



Internet-Technologien

Sommersemester 2002

28. Juni 2002

Ingo Meents
IBM Deutschland Speichersysteme GmbH
Mainz
meents@de.ibm.com

Inhalt

- Nachtrag: Unicode
- XML: XSLT, Xpath
- Middleware
 - CORBA
 - RMI
 - COM / DCOM
- Datenbanken
 - Dokumenten orientiert
 - Workgroup Computing
 - Lotus Notes / Domino
 - Relationale Datenbanken: Reporting
 - dynamisch: PHP
 - statisch: Business Intelligence (BO)

Nachtrag XML

- Wie erkennt ein XML Parser das korrekte encoding, bevor er das 'encoding' liest?



- Ausführliche Erklärung von encodings
 - <http://www.czyborra.com/utf>

3

Verschiedene Encodings

```
Header - C:\Meerits\Lehrauftrag\Session3\amf1.txt
0 3c 34 78 6d 6c 20 76 65 72 73 69 61 6a 3d 22 31
10 2a 30 22 20 65 6e 63 64 64 65 6a 67 3d 22 55 54
20 46 2d 38 22 3a 20

...<?xml version="1
...<? encoding="UT
?>

Header - C:\Meerits\Lehrauftrag\Session3\amf_utf8.txt
0 ef bb bf 3c 34 78 6d 6c 20 76 65 72 73 69 61 6a
10 3d 22 31 2a 30 22 20 65 6a 63 64 64 65 6a 67 3d
20 22 55 54 2d 38 22 3a 20

...<?xml version
...<? encoding=
"UTF-8">

Header - C:\Meerits\Lehrauftrag\Session3\amf_utf16.txt
0 ff fe 3c 00 34 00 78 00 6d 00 6c 00 20 00 76
10 65 00 72 00 73 00 69 00 61 00 6a 00 3d 00 22
20 31 00 2a 00 30 00 22 00 20 00 65 00 6a 00 63
30 64 00 64 00 65 00 6a 00 67 00 3d 00 22 00 55
40 54 00 46 00 2d 00 38 00 22 00 3a 00 20 00

...<?xml version="1
...<? encoding="
"UTF-16">
```

ASCII

UTF-8
byte-order mark
EF BB BF

UTF-16
byte-order mark
FF FE

erzeugt mit Win2000 Notepad

4

Beispiel: Xerces

- XML Parser aus Apache Projekt
- <http://xml.apache.org/xerces-c/index.html>
- Klasse
 - XMLRecognizer
- Methode
 - `basicEncodingProbe(const XMLByte* const rawBuffer ,
const unsigned int rawByteCount)`
- in Datei
`\xerces-c-src1_7_0\src\xerces\framework\XMLRecognizer.cpp`

5

Beispiel: Xerces

```
const char      XMLRecognizer::fgASCIIPre[] = { 0x3C, 0x3F, 0x78, 0x6D, 0x6C, 0x20 };
const unsigned int XMLRecognizer::fgASCIIPreLen = 6;
[...]
const char      XMLRecognizer::fgUTF8BOM[] = {(char)0xEF, (char)0xBB, (char)0xBF};
const unsigned int XMLRecognizer::fgUTF8BOMLen = 3;

// -----
// XMLRecognizer: Encoding recognition methods
// -----
XMLRecognizer::Encodings
XMLRecognizer::basicEncodingProbe( const XMLByte* const rawBuffer
                                   , const unsigned int rawByteCount)
{
    // As an optimization to check the 90% case, check first for the ASCII
    // sequence '<?xml', which means its either US-ASCII, UTF-8, or some
    // other encoding that we don't do manually but which happens to share
    // the US-ASCII code points for these characters. So just return UTF-8
    // to get us through the first line.
    if (rawByteCount >= fgASCIIPreLen) {
        if (!memcmp(rawBuffer, fgASCIIPre, fgASCIIPreLen))
            return UTF_8;
    }

    // If the count of raw bytes is less than 2, it cannot be anything
    // we understand, so return UTF-8 as a fallback.
    if (rawByteCount < 2)
        return UTF_8;

    // We know its at least two bytes, so lets check for a UTF-16 BOM. That
    // is quick to check and enough to identify two major encodings.
    if ((rawBuffer[0] == 0xFE) && (rawBuffer[1] == 0xFF))
        return UTF_16B;
    else if ((rawBuffer[0] == 0xFF) && (rawBuffer[1] == 0xFE))
        return UTF_16L;

    [...]
}
```

6

XML - Extensible Style Sheet Language XSL

- Standardisierte Anwendung von Transformationen und Formatierungen auf XML Dokumente
- noch in der Entwicklung befindlich
- 2 Teile
 - XSL Transformations: Transformationssprache für XML Dokumente
 - Semantik: Vokabular für Formatierungsobjekte zur Wiedergabe von Dokumenten
- Anleihen bei DSSSL und CSS
- XSL sieht aus wie XML
- deklarative Programmierung
- funktionale Teile

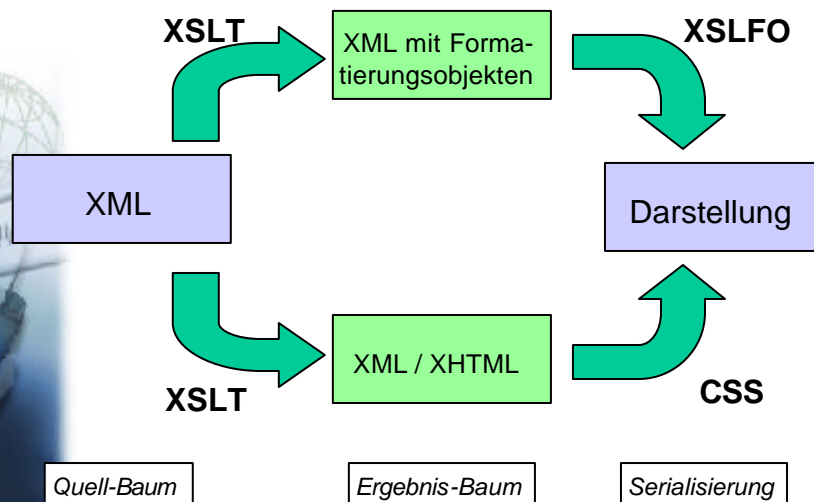
7

XSL: Eigenschaften

- XSL erlaubt vollständige Überarbeitung
 - Umsortierung von Elementen: Fussnoten an Text-Ende
 - Duplizierung: Titel für Verzeichnis, Window-Title, Überschrift
 - Unterdrückung: Anmerkungen
 - Hinzufügungen: Logo, Copyright-Hinweise
 - Sortierung
 - Numerierungen
- Anwendungen
 - Formatierung
 - e-Commerce: Umwandeln von Dokumenten zu verschiedenen DTDs
 - Einbindung eigener Komponenten: Java, ActiveX, etc.
 - Einbindung eigener Scripte: JavaScript, Python

8

XSL: Verarbeitung und Darstellung



9

XSL: Templates

- Wurzel-Element:

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/XSL/Transform/1.0"
  result-ns="http://www.w3.org/TR/REC-html40">
  <!-- ... Vorlagenregeln ... -->
</xsl:stylesheet>
```

- Vorlagenregel (*template*)

```
<xsl:template match="wichtig">
  <em><xsl:apply-templates /></em>
</xsl:template>
```

- dargestellt durch ein `xsl:template` Element
- Muster (*match pattern*)
 - Attribut `match` des Elementes
 - beschreibt, welche Knoten verarbeitet werden sollen
- Vorlage (*literal result elements*)
 - Inhalt des Elementes
 - beschreibt zu erzeugende XML-Struktur
 - `xsl` Namespace: Anweisung
 - anderer Namespace: Übernahme in Ergebnis

10

XSL Beispiel: DTD und Dokument

```

<!ELEMENT
  buch (titel, (abschnitt|anhang)+)>
<!ELEMENT
  abschnitt (titlel, (absatz|liste)+) >
<!ELEMENT
  anhang (titlel, (absatz|liste)+) >
<!ELEMENT
  titel (#PCDATA)>
<!ELEMENT
  absatz (#PCDATA|betont)*>
<!ELEMENT
  betont (#PCDATA)>
<!ELEMENT
  liste (eintrag)>
<!ELEMENT
  eintrag (#PCDATA|betont)*>

```

11

Style Sheet

```

<?xml version="1.0" ?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl"
  result-ns="http://www.w3.org/TR/REC-html40">

  <xsl:template match = "buch" >
    <html>
      <body bgcolor="#FFDDFF">
        <title><xsl:value-of select = "titel"/></title>
        <h1> <xsl:value-of select="titel" /></h1>
        <xsl:apply-templates select="abschnitt"/>
        <hr />
        <xsl:if test="anhang">
          <hr />
          <h2>Anhang</h2>
          <xsl:apply-templates test="anhang"/>
        </xsl:if>
        <p>Copyright 2000 Argon Verlag Berlin</p>
      </body>
    </html>
  </xsl:template>

  <xsl:template match="absatz">
    <p><xsl:apply-templates /></p>
  </xsl:template>

  <xsl:template match="abschnitt|anhang">
    <h2><xsl:value-of select="titel"/></h2>
    <xsl:apply-templates select="absatz|liste"/>
  </xsl:template>

  <xsl:template match="betont">
    <em><xsl:apply-templates /></em>
  </xsl:template>

  <xsl:template match="liste">
    <ol> <xsl:apply-templates /> </ol>
  </xsl:template>

  <xsl:template match="eintrag">
    <li> <xsl:apply-templates/> </li>
  </xsl:template>
</xsl:stylesheet>

```

12

XSL Beispiel: Ergebnis

```
java \  
-cp [...]saxon.jar:[...]saxon:[...]xml4j.jar \  
com.icl.saxon.StyleSheet \  
gengolf.xml buch.xsl
```



13

XSL Templates

- deklarativ

```
<xsl:template match="absatz">  
  <p><xsl:apply-templates/></p>  
</xsl:template>
```

- prozedural

```
<xsl:for-each select="wichtig">  
  <em><xsl:apply-templates/></em>  
</xsl:for-each>
```

14

XSLT Selektion

- XPath eigener (Grundlagen-) Standard
 - <http://www.w3.org/TR/xpath>
 - Adressierung von Objekten innerhalb eines XML Dokuments
 - keine Manipulation von Objekten
 - Anwendung in XSLT, Xpointer
- Beispiel
 - `<xsl:template match="info[@type='public']">`
 -
 - `</xsl:template>`
- Adress-Schritte
 - Achse
 - Knotentest
 - keine oder mehrere Prädikate

15

XPath: Achsen

- Auswahl von Elementen / Navigation im Baum
 - ancestor, ancestor-or-self, child, descendant, descendant-or-self, following, following-sibling, parent, preceding, preceding-sibling, self
- Attribut-Achse "@", "attribute::"
 - Zugriff auf Attribut-Knoten eines Elementknotens
- Namensraum-Achse
 - Zugriff auf Namensraum-Knoten eines Elementknotens

16

XPath: Knotentests

- Knotentests für alle Achsen
 - * für Knoten vom Haupttyp (Element, Attribut, Namensraum)
 - node() jeder Knoten mit beliebigen Typ
- Knotentests nur für Inhalts-Achsen
 - text()
 - comment ()
 - processing-instruction()
 - processing-instruction([zielname])

17

XPath: Prädikate

- Filtern von Knoten aus einer Knotenmenge
- Syntax: “[...]”
- Ausdruck mit Adresspfad
 - existiert ein Knoten, dann true
 - z.B. //absatz[fussnote]
 - Verkettung möglich: //absatz[fussnote][@wichtig]
- Zeichenketten-Werte
 - Attribute
 - @typ='sortiert'
 - Wert eines Elements: //text()
 - abschnitt[titel='Einführung']
- Kontextposition
 - position()=5 oder kurz 5
 - last()
 - Arithmetik: last() -1

18

XPath: Abkürzungen

- “child:.” *Default-Achse*
- “@” für “attribute:.”
- “//” für /descendant-or-self::node()/
- “*” für alle Kinder des jeweiligen Knotens
- “.” für “self::node()”
- “..” für “parent::node()”
- Beispiel:

```
/meindok/abschnitt[@sicherheit="öffentlich"]/absatz[7]
```

```
/child::meindok/child::abschnitt[attribute::sicherheit="öffentlich"]  
/child::absatz[position()=7]
```

19

XPath: Beispiele

- eintrag
 - eintrag-Element-Knoten, die Kinder des Knotextknotens sind
- eintrag/absatz
 - absatz-Enkel des Kontextknotens mit eintrag-Eltern
- //eintrag/absatz
 - absatz-Elementknoten, die Kinder beliebiger eintrag-Knoten aus dem gesamten Dokument sind
- */@*
 - Attributknoten, die zu bel. Elementknoten gehören, die Kind-Elemente des Kontextknotens sind
- //sortierte-liste[eintrag[@typ]/absatz[2]]//absatz
 - Absatz-Elemente, die Nachfahren eines beliebigen sortierte-liste Elements mit einem eintrag-Kind-Element sind, das ein Typ Attribut und mindestens 2 absatz-Kind-Elemente hat

20

XSL: Wichtige Elemente

- Abarbeitung woanders fortsetzen
 - <xsl:apply-templates>
- Wert eines Elements übernehmen
 - <xsl:value-of>
- Bedingungen
 - <xsl:if>
 - <xsl:choose> , <xsl:when>, <xsl:otherwise>
- Sortieren
 - <xsl:sort select="@DATE" order="descending"/>
- Numerieren und Formatieren
 - <xsl:number>
- Flow-Control
 - <xsl:while>
- und viele mehr, siehe z.B. Dokumentation SAXON

21

XSL: Formatierungsobjekte

- noch in Entwicklung, Implementierungen experimentell
- Objekte:
 - fo:root, fo:page-sequence, fo:flow, fo:block, fo:inline-graphic, do:display-graphic, fo:display-rule, fo:display-sequence, fo:table, fo:list-block, fo:simple-link, fo:page-number
- Beispiel:

```
<xsl:stylesheet
xmlns:xsl="http://www.w3.org/XSL/Transform/1.0"
xmlns:fo="http://www.w3.org/XSL/Format/1.0"
result-ns="fo">
  <xsl:template match = "absatz">
    <fo:block font-size="12pt">
      <xsl:apply-templates/>
    </fo:block>
  </xsl:template>

  <xsl:template match = "title">
    <fo:block font-size="12pt" font-weight="bold">
      <xsl:apply-templates/>
    </fo:block>
  </xsl:template>
</xsl:stylesheet>
```

22

XML: Was gibt's noch?

- Standards
 - Namensräume
 - <http://www.w3.org/TR/REC-xml-names/>
 - XML pointer language
 - <http://www.w3.org/XML/Linking>
 - XML linking language
 - <http://www.w3.org/XML/Linking>
 - XML Schemata
 - <http://www.w3.org/XML/Schema>
 - Web-Services
- APIs
 - DOM
 - SAX
- ... und vieles mehr <http://www.w3c.org>

23

Verteilte Objekte

- Ziel: Nutzung verschiedener Objekte in einem heterogenen Netzwerk wie lokale Objekte
 - auf verschiedenen Rechnern
 - auf verschiedenen Plattformen
 - in verschiedenen Programmiersprachen
- Umgehen der Probleme der Socket-Programmierung
- Bekannteste Vertreter
 - Object Request Broker Architecture (CORBA)
 - OMG
 - Distributed Component Object Model (DCOM)
 - Microsoft
 - Remote Method Invocation (RMI)
 - SUN
- binäre Protokolle

24

CORBA

- Object Request Broker (ORB)
 - Zentrales Management der Kommunikation von Objekten (lokal oder im Netz)
 - Auffinden von Implementierungen
 - Aufrufen der Methoden
 - Übermitteln der Ergebnisse
 - Protokoll: Internet Inter ORB Protocol (IIOP)
- Objekte
 - implementieren Inteface
 - definiert mittels Interface Definition Language
- Client
 - bekommt eine Referenz auf ein Server-Objekt
 - dann Verwendung wie ein lokales Objekt

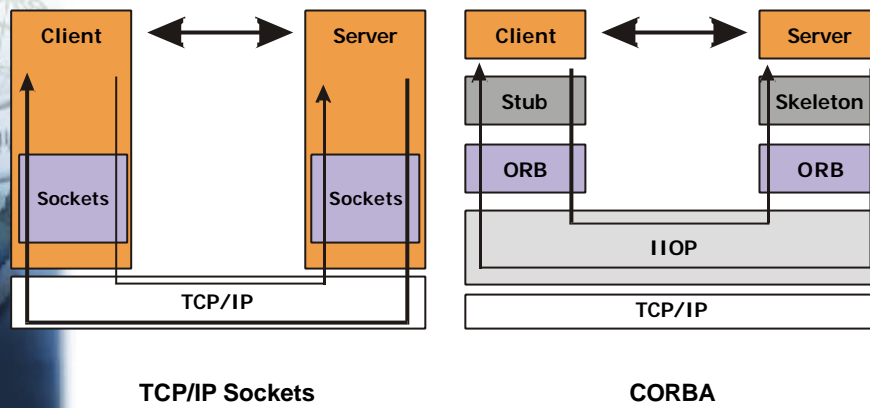
25

CORBA

- Standard durch OMG
- kommerzielle Implementierungen
 - Inprise: VisiBroker
 - Iona: Orbix
- freie Implementierung
 - MICO
 - <http://www.mico.org>
- für viele Sprachen verfügbar

26

Sockets und CORBA



27

CORBA Beispiel - IDL

```
interface Account {
    void deposit( in unsigned long amount );
    void withdraw( in unsigned long amount );
    long balance();
};

interface Bank {
    Account create ();
};
```

28

CORBA Beispiel - Server (1)

```
/* A Bank factory that creates Account objects */

#include <fstream.h>
#include "account.h"

/* Implementation of the Account */
class Account_impl : virtual public POA_Account
{
public:
    Account_impl ();
    void deposit (CORBA::ULong);
    void withdraw (CORBA::ULong);
    CORBA::Long balance ();

private:
    CORBA::Long bal;
};

Account_impl::Account_impl () {
    bal = 0;
}

void Account_impl::deposit (CORBA::ULong amount){
    bal += amount;
}

void Account_impl::withdraw (CORBA::ULong amount){
    bal -= amount;
}

CORBA::Long Account_impl::balance () {
    return bal;
}
```

29

CORBA Beispiel - Server (2)

```
/* Implementation of the Bank */

class Bank_impl : virtual public POA_Bank {
public:
    Account_ptr create ();
};

Account_ptr Bank_impl::create () {
    /* Create a new account (which is never deleted) */
    Account_impl * ai = new Account_impl;

    /*
     * Obtain a reference using _this. This implicitly activates the
     * account servant (the RootPOA, which is the object's _default_POA,
     * has the IMPLICIT_ACTIVATION policy)
     */

    Account_ptr aref = ai->_this ();
    assert (!CORBA::is_nil (aref));

    /* Return the reference */
    return aref;
}
```

30

CORBA Beispiel - Server (3)

```
int main (int argc, char *argv[]) {
    /* Initialize the ORB */
    CORBA::ORB_var orb = CORBA::ORB_init (argc, argv);

    /* Obtain a reference to the RootPOA and its Manager */
    CORBA::Object_var poaobj = orb->resolve_initial_references ("RootPOA");
    PortableServer::POA_var poa = PortableServer::POA::_narrow (poaobj);
    PortableServer::POAManager_var mgr = poa->the_POAManager();

    /* Create a Bank */
    Bank_impl * micocash = new Bank_impl;

    /* Activate the Bank */
    PortableServer::ObjectId_var oid = poa->activate_object (micocash);

    /* Write reference to file */
    ofstream of ("Bank.ref");
    CORBA::Object_var ref = poa->id_to_reference (oid.in());
    CORBA::String_var str = orb->object_to_string (ref.in());
    of << str.in() << endl;
    of.close ();

    /* Activate the POA and start serving requests */
    printf ("Running.\n");
    mgr->activate ();
    orb->run();

    /* Shutdown (never reached) */
    poa->destroy (TRUE, TRUE);
    delete micocash;
    return 0;
}
```

31

CORBA Beispiel - Client

```
#include "account.h"
#ifdef HAVE_UNISTD_H
#include <unistd.h>
#endif

int
main (int argc, char *argv[])
{
    CORBA::ORB_var orb = CORBA::ORB_init (argc, argv);

    // IOR is in Bank.ref in the local directory
    char pwd[256], uri[300];
    sprintf (uri, "file://%/Bank.ref", getcwd(pwd, 256));

    // Connect to the Bank
    CORBA::Object_var obj = orb->string_to_object (uri);
    Bank_var bank = Bank::_narrow (obj);
    if (CORBA::is_nil (bank)) {
        printf ("oops: could not locate Bank\n");
        exit (1);
    }

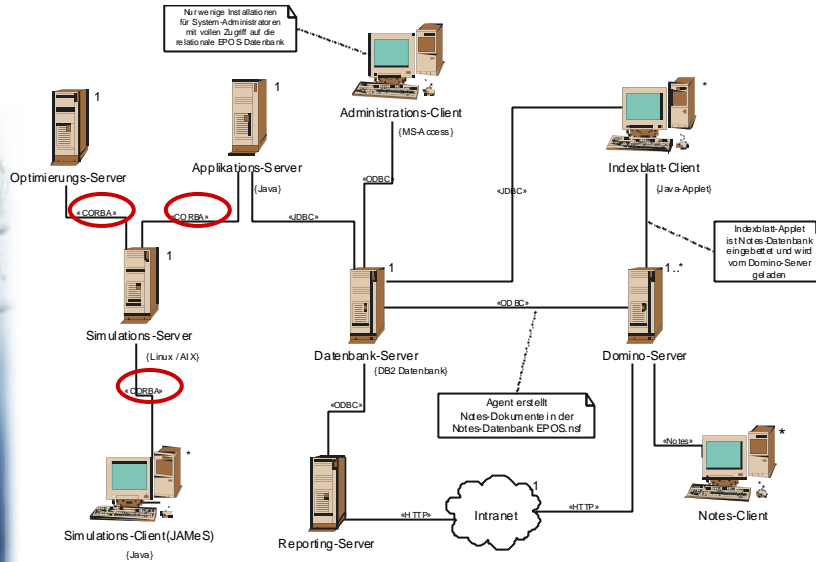
    // Open an account
    Account_var account = bank->create ();

    if (CORBA::is_nil (account)) {
        printf ("oops: account is nil\n");
        exit (1);
    }

