

Mobile Web Services

Dr. Thomas Wieland
Siemens AG, Corporate Technology
OOP 2002, München

Gliederung

- **Warum Web Services?**
- **Wie implementiert man das?**
 - Microsoft Mobile Internet Toolkit for .NET
 - Microsoft Smart Device Extensions
 - PocketSOAP
 - kSOAP
 - SOAP auf dem Linux-PDA
- **Wozu braucht man das?**

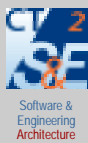
Warum Web Services?



Software &
Engineering
Architecture

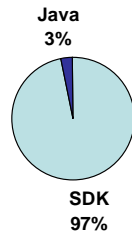
Herausforderungen bei der Entwicklung mobiler Anwendungen

- **Unterschiedlichkeit der Endgeräte**
 - Oft nur geringe CPU und Speicher-Ressourcen verfügbar
- **Business-Logik fast vollkommen auf dem Server**
- **Benutzer in Bewegung**
 - Notwendigkeit von Session-Management, -Roaming
- **Einfacher Umgang**
 - Keine Client-Installation nötig
 - Leicht zu aktualisieren
- **Effiziente Entwicklung**
 - Investitionen in die Desktop-Welt sichern
 - Erfahrungen der Entwickler nutzbar machen

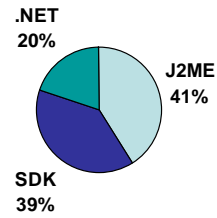


Software &
Engineering
Architecture

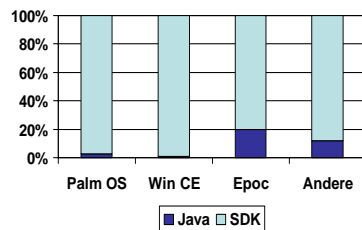
Marktstudie: Intelligenz auf mobilen Endgeräten



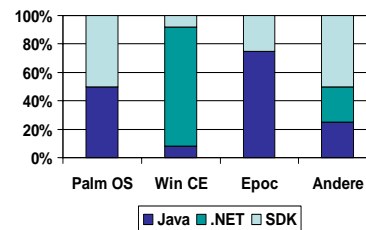
2001



2004



Mobile Web Services



© Siemens AG, T. Wieland, 27.01.2002

Quelle: Gartner

Im Trend: Web Services

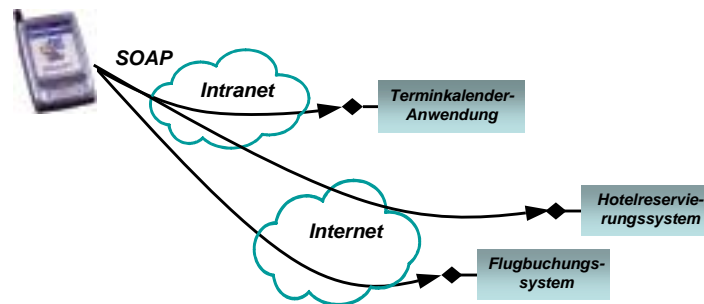
- Ein Web Service ist eine Anwendung, die es erlaubt, ihre Methoden über eine Web-Schnittstelle aufzurufen.
- Auf diese Weise können Informationen, die auf einem Webserver liegen, durch andere Programme genutzt werden.
- Der Aufruf der Methoden erfolgt mittels SOAP, die Beschreibung der Schnittstellen mittels WSDL, beides auf XML basierend.
 - D.h. Web Service = SOAP + WSDL
- Auch die Einbindung mobiler Endgeräte ist möglich.

Mobile Web Services

© Siemens AG, T. Wieland, 27.01.2002

Beispiel für Webservice

- Informationen aus Terminkalender können mit Flugplänen abgeglichen werden
- Wenn sich auf einer Seite etwas ändert, kann das an die andere weitergegeben werden



Mobile Web Services

7

© Siemens AG, T. Wieland, 27.01.2002

Vorteile des Web Service Paradigmas

- Infrastrukturdienste wie Benutzeridