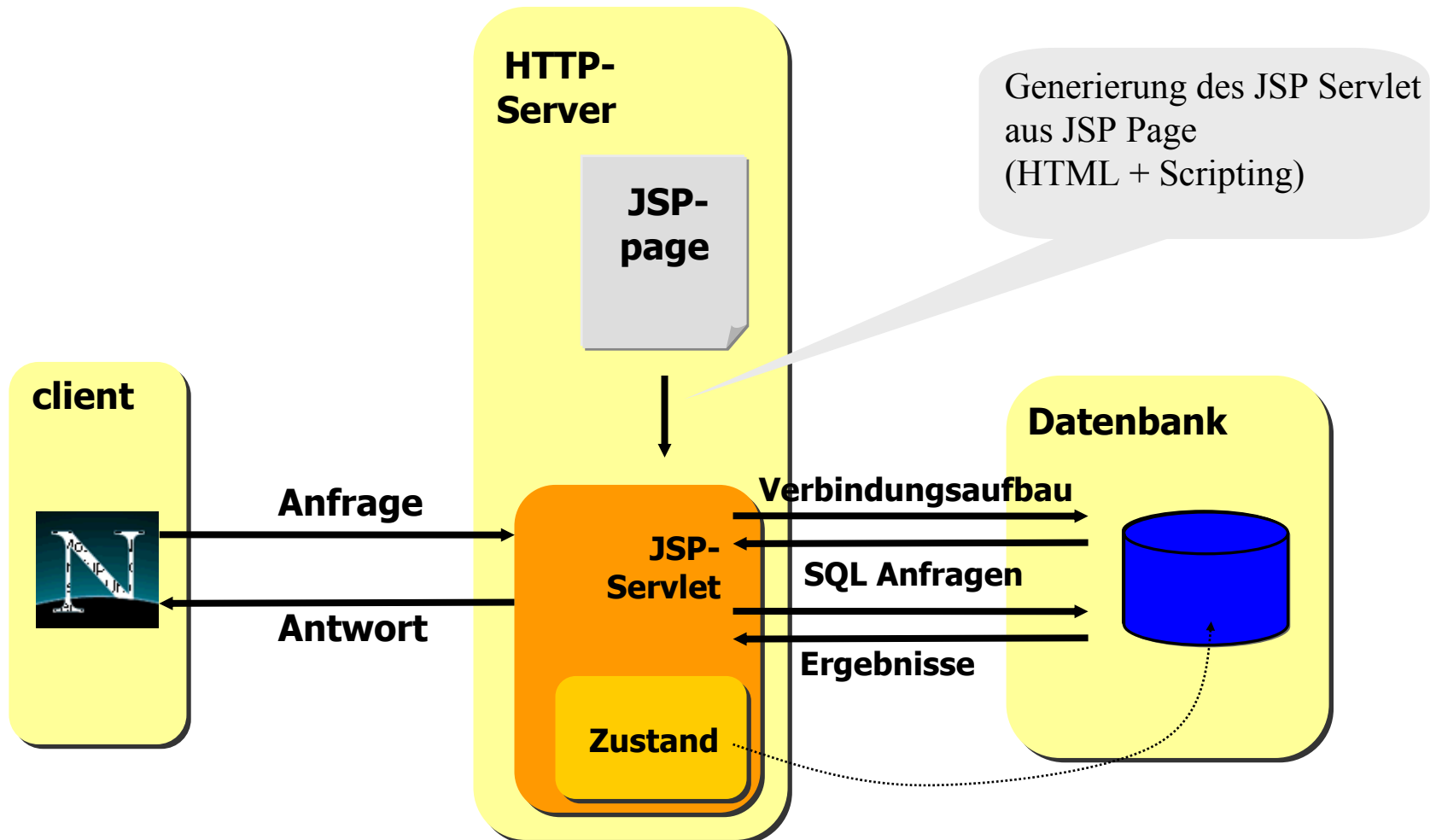
Server-seitige Ansätze: JSP Java Server Pages



Java Server Pages (JSP)

- Bestandteil der Java 2 Plattform Enterprise Edition
- Trennung von (HTML-)Präsentation und Inhalt
- Zugriff auf weitere HTML-Quellen über Beans und Servlets
- Ablauf
 - ▶ Erstellung einer JSP
 - ▶ Aufruf der JSP durch den Benutzer
 - Überprüfung, ob sich die JSP geändert hat oder ob sie neu ist
 - (Gegebenenfalls Übersetzung der JSP in ein Servlet)
 - Ausführung des Servlets
 - ▶ Rückgabe des Ergebnisses

JSP Ablauf



JSP Elementtypen



- JSP-Direktiven
- JSP-Skriptelemente
- JSP-Aktionen

- Nachrichten/Anweisungen an den HTTP-Server
- Keine Ausgabe
- Syntax: `<%@ ... %>`
- include-Direktive
 - ▶ Einfügen von Texten vor der Übersetzung
 - ▶ `<%@include file="Pfadangabe"%>`
 - ▶ Beispiel `<%@ include file="copyright.txt" %>`
- page-Direktive
 - ▶ Steuerung der Übersetzung
 - ▶ `<%@page attrib1="txt" attrib2="txt" ... %>`
 - ▶ Beispiel `<%@ page errorPage="/error.html" %>`
- taglib-Direktive
 - ▶ Zur Erzeugung benutzerspezifischer Tags

JSP-Skriptelemente (1)

■ Vereinbarungen / Deklarationen

- ▶ Deklarationen von Variablen, Methoden und inneren Klassen
- ▶ Syntax: `<%! Vereinbarung(en) %>`
- ▶ `import` über die Direktive
- ▶ Beispiel:
 - Instanzvariablen: `<%! int i=0; float f;%>`
 - Methode: `<%! public String Zeit() {...} %>`

■ Anweisungsfragemente/Scriptlet

- ▶ Einbettung von Java-Fragmenten
- ▶ Syntax: `<% Anweisungsfragment(e) %>`
- ▶ Lokale Variablendeklaration
- ▶ Beispiel:
 - `<% int i=1; while (i<10) {out.println(i);i++;} %>`

JSP-Skriptelemente (2)

■ Ausdrücke

- ▶ Einfachste Art eines Skripts
- ▶ Syntax: `<%= Ausdruck %>`
- ▶ Umwandlung zur Laufzeit in String (`toString`-Methode)
- ▶ Beispiel: `<%=Uhr.getTime()%>` entspricht
- ▶ `<%out.print(Uhr.getTime());%>`

■ Kommentare

- ▶ Syntax: `<%-- comment --%>`

- Ziel
 - ▶ Erzeugen und Verändern von Objekten
 - ▶ Steuerung der aktuellen Ausführung
- Aktion `include`
 - ▶ Aufruf einer Seite zur Ausführungszeit (auch JSP möglich)
 - ▶ Syntax: `<jsp:include page="Pfadangabe" />`
- Aktion `forward`
 - ▶ Weiterleitung der Anfrage an eine andere Seite
 - ▶ Syntax: `<jsp:forward page="Pfadangabe" />`

- Angenommen, klick-and-bau.com verwaltet Artikelstammdaten in der folgenden Tabelle:

```
ARTIKEL(  
  Id CHAR(12),  
  Name VARCHAR(25),  
  Beschreibung VARCHAR(512),  
  Photo_URL VARCHAR(512),  
  Hersteller_Id CHAR(12)  
  Kategorien VARCHAR(256),  
  Status INT,  
  Preis NUMERIC(8,2)  
)
```

- Aktuelle Produktname und Photo sollen nun über das Web zur Verfügung gestellt werden

JSP: Beispiel

```
<%-- page-Direktive --%>
<%@ page errorPage="error.html"
import="java.sql.*" session="false" %>

<html>
  <body>
    <h1 align="center"> Produktliste vom
      <%-- Ausdruck --%>
      <%=new java.util.Date().toLocaleString()%>

    </h1>

    ...
```

JSP: Beispiel (Forts.)

```
...  
<%-- Datenbank-Verbindung --%>  
<%-- Deklaration der Instanzvariablen --%>  
<%! Connection con %>  
  
<%-- Skriptlet zum Verbindungsaufbau und Abfrage--%>  
<% Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");  
con = DriverManager.getConnection  
    ("jdbc:odbc:ProduktDatenbank", "name", "pwd");  
Statement stmt = con.createStatement();  
ResultSet rs = stmt.executeQuery  
    ("SELECT name, photo_URL FROM artikel");  
%>  
...
```

JSP: Beispiel (Forts.)

```
...
<!-- Ausgabe in zwei Spalten...-->
<table>
<tr>
  <th>Name</th><th>Bild</th>
</tr>
<!-- ...Zeile f,r Zeile-->
<% while (rs.next()) { %>
  <tr><td> <%= rs.getString(1)%> </td>
    <td>  </td>
  </tr>
<%}%>
</table>
<!-- Verbindung schlieflen-->
<% rs.close();%>
</body> </html>
```

- Standardaufgaben sollten auch weiterhin in Komponenten ausgelagert werden
 - ▶ In Java: EJB (siehe Vorlesung 2)
 - ▶ Einbindung über JSP-Aktionen
- Aktion `useBean`
 - ▶ Deklaration der Bean
 - ▶ Syntax: `<jsp:useBean id="varName" class="KlassenName" scope="GültigkeitsBereich" />`
 - ▶ `scope={page | request | session | application}`

- Aktion `getProperty`
 - ▶ Lesen der Eigenschaft einer Komponente
 - ▶ Syntax: `<jsp:getProperty name="varName" property="AttributsName" />`
- Aktion `setProperty`
 - ▶ Schreiben der Eigenschaft einer Komponente
 - ▶ Syntax:
 - `<jsp:setProperty name="varName" property="AttributsName" value="Wert" />`, oder
 - `<jsp:setProperty name="varName" property="AttributsName" param="cgi-parameter" />`

Weiter JSP Feature



- Benutzerdefinierte Tags
- Mehrsprachigkeit (Unicode)
- Sitzungsverfolgung
- Java-APIs stehen voll zur Verfügung

Benutzerdefinierte Tags

- Kapseln von immer wiederkehrender Funktionalität als Tag
- JavaServer Pages Standard Tag Library (JSTL)
- Core Tags: Standardfunktionalität
 - ▶ choose, forEach, if, ...
- XML Tags: Zugriff auf XML-Dokumente
 - ▶ parse, transform, ...
- Internationalisierung
 - ▶ setLocale, requestEncoding, setTimeZone, setBundle, ...
- SQLTags: Standard DB Funktionen
 - ▶ setDataSource, query, update

- Standardisierter Weg zur Trennung von
 - ▶ Gestaltung (Designer)
 - ▶ Anwendungslogik (Entwickler)
- Grundlage für komponentenorientierte HTML-Generierung
 - ▶ Höhere Wiederverwendung der HTML-generierenden Quelle
 - ▶ Trennung von Präsentation und Anwendungslogik

Server-seitiger Ansätze - Dilemma

■ Plus

- ▶ HTML ist unabhängig vom Web-Browser
- ▶ Keine aktiven Inhalte am Client (Sicherheit)
- ▶ HTTP Protokoll

■ Minus

- ▶ Mangelnder Komfort in der Benutzerführung
- ▶ Jede Interaktion erfordert Kommunikation mit dem Server
 - Hoher Kommunikationsaufwand
 - Hohe Belastung des Servers

Client-seitige Ansätze

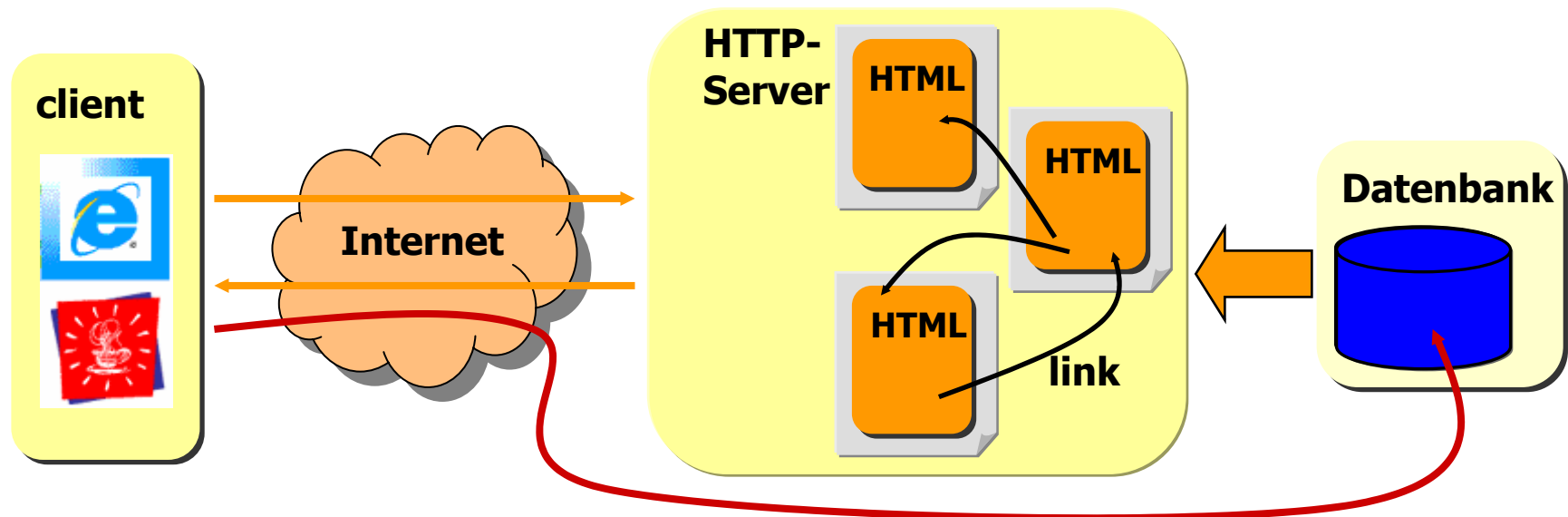


Zur Erinnerung

■ Zwei Extreme:

- ▶ Server-seitig = HTML- Client
- ▶ Client-seitig = Windows-Ähnlicher Client
 - Direkter DB Zugriff sollte vermieden werden

■ Kombination denkbar und sinnvoll



Client-seitige DB-WWW-Integration

- Realität:
 - ▶ Teilverlagerung von Funktionalität auf die Client-Seite
 - ▶ Nicht nur die Extreme setzen sich durch
- Nicht mehr HTML-Generierung
 - ▶ Erweiterung 1: mehr Interaktion/Überprüfung auf dem Client
 - ▶ Erweiterung 2: Windows-Ähnliche Benutzerführung
 - ▶ Erweiterung 3: Unmittelbarer Zugriff auf die Daten (eher abzuraten)

Funktionalität auf der Client-Seite

- Browser-spezifische Erweiterungen: Plug-Ins
 - ▶ Z.B. Macromedia Shockwave/Flash
- Skript-Sprachen
 - ▶ Z.B. JavaScript, Visual Basic Script
- Sprachen
 - ▶ Java
 - ▶ VisualBasic, C++ über ActiveX controls
- Übliche Aufteilung
 - ▶ Benutzerschnittstelle auf dem Client
 - ▶ Datenbankdienste weiterhin auf dem Server
 - Vom Direktzugriff von Client auf die DB ist aus Sicherheitsgründen abzuraten

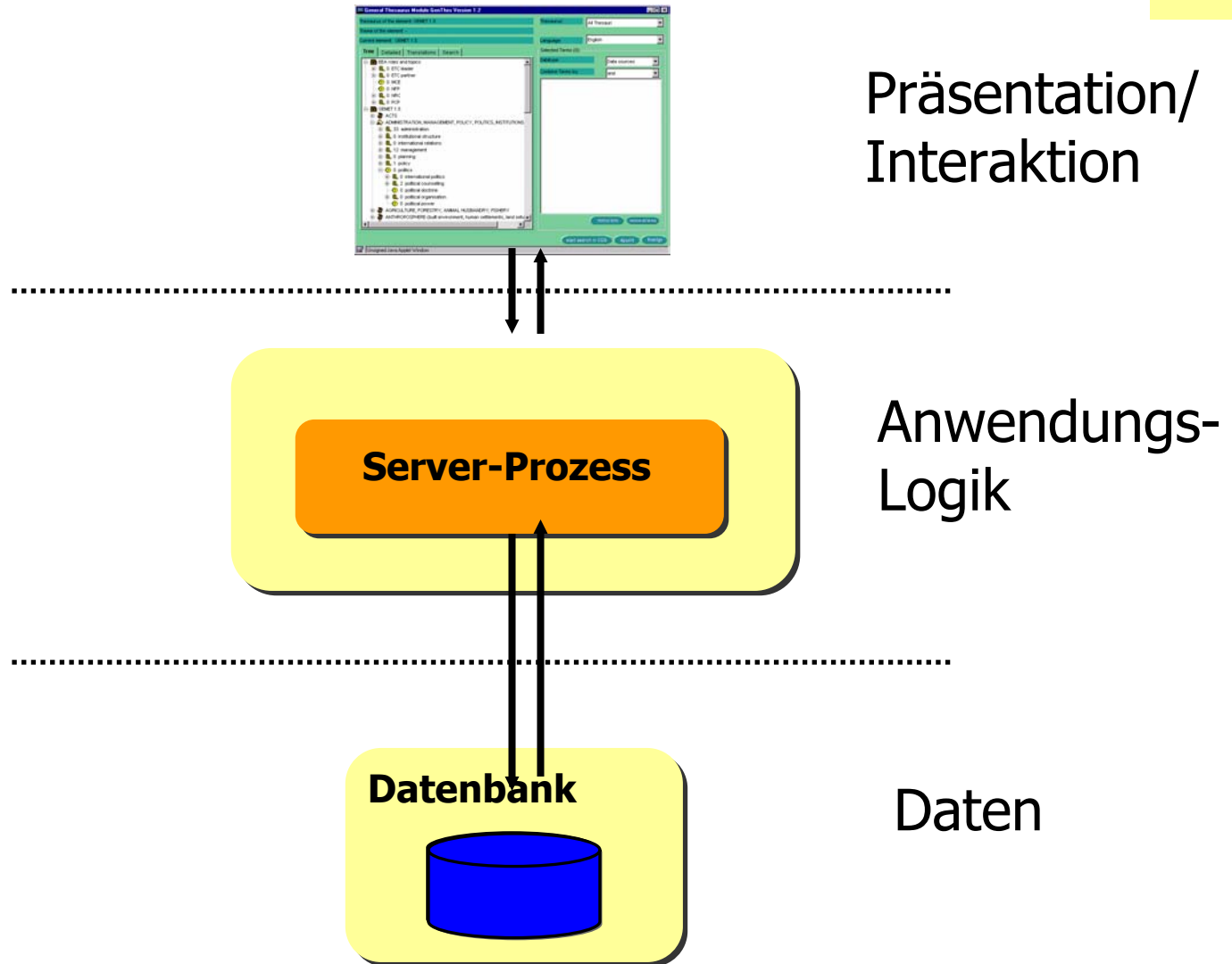
Beispiel: Client-seitige Integration

The screenshot displays the 'General Thesaurus Module GenThes Version 1.2' interface. The window title is 'General Thesaurus Module GenThes Version 1.2'. The interface is divided into several sections:

- Header:** 'Thesaurus of the element: GEMET 1.5', 'Theme of the element: -', and 'Current element: GEMET 1.5'.
- Navigation:** 'Tree', 'Detailed', 'Translations', and 'Search' tabs.
- Tree View:** A hierarchical tree structure showing categories and sub-categories. The 'GEMET 1.5' category is expanded, showing sub-categories like 'ACTS', 'ADMINISTRATION, MANAGEMENT, POLICY, POLITICS, INSTITUTIONS', 'AGRICULTURE, FORESTRY; ANIMAL HUSBANDRY; FISHERY', and 'ANTHROPOSPHERE (built environment, human settlements, land setu)'. The 'ADMINISTRATION, MANAGEMENT, POLICY, POLITICS, INSTITUTIONS' category is further expanded, showing sub-categories like 'administration', 'institutional structure', 'international relations', 'management', 'planning', 'policy', 'politics', 'international politics', 'political counselling', 'political doctrine', 'political organisation', and 'political power'.
- Search Controls:** 'Thesaurus: All Thesauri', 'Language: English', 'Datatype: Data sources', and 'Combine Terms by: and'.
- Buttons:** 'remove term', 'remove all terms', 'start search in CDS', 'quit', and 'help'.

The window is identified as an 'Unsigned Java Applet Window'.

Beispiel: Client-seitige Integration



Resümee „klassische“ Ansätze

- State of the Art:
 - ▶ Server-seitiges Scripting
 - Parameterübergabe über CGI
 - HTML-Generierung auf Serverseite
 - ▶ Bei starker Interaktion und komplexer Benutzerführung
 - Client-seitige Erweiterung (JavaScript, Java)
- Keine „beste Lösung“ für alle Fälle
- Kriterien im Einzelfall entsprechend Anforderungen gewichten
- Immer zu beachten:
 - ▶ rasanter technologischer Fortschritt
 - ▶ rasche Produktentwicklungen

Wie entscheide ich mich?



- Kosten
 - ▶ Freeware, Shareware, kommerzielle Produkte
- Plattformunterstützung
 - ▶ Auf Client- & auf Server-Seite
- Intranet vs. Internet Anbindung
- Funktionsumfang & Design
 - ▶ HTML hat seine Grenzen
- Warten auf den zweiten Teil: Der XML-Ansatz

Der Wandel des Web

1988 HTML, HTTP, Java, Java Script, Active X, ODBC, JDBC,

1998 XML, XPath, XSLT, SOAP, XML/DB, WebServices

Der Wandel des Web

■ Rolle des Web

- ▶ 1988: Weltweiter Austausch von Forschungsergebnissen
- ▶ Heute: Vielfältiger Einsatz zum Austausch, zur Verarbeitung und Visualisierung von Informationen.

■ Stand der (standardisierten) Technik im Web:

- ▶ Visualisierung in HTML oder proprietäre Plugins
- ▶ Austauschformate
 - EDI - Electronic Data Interchange
 - Proprietäre Formate

■ Ziel der nächsten Jahre

- ▶ Die technologische Basis für das Web konsolidieren
- ▶ XML Verarbeitungsmodell verbreiten

HTML-Beispiel

Biotopsbericht - Netscape

File Edit View Go Communicator Help

Biotoptyp - Tabelle (16.07.1998)

Kartierungsart §-24a-Kartierung (Typ1)
Gemeinde 317126 Seebach

Biotopschlüssel	Name	Fläche (ha)	Anzahl
1110	Naturnahe Quelle	0.0516	17
2110	Offene Felsbildung	0.0000	3
2310	Hohlweg	0.0194	1
4110	Feldgehölz	1.0576	9
4120	Feldhecke	0.5635	15
Gesamtsumme Teilbiotypen		1.6921	45
Gesamtsumme Biotypen		1.6921	45

Sonderbedingungen: keine

```
<html>
<head><title>Biotopsbericht</title></head>
<body bgcolor="#FFFFCC" text="#000000">
<h1>Biotoptyp - Tabelle ( 16.07.1998 )</h1>
<table border="0" name="Kritärien" >
  <tr>
    <td><b>Kartierungsart</b></td>
    <td>&sect;-24a-Kartierung (Typ1)</td> </tr></table>
<table border="0" height="214">
  <tr bgcolor="#CCCCFF">
    <td><b>Biotopschl&uuml;ssel</b></td>
    <td><b>Name</b></td>
    <td align="right"><b>Fläche (ha)</b></td>
    <td align="right"><b>Anzahl</b></td></tr>
  <tr>
    <td>1110</td>
    <td>Naturnahe Quelle </td>
    <td align="right">0.0516 </td>
    <td align="right">17 </td></tr>
  <tr border="0">
    <td colspan="2"><b>Sonderbedingungen:</b></td>
    <td colspan="2">keine</td> </tr></table> </body>
```

Struktur

Layout

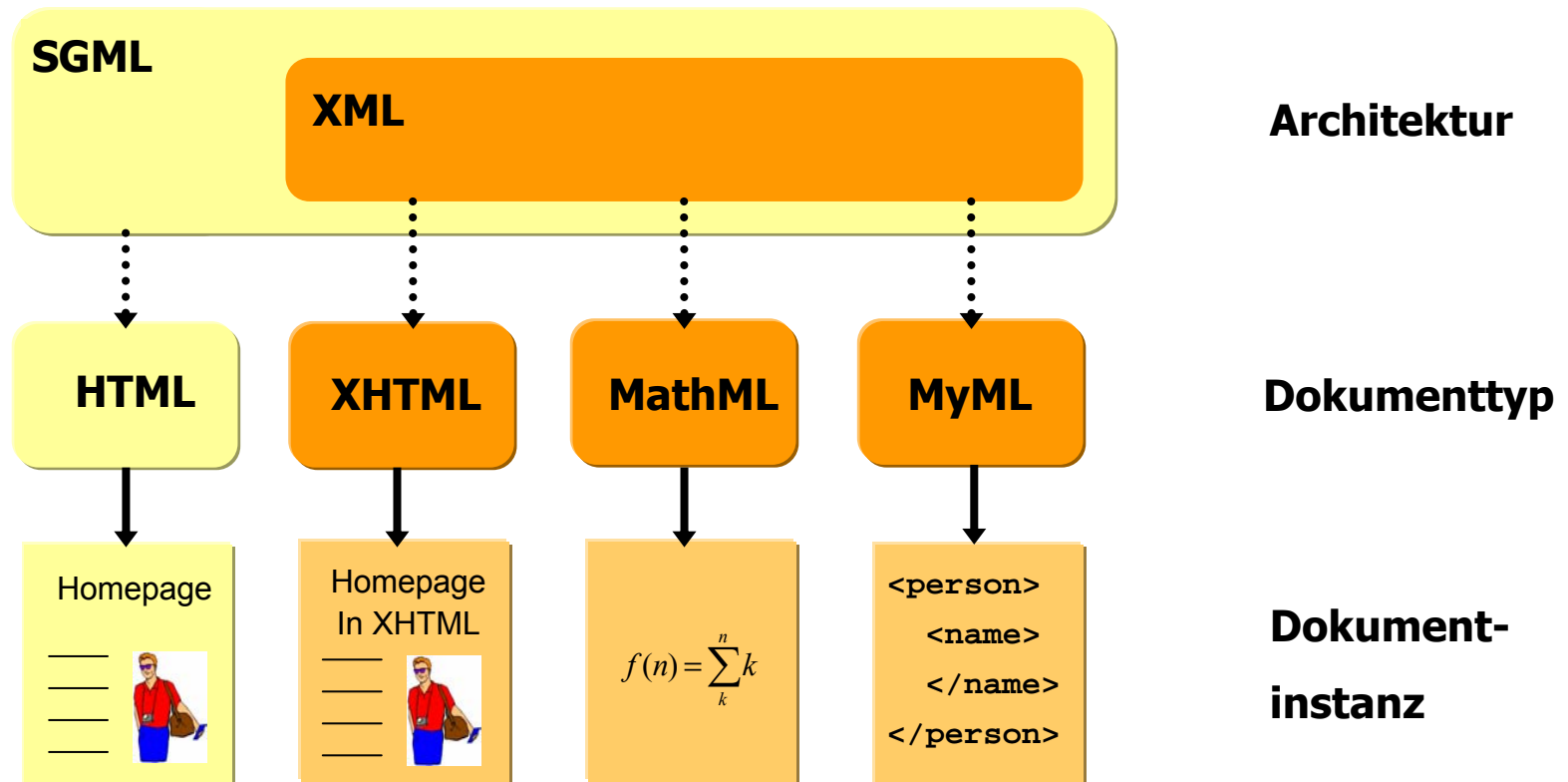
Inhalt

eXtensible Markup Language XML



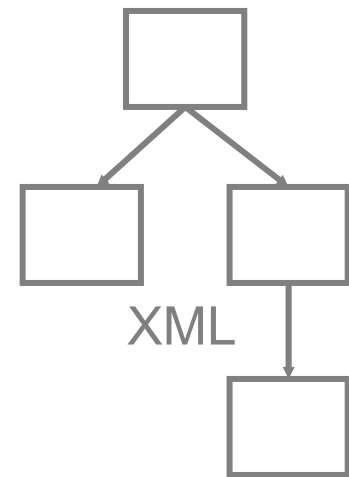
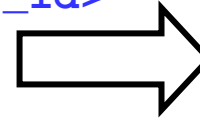
- eXtensible Markup Language (XML)
 - ▶ XML 1.0 ist Recommendation des W3C seit dem 10.02.98
 - ▶ Zielsetzung:
 - Format zu Austausch, Verarbeitung und Darstellung von Information im Internet
 - Einfacher als SGML (Standard Generalized Markup Language)
 - Mächtiger als HTML
 - ▶ Aktueller Stand:
 - Viele XML basierte-Ansätze sind bereits verabschiedet
 - Weitere werden folgen
 - Über XML wird ein neues Verarbeitungsmodell für das Web definiert

Einordnung von XML, HTML und XHTML



XML-Beispiel

```
<warenkorb>
  <artikel>
    <id>xy1234</id>
    <name>Billy Regal</name>
    <beschreibung> Das ist ein Regal
  </beschreibung>
    <photo_url>http://... </photo_url>
    <hersteller_id>R2D2</hersteller_id>
    <preis>98</preis>
  </artikel>
  <artikel>
    <id>xy1234</id>
    <name>Billy Regal</name>
  </artikel>
</warenkorb>
```



Aufbau von XML (Markup)

■ XML-Dokumente bestehen aus Zeichendaten und Auszeichnungen:

▶ XML- und Dokumenttyp Deklarationen

- Verweist auf Grammatik (Dokumenttypdefinition)

```
<?xml version="1.0"?>
```

```
<!DOCTYPE root_element SYSTEM "klick-and-bau.dtd">
```

▶ Elementen (<name attribute="Wert">)

▶ Entity Referenzen (< statt <)

▶ Kommentare (<!-- Das ist ein Kommentar-->)

▶ Processing Instructions

- Werden an die Aufrufende Instanz weitergeleitet
- <?name pidata?>

Aufbau von XML (Gültigkeit)

- Wohlgeformte Dokumente (well-formed documents):
 - ▶ Ein Dokument besitzt geordnete syntaktische Eigenschaften.
- Gültige Dokumente (valid documents):
 - ▶ Ein Dokument ist wohlgeformt und folgt einem a priori gegebenen Schema.
 - ▶ Das Schema wird durch eine Dokumenttyp-Definition (DTD) formalisiert.

Aufbau von XML (Deklaration)

■ Elementtyp-Deklaration

- ▶ `<!ELEMENT br EMPTY>`
- ▶ `<!ELEMENT name (#PCDATA)>`
- ▶ `<!ELEMENT artikel (name,photo_url)>`

■ Attributlisten-Deklaration

- ▶ `<!ATTLIST artikel
artikelID ID #REQUIRED
status (verf,gbar|nicht verf,gbar) "verf,gbar">`

■ Entity-Deklaration

- ▶ `<!ENTITY FZI "Forschungszentrum Informatik">`
- ▶ `<!ENTITY waehrung SYSTEM
"http://www.klick-and-bau.com/waehrung.xml">`

Anwendungen von XML

■ Offizielle W3C DTD

- ▶ Wissenschaft: MathML (Mathematical Markup Language)
- ▶ Graphik, Visualisierung:
 - SVG (Scaleable Vector Graphics)
 - XHTML
- ▶ Metadaten: RDF (Resource Description Framework)

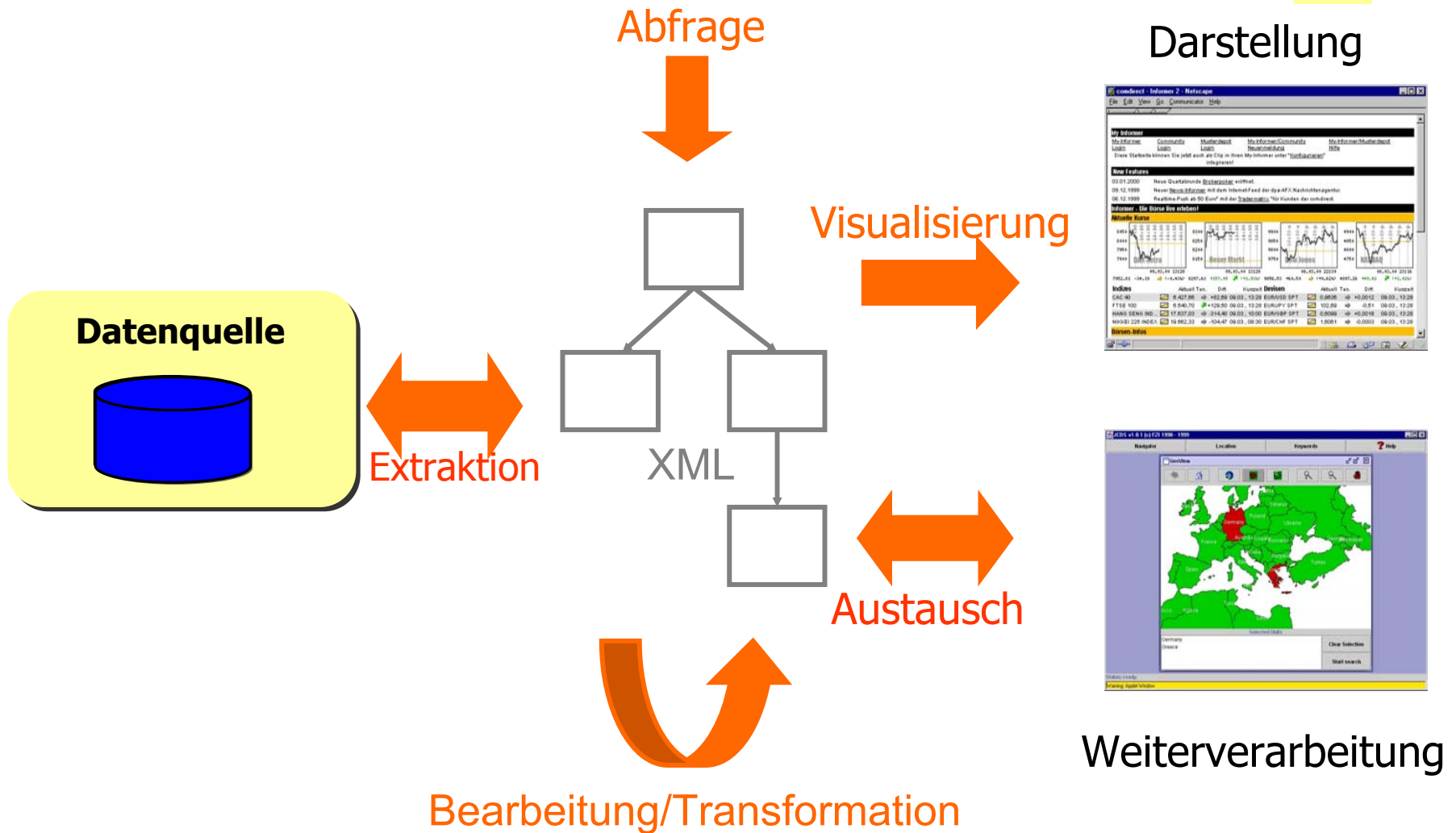
■ Private Initiativen & Interessengemeinschaften

- ▶ Wissenschaft: CML (Chemical Markup Language)
- ▶ Computer, Kommunikation:
 - CDF (Channel Definition Format)
 - WML (Wireless Markup Language)

■ Und viele viele mehr, siehe beispielsweise

- ▶ oasis-open.org, xml.org

XML Verarbeitungsmodell



XHTML



- Auf XML basierendes HTML
- Ziel: Kaum Unterschiede zu HTML 4.0
 - ▶ + Wohlgeformtheit und Gültigkeit, so dass es mit XML-Tools bearbeitet werden kann
 - ▶ + Modularisierung
- Wichtigste Änderungen
 - ▶ Alle Tags müssen abgeschlossen sein (z.B. `<hr/>`)
 - ▶ Keine Überschneidungen, nur Verschachtelung
 - z.B. `<p></p>`
 - ▶ Groß- und Kleinschreibung
 - ▶ Attributwerte in Anführungszeichen
 - ▶ 3 statt 1 DTD für XHTML

3 DTD für XHTML

■ Strict

- ▶ **Die** W3C Empfehlung
- ▶ Enthält alle HTML Elemente außer
 - veralteten Elementen (z.B. applet, center)
 - Darstellungsattributen (z.B. bgcolor, link, vlink)
 - framesets

■ Transitional

- ▶ Alles außer frameset

■ Frameset

- ▶ Alles erlaubt

■ `<!DOCTYPE html PUBLIC "-//W3C/DTD XHTML 1.0 Strict//EN" "DTD/xhtml1-
{strict|transitional|frameset}.dtd">`

Modularisierung von XHTML

- Die 3 DTD basieren auf ca. 30 Modulen, z.B.
 - ▶ `%xhtml-struct.module`: Das absolute Minimum
 - ▶ `%xhtml-text.module`: Alles, was mit Text zu tun hat
 - ▶ `%xhtml-list.module`: Listen-Elemente
 - ▶ `%xhtml-table.module`: Tabellen-Elemente
 - ▶ ...
- Module können wiederverwendet werden
 - ▶ In vordefinierten (oder eigenem) Module-Mix
 - Fließtext (`%Inline.mix`)
 - Text & Listen (`%mymix.mix`)
 - ▶ Für anwendungsspezifische DTDs

XHTML- Module Beispiel

```
<artikel>
```

```
<id>xy1234</id>
```

```
<name>Billy Regal</name>
```

```
<beschreibung>
```

klick-and-bau Tags

```
<p> Dies ist ein ein noch nie dagewesenes  
St,ck Handarbeit in seiner vollendetsten Form.
```

```
<em>200 Tage</em>hat der Meister daran  
gearbeitet bis er endlich soweit war. </p>
```

```
</beschreibung>
```

```
<preis>98</preis>
```

```
</artikel>
```

XHTML Tags

Kommunikation mit SOAP



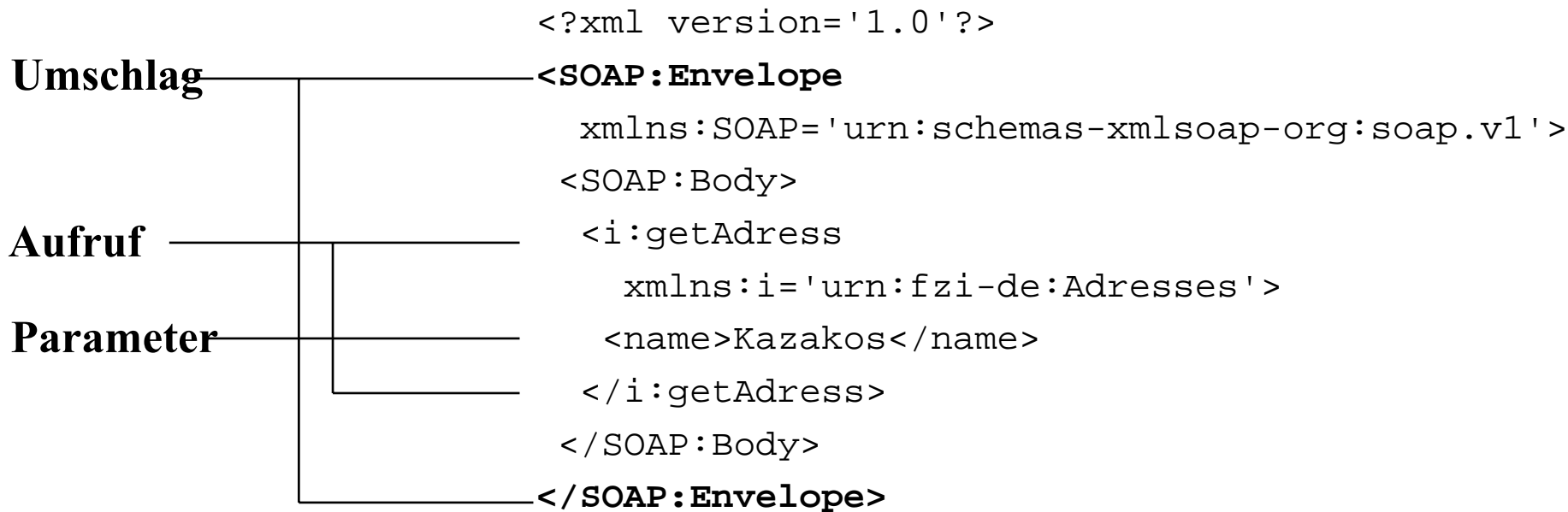
SOAP- Simple Object Access Protocol



- Einfaches Protokoll zu Aufruf von Web-Diensten
- Basisprotokoll: HTTP
- Konkurrenz zu: RMI, CORBA, DCOM (?)
- Standardisierung von IETF (nicht W3C)
- Aktuelle Version 1.1
- Zukunft im W3C: XML Protocol

- Basisdienst: Kommunikation zwischen Komponenten
 - ▶ Keine weitergehende Middleware-Funktionalität
 - ▶ Z.B. keine Objektreferenzen, keine Speicherbereinigung
- Drei Bestandteile:
 - ▶ Nachrichtenstruktur/Umschlag (Fakultativ)
 - ▶ Kodierungsregeln (Optional)
 - für die Kodierung von Datenstrukturen in XML
 - ▶ Konvention für die Verwendung als RPC-Mechanismus (Optional)