    // Deposit and withdraw some money
    account->deposit (700);
    account->withdraw (450);
    printf ("Balance is %ld.\n", account->balance ());
    return 0;
}
```

32

CORBA in EPOS



EPOS Interface Definition

ams.idl

DCOM

- Protokoll: Object Remote Procedure Call (ORPC)
- basiert auf DCE Remote Procedure Call
 - Distributed Computing Environment, Open Software Foundation
- interagiert mit den Run-time Services von *Component Object Model (COM)*
 - Komponenten-Technologie von Microsoft
 - ähnlich EJB, aber weniger mächtig
 - Binärstandard, d.h. nicht auf eine bestimmte Programmiersprache festgelegt
- Deployment über Windows Registry
- Portierung auf UNIX-Systeme verfügbar
- Ablösung durch .net
- <http://www.microsoft.com/com/tech/DCOM.asp>

35

JAVA RMI

- <http://java.sun.com/products/jdk/rmi/index.html>
- Protocol: Java Remote Method Protocol (JRMP)
- basiert auf Java's Objekt-Serialisierung
- Server und Client in Java
- Distribution von Referenzen: RMIRegistry auf Server
- Client
 - startet lookup nach Objekten-Referenzen
 - ruft deren Methoden auf
- Identifizierung von Serverobjekten mittels URL
 - `rmi://localhost/MYOBJECT`
- durch Java Plattform-unabhängig
- auch RMI over IIOP

36

Datenbanken Einleitung

- Relational
 - Speicherung strukturierter Daten
 - Konsistenz (Normalformen)
 - Performance (Indices, Query-Optimierer)
 - z.B. IBM DB2, ORACLE, MySQL, Microsoft SQL Server
- Dokumentenorientiert
 - Unterstützung unstrukturierter Prozesse
 - Flexibilität
 - Integration externer Ressourcen
 - Workgroup Computing
 - z.B. Lotus Notes

37

Dokumentenorientiert

- traditionelles LAN
 - gemeinsame Nutzung von Druckern, Festplatten, Datenbanken, e-mail
- vernetzte Zusammenarbeit:
 - Computer Supported Collaborative Work (CSCW)
 - Vermeidung von Medienbrüchen
 - dazugehörige Software: Groupware
 - Ziele
 - Organisation von flexiblen Teams
 - Verwaltung der Arbeitsergebnisse
 - Kooperation in heterogenen Netzen
- Lotus Notes
 - Unabhängigkeit von zugrundelegener Technik
 - eigene Benutzerkonten, Adressbücher, viele Protokolle

38

Groupware

- verschiedene Definitionen der Hersteller
 - von e-mail bis zu komplexen Systemen
- Hauptfunktionen
 - Kommunikation: Nachrichtenübermittlung durch elektronische Dokumente
 - Kooperation: gemeinsamer virtueller Arbeitsplatz für eine Arbeitsgruppe
 - Koordination: Vorgangsverfolgung
- Hauptbestandteile
 - Datenbank: Datenbasis, Nachrichten, Dokumente
 - Verteilungs- und Zugriffsmodell
 - Entwicklungsumgebung
- Anforderungen
 - Verfügbarkeit auf vielen Systemen
 - Integration externen Ressourcen
 - Mobilität

39

Groupware und Workflow

Kooperative Prozesse		semi-strukturierte Prozesse		Strukturierte Prozesse	
Ad Hoc Workflow	Koordinierte Teamarbeit	Team Workflow		Workflow mit Ausnahmebehandlung	starrer Workflow
Spontanes Routing von Informationen	Mehrere Teams Arbeitsteilung Gemeinsames Ziel	+ stärker sequentielle Arbeitsschritte Zwischenergebnisse		Routineabläufe mit eingeplanten Ausnahmesituationen	100% vordefiniert

← *Flexibilität* Prozessorientierung →



40

Lotus Notes

- dokumentenorientierte Datenbank
 - Speicherung von Backend-Dokumenten (Notes) und Formularen
-> Dokument
- Einbettung von (Rich-)Text, Grafiken, Tabellen, Webseiten, Frame-Sets, Dateien
- Namens-/Adressbücher
- Notes e-mail
- Kalender-Management
- Replizierung
- Beispielanwendungen: Urlaubsplanung, Reiseplanung, Teamrooms, Lizenzverwaltung, PC-Bestellung
- flexibles Sicherheitskonzept
- hohe Verfügbarkeit:
 - Server: Windows, Unix, OS2, AS400
 - Clients: Windows, Unix, OS2, AS400, Mac

41

Lotus Notes - Entwicklungsumgebung

- Notes Designer zum Erstellen von
 - Formularen, Views, Navigatoren, Sites, Framesets, ...
- Programmierung
 - Simple Actions: Delete document, Modify Field, etc.
 - @Functions: @if(@IsNewDoc; „New Document“, DocTitle)
 - Lotus Script, JavaScript
 - APIs für C, C++
 - Java
 - Agenten
 - Einbettung von Java Applets
 - LSX (Lotus Software eXtension)
 - erstellt in C++ (Datenbank Zugriff, RichText Verarbeitung, etc)
 - Object-library
 - Verwendung mit Java, Lotus-Script
 - XML Unterstützung

42

Lotus Notes Beispiel - EPOS Datenbank

The screenshot shows the Lotus Notes interface for the EPOS database. The left sidebar contains navigation options: Workcenter (by group, by responsibility, Problems / Warnings, Plan / Actual, Input State, by managers, by MESA ID, by Operation, by Product, by Replication), Archives, Data input, and Feedback (by priority). The main area displays a tree view of data:

- IBM SSD
- MeeKram Inc.
 - Europe
 - Frankfurt
 - Fab X
 - Simple Water
 - Workcenter A (Status: X X X)
 - MeeKra Water
 - Lap
 - Brush Cleaner (Status: ✓)
 - Lap tool (Status: ✓)
 - Photo
 - Plate
 - NFe Plate Cells (Status: ✓)
 - Strip Tool 96x (Status: ✓)
 - Sputter
 - Sputter PI 2054 (Status: ✓)
 - Ultra 5 (Status: ✓ X X)
 - Vacuum CV 4 (Status: ✓)
 - Test
 - Automatic Microscope (Status: X)
 - Depth-Tester (Status: ✓)
 - Microscope (Status: ✓)

The status indicators (X, ✓) likely represent different states or completion levels of the tasks or components.

43

Lotus Notes Beispiel - Designer

The screenshot shows the Lotus Notes Designer interface for the 'Workcenter2-Form'. The left sidebar shows the 'Recent Databases' list, including EPOS (workcenter.net) and TestDB. The main area displays the form design for 'TacticalInfo' with the following fields and sections:

- MESA ID: MESA_id
- General Information
 - Module name
 - company name
 - division name
 - location name
 - product name
 - cell name
- Rejection
 - RejectCode
 - RejectReason
- Problems and Warnings

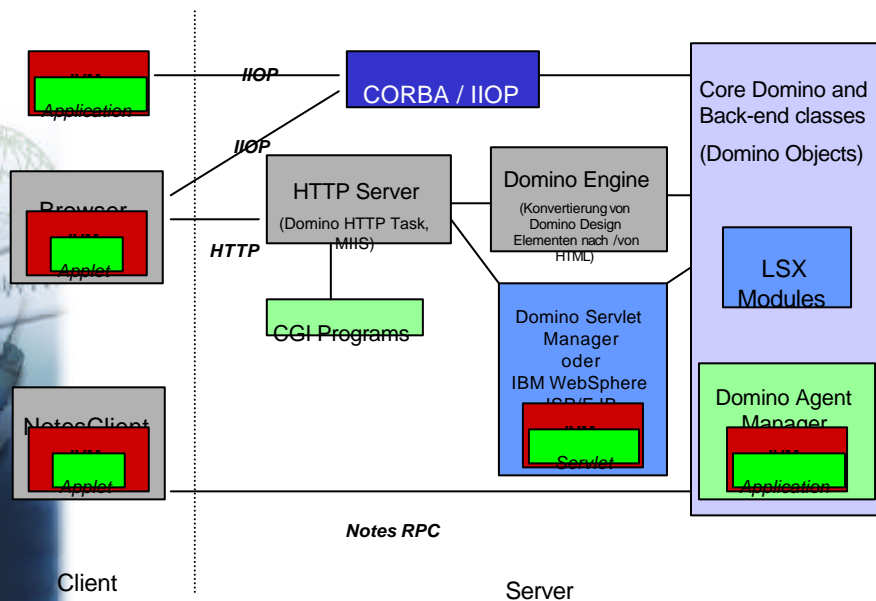
44

Lotus Domino

- Internet-Anbindung
- Realisierung als Notes Server Task
- Entwicklung und Verwaltung von Web Applikationen
- dynamischer Zugriff auf Datenbankinhalte
 - Umwandlung von Grafiken in GIF/JPEG
 - Funktionalität von Views
 - File Attachments
- Erstellung von Dokumenten über Browser
- Unterstützung der Sicherheitsmechanismen
- Volltextsuche in Datenbanken

45

Lotus Notes/Domino



46

EPOS Workcenter im Web-Browser

Signed	Title	Mail	Mass	Exp.	Created	Last Modified
	IBM SSD					731
	MeeKra Inc.					14
	Europe					14
	Frankfurt					14
	Fab X					1
	MeeKra Wafer					13
	Exp					2
	Photo					3
	Photo_Cleste	✓	✗	✗	10.05.2002	15:47
	Stepper 0815	✓	✗	✓	14.03.2002	11:23
	Stepper 0711	✓	✗	✓	14.03.2002	11:22
	Plate					2
	Sputter					3
	Test					3
	Zanders AG					14
						750

47

EPOS Home Page

IBM EPOS

HOME | Search | Workcenter | Planning | Reporting

WHERE ARE YOU?

- ic3.Ros.com
- IBM STD DMEA
- ESU Head Office
- EPOS

WHERE CAN YOU GO?

1. EPOS
2. Workcenter
3. Planning
4. Reporting
5. Process
6. Web Sites

Workcenter | **Planning** | **Reporting**

NEWS


Article Title	Date
Final Version of "Integrated Simulation"	23.11.2001
Literature	08.05.2001
Das neue Variance-Panel	29.01.2001
Tv/News Nr. 2	11.10.2000
Tv/News	22.09.2000
Wide Simulation Results	07.08.2000
Integrierte Simulation mit EPOS	30.05.2000
Integrierte Simulation	30.05.2000
Die EPOS Indexblätter	30.05.2000
Einführung	30.05.2000

LINKS

- Simulation Status
- Simulation Report: Waker HDD
- Concept of Integrated Simulation
- Status Tool Parameter Sheet Input
- Process of Capacity Planning
- Online Help
- General Documents


48

Notes/Domino Vor- und Nachteile

- 
- Flexibilität bei unstrukturierten Daten
 - Integration verschiedener Medientypen
 - Plattformunabhängigkeit
 - schnelle Prototypenentwicklung
 - Web-Integration
 - weite Verbreitung
 - Replikationsmechanismen
 - Sicherheitsmechanismen
 - Interoperabilität
 - e-mail (SMTP / POP)
 - CORBA / IIOP
 - XML
 - Thick-Client (Performance)
 - schlechte Performance bei sehr großen Datenvolumen und hohen Transaktionsraten
 - aufwendig bei strukturierten Daten Konsistenz zu erhalten

49

Relationale Datenbanken

- 
- persistente Speicherung strukturierter Informationen
 - Abfragen mittels Structures Query Language (SQL)
 - Ziel: Interaktion mit Datenbanken über das WWW
 - Spezialfall: Reporting
 - dynamisch
 - Vorteil: aktuelle Daten
 - Nachteil: hohe Last auf Server, Redundante Generierung
 - statisch
 - Vorteil: einmalige Generierung
 - Nachteil: Daten nur periodisch aktualisiert, Reports evtl. nicht benötigt

50

Datenbank-Inhalte ins Web

- Skript-/Programmiersprachen
 - Perl
 - DBI Database independent interface for Perl
 - kapselt Datenbank-Spezifische Treiber, z.B. für CSV, File, ODBC, Oracle, Proxy, Sybase, MySQL
 - PHP
 - Beispiel folgt
 - Java
 - JDBC
 - ASP
 - ActiveX Data Objects (ADO), ODBC
 - Python
- Tools
 - Business Objects, Cold Fusion, Brio, SAS, Crystal Reports, ...

51

Beispiel: Perl Datenbankzugriff

Deitel Seite 930ff