SOAP Beispiel

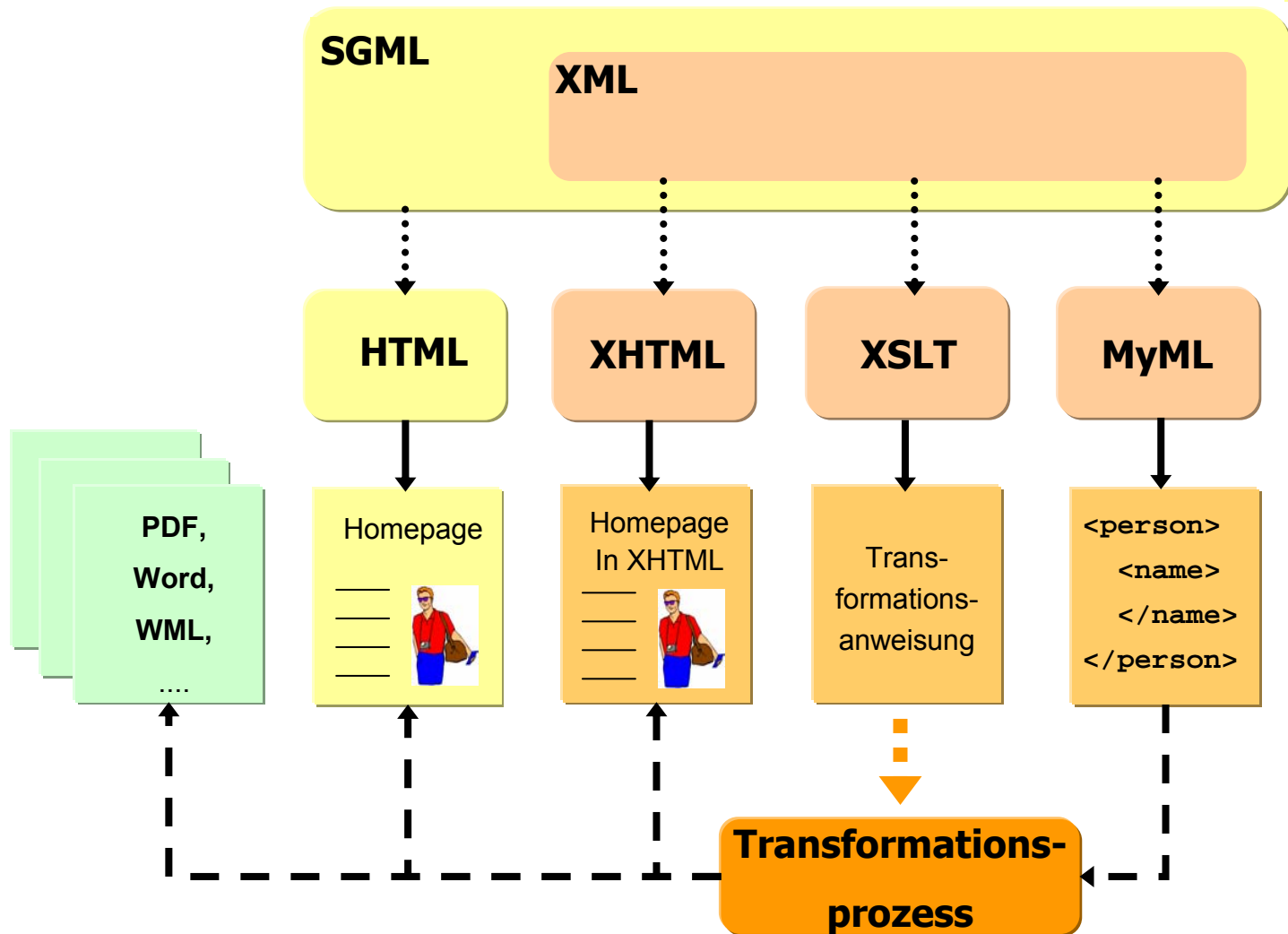


XML-Transformation & Präsentation

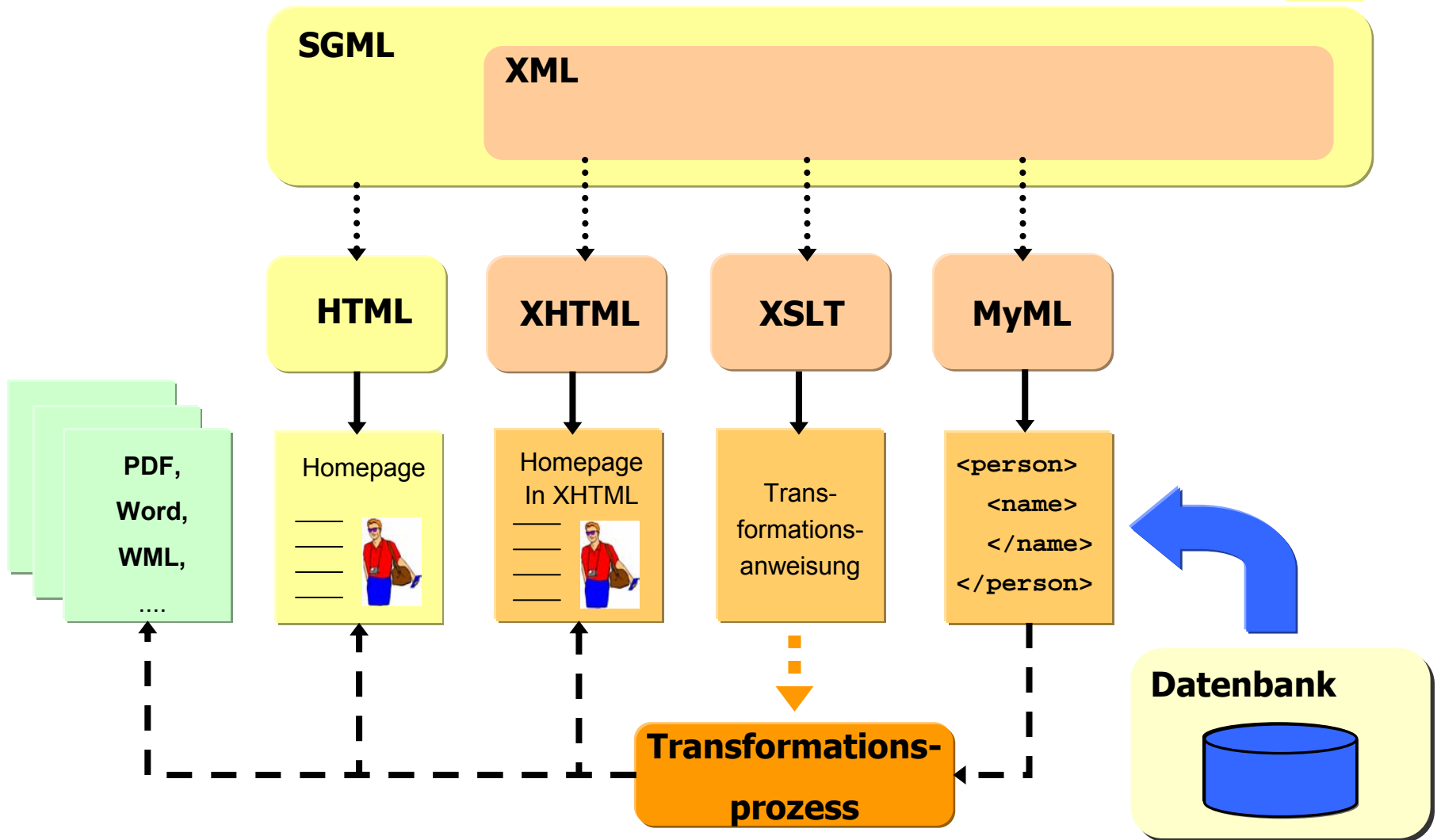


- Problem
 - ▶ Transformation und Visualisierung des XML-Baums
 - ▶ Bedingt: Suche im XML-Baum
- Ursprünglich XSL: eXtensible Stylesheets Language
 - ▶ Zu großes Themengebiet
 - ▶ Keine Einigungen im Detail
- Jetzt: Präsentations- und Transformationsprozess in 3 Schritten:
 - ▶ Selektion von XML-Elementen im Baum (XPath)
 - ▶ Transformation von XML-Bäumen (XSLT)
 - ▶ Ausgabeformatierung der Daten (XSLT und XSL-FO)