```
#!/usr/bin/perl
use CGI qw( :standard );
use DBI;
use DBD::mysql;

$dtd = "-//W3C//DTD XHTML 1.0 Transitional//EN"
      "\http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd";

print( header() ); # Content-type: text/html\n\n
print( start_html( { dtd => $dtd,
                    title => "Authors" } ) );

# connect database
$dbhandle = DBI->connect( "DBI:mysql:books", "root", "password", { RaiseError => 1 } );

# retrieve the names and IDs of all authors
$query = "SELECT FirstName, LastName, AuthorID FROM Authors ORDER BY LastName";
$stmtHandle = $dbhandle->prepare( $query );
$stmtHandle->execute();

print( h2( "Choose an author:" ) );
print( start_form( { action => 'process_author.pl' } ) );

# select box
print( "<select name = \"author\">\n" );
while ( @row = $stmtHandle->fetchrow_array() ) {
    print( "<option>" );
    print( "$row[ 2 ]. $row[ 1 ], $row[ 0 ]" ); #id. last, first
    print( "</option>" );
}
print( "</select>\n" );

print( submit( { value => 'Get Info' } ) );
print( end_form(), end_html() );

# clean up -- close the statement and database handles
$dbhandle->disconnect();
$stmtHandle->finish();
```

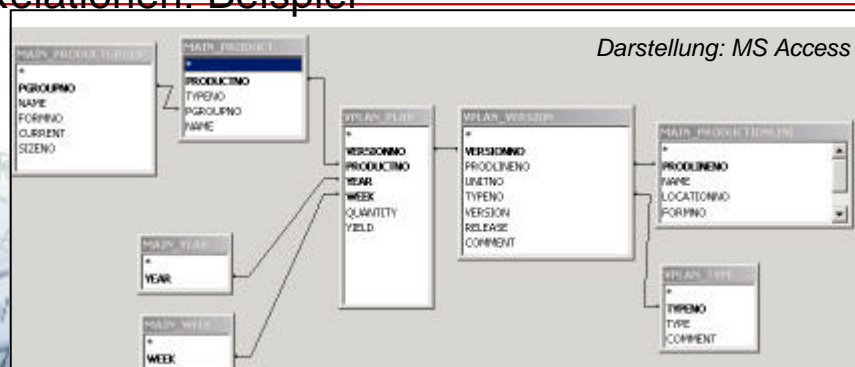
52

Beispiel: PHP Unified ODBC Functions

- nur Name, Syntax wie bei ODBC, native Treiber werden verwendet und nicht generisches ODBC
- Verbindungs-Management
 - odbc_connect, odbc_close, odbc_close_all
- Anfrage definieren
 - odbc_prepare, odbc_execute
- Daten holen
 - odbc_fetch_array, odbc_fetch_into, odbc_fetch_object, odbc_fetch_row
 - odbc_field_len, odbc_field_name, odbc_field_num, odbc_field_type
 - odbc_result_all
- Transaktionsmanagement
 - odbc_commit, odbc_rollback
- Fehlerbehandlung
 - odbc_error, odbc_errormsg

53

Relationen: Beispiel



```
SELECT
    shipplan.version,
    p.name,
    plantable.year, plantable.week, plantable.quantity, plantable.yield
FROM
    (select * from vplan.version where typeno = 0) shipplan
    join
    vplan.plan plantable on shipplan.versionno=plantable.versionno
    join
    main.product p on plantable.productno=p.productno
WHERE
    prodlineno = 0
```

54

Reporting - Apache / PHP / VHGraph

- Ziel: Dynamische Reports
 - tabellarisch
 - grafisch
- Hauptkomponenten
 - Apache Web Server <http://www.apache.org>
 - PHP <http://www.php.net>
 - VHGraph <http://www.vhconsultants.com>
 - IBM DB2Connect+

57

Dynamischer PHP Report (1)

```
<?
//==== Includes einbinden: Funktionsdefinitionen =====
include("../public_include/httpkopf.pinc"); // eigene Funktion für HTTP Header
include("../public_include/colordef.pinc"); // definiert globale Variablen für Farben
include("../public_include/cssdef.pinc"); // gibt CSS Definitionen aus
include("../public_include/footer_epos.pinc"); // HTML EPOS Footer
include("../public_include/httpbodystart.pinc"); // Funktion für Logo

//===== Variablen init. =====
$frame_breakout = 0;
$stable_colstart = 8; // Ab welcher ODBC-Spalte soll die Tabelle generiert werden ?

//===== START DER SEITE =====
ColorDef();
HTTpkopf();

//===== Datenbank Connect + check =====
$connect = @odbc_connect("epos", "username", "password");
if (!$connect){
?>
<SCRIPT LANGUAGE="JavaScript">
<!--
alert('Error connecting to database')
// -->
</SCRIPT>
?>
}

echo("<HTTPHeadOut
<TITLE>EPOS Webserver: Volume Plans</TITLE> ");

include("../public_include/htmlkopf.pinc");
```

58

Dynamischer PHP Report (2)

```
echo("<TABLE width='100%' border='0' cellpadding='0' cellspacing='0'>
<TR><TD><H2>Volumeplan by Productgroups</H2></TD>
<TD>\n");
navigation(0, '', 0); // Buttons für Javascript Navigation, eigene Funktion
echo("</TD></TR></TABLE>\n<BR clear='all'\n");

if ($connect){ // Verbindung zur Datenbank OK ?

// Aufbau der dynamischen FORMS -----
$form1_query = "select distinct pl.prodlino, loc.name || ' - ' || pl.name
from main.productionline pl
join main.location loc on pl.locationno = loc.locationno
order by 2";

echo "<FORM method='get' action='..\$PHP_SELF'>\n";

// eigene Funktion, die eine InputBox mit den Werten der Query fuellt
// Parameter: DB connection, Prompt, query, Variablenname, Variable contents, refresh flag
FormSelect($connect, "Select ProductionLine", $form1_query, "form1_res", $form1_res, 1);

//noch ein Pulldown
if (isset($form1_res)){ // Zweites Pulldown anzeigen ?