XSLT-Transformation

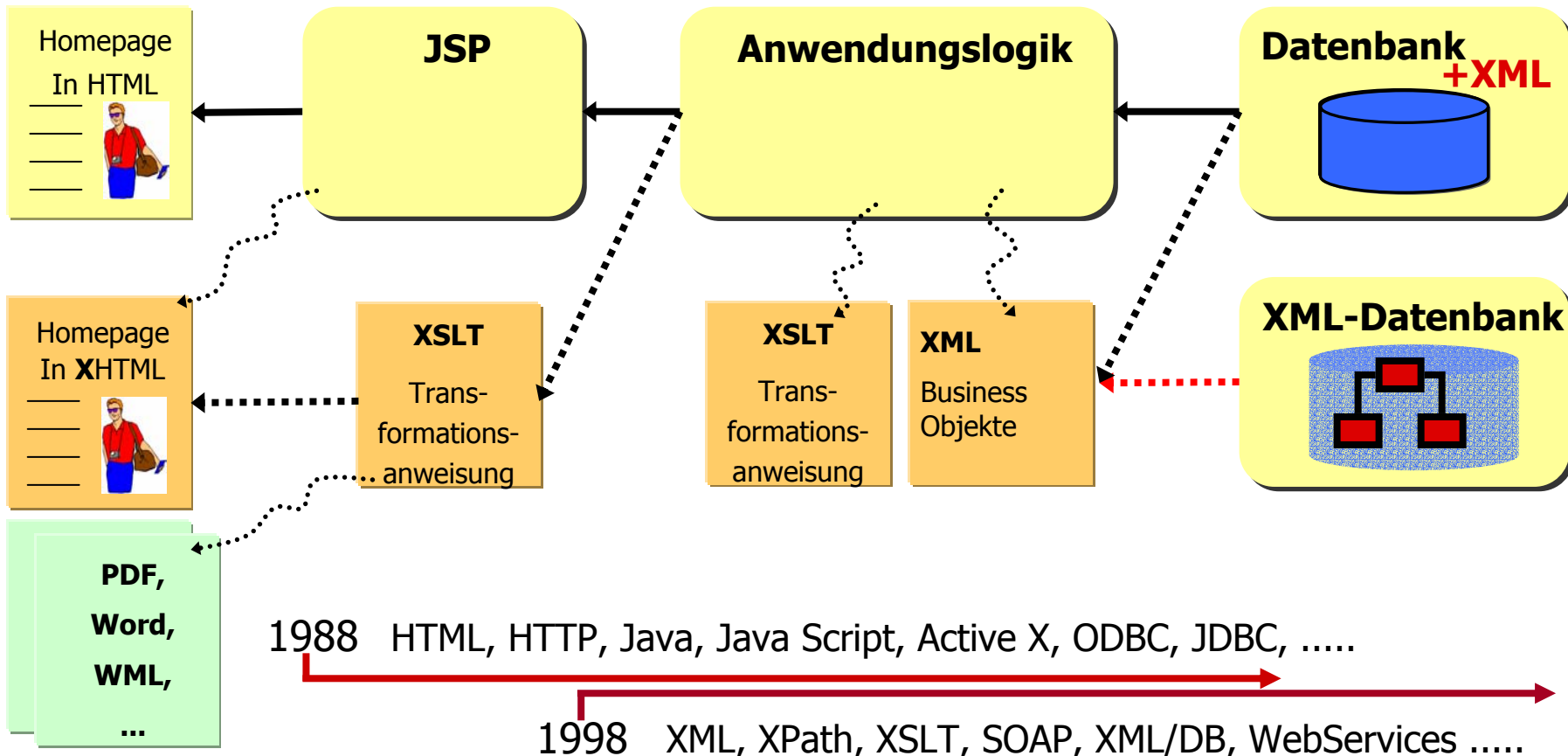


XSLT-Transformation

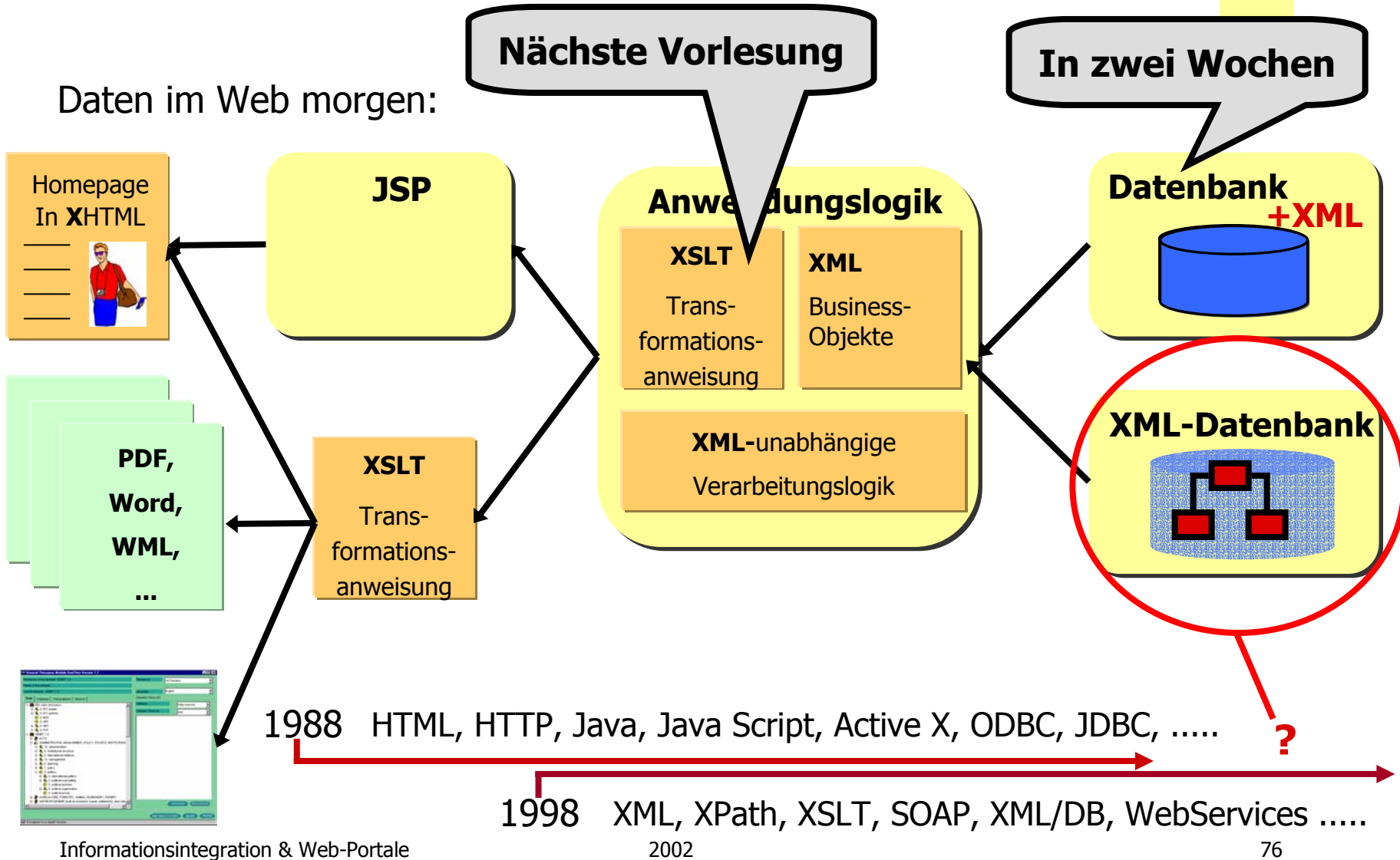


Zusammenführung

Daten im Web heute:



Zusammenführung



- Einschlägige Zeitschriften: iX, c't, ...
 - ▶ hilfreich vor allem bei der Auswahl von Werkzeugen
- Volker Turau, Java Server Pages - Dynamische Generierung von Web-Dokumenten, dpunkt Verlag, 2000
- E.R. Harold, W.S. Means: XML in an Nutshell. A Desktop Quick Reference, O'Reilly, January 2001
- Neil Bradley: The XML companion, 2nd edition, Addison-Wesley 1999
- K. Turowski, Klement J. Fellner (Hrsg.): XML in der betrieblichen Praxis. Standards, Möglichkeiten, Praxisbeispiele. dpunkt Verlag, 2001.