$form2_query = "select versionno, '[' || char(versionno) || ']' ,version from vplan.version
where prodlino = $form1_res
order by 1 desc";

FormSelect($connect, "Select Versionno.", $form2_query, "form2_res", $form2_res, 0);

} //endif isset($form1_res)

// Go Button
echo "
<INPUT id='inputbutton' type='submit' value='GO!' >
</FORM>
<HR><BR>
<P>
*;
```

9

Dynamischer PHP Report (3)

```
if (isset($form1_res) && isset($form2_res)){ //startit: Sind alle Variablen gefuellt?
$dataquery = "select
mod.name as 'Modelname'",
com.name as 'Companyname',
div.name as 'Divisionname',
loc.name as 'Locationname',
pl.name as 'Prod.linename',
char(date(v.release),USA) as 'Release',
v.comment as 'Comment',
v.version
from vplan.version v
join main.productionline pl on pl.prodlino = v.prodlino
[... ]
where v.versionno = $form2_res
and pl.prodlino = $form1_res
";

$datares = odbc_exec($connect, $dataquery);
$odbc_colnum = odbc_num_fields($datares) - 1; //Hide last col.
$odbc_rownum = odbc_num_rows($datares);

//==== Print Headlines + Table-Start ====
if(odbc_fetch_row($datares, 1) AND ($table_colstart > 1)) {
echo ("
<TABLE border='0' cellspacing='0' cellpadding='1'>
");
for ($i = 1; ($i < $table_colstart) AND ($i <= $odbc_colnum); $i++) {
echo ("
<TR>
<TD><P><B>".odbc_field_name($datares, $i).":&nbsp;</B></P></TD>
<TD><P>".odbc_result($datares, $i)."</P></TD>
</TR>
");
} // end for
echo ("\n </TABLE><BR clear='all'\n");
$locpl = odbc_result($datares,4)."/".odbc_result($datares,5);
$V_ver = odbc_result($datares,8).[$form2_res];
}
```

60

Dynamischer PHP Report (4)

```
//===== Data for Crosstable =====
$dataquery = "select distinct
char(vp.year) || '/' || substr(digits(vp.week),4,2),
pg_name,
sum(vp.quantity)
from vplan.plan vp
join main.product p on p.productno = vp.productno
join main.productgroup pg on pg.pgroupno = p.pgroupno
where vp.versionno = $form2_res
group by vp.year, vp.week, pg_name
";

$datares = odbc_exec($connect, $dataquery);
$odbc_colnum = odbc_num_fields($datares);
$odbc_rownum = odbc_num_rows($datares);

//===== Print CrossTable =====
//===== Build 2-dimensional Array from ODBC-Result =====

$ir = 1;
while(odbc_fetch_row($datares, $ir++){ //startwhile
    $crosstab[odbc_result($datares, 2)][odbc_result($datares, 1)] = odbc_result($datares, 3);
} //endwhile

$shlink =
"vplan_pdet.phtml?form1_res=".urlencode($locpl)."&pl_no=".$form1_res."&form2_res=".$form2_res."&pgroup";
$vlink =
"vplan_wdet.phtml?locpl=".urlencode($locpl)."&form1_res=".$form1_res."&form2_res=".$form2_res."&y_w";
CrossTable($crosstab,"$3.2f",$shlink,$vlink);
//=====
```

61

Dynamischer PHP Report (5)

```
//===== Build data-file for dynamic bargraph-image =====
$file = SessionID(); // Zufallszahl basiert
$datei=fopen("temp/$file.tmp", "w");
if (!$datei == false) {
    fputs($datei, $xlimit."\n");
    fputs($datei, $ylimit."\n");
    fputs($datei, $locpl." ".$v_ver."\n");
    fputs($datei, "Weeks\n");
    fputs($datei, "Quantity\n");
    for ($i = 0; $i <= $ylimit; $i++) {
        fputs($datei, $yaxis[$i]."\n"); //== y_w ==
    }
    for ($ir = 0; $ir <= $xlimit; $ir++) {
        fputs($datei, $xaxis[$ir]."\n"); //== pgroups ==
        for ($i = 0; $i <= $ylimit; $i++) {
            $stacked[$i] += $crosstab[$xaxis[$ir]][$yaxis[$i]];
            fputs($datei, $stacked[$i]."\n"); //== stacked week-data per pgroup ==
        }
    }
    $datei_ok = @fclose($datei);
}

//=====
if ($datei_ok && $odbc_rownum)
    echo "
<HR width=*80%*>
<CENTER>
<img src=\gfx/stackedareagraph.php3?session=".$file."><BR>
</CENTER>
<BR clear=*all\*>
";

// End of Page (Navigation + Footer)
navigation(1, '', 0);
} //endif form1_res<0 : Sind alle Variablen gefüllt?
odbc_close($connect); //== Verbindung zur Datenbank sauber beenden ==
}
FooterEPOS(); ?>
```

62

Ressourcen

- Notes / Domino
 - Domino 5 - Web Programming with XML, Java, and JavaScript, Randall A. Tamura, 2000, QUE
- *Datenbanken und mehr*
 - *Internet & World Wide Web - How to program*, H.M. Deitel, P.J. Deitel, T.R. Nieto, 2002, Prentice Hall
- CORBA C++
 - *Advanced CORBA Programming with C++*, Michi Henning, Steve Vinoski, 1999, Addison Wesley

63

Wie geht's weiter?

Termin	Inhalt	Vortragender
3. Mai	Einführung XHTML, JavaScript	Meents
10. Mai	WebDesign Java	Kramer
29. Mai	Web/Application Server Middleware (XML)	Meents
7. Juni	Java 2 Enterprise Edition	Kramer
21. Juni	Workgroup Computing Datenbanken	Meents
5. Juli	e-commerce Ausblick	Kramer

64

Ausblick

- e-mail: meents@de.ibm.com
 - Fragen
 - Feedback, Vorlesungsfeedback
 - Anregungen
- Folien

<http://www.informatik.tu-clausthal.de/~meents/ss2002/IntTech>

Daumen drücken!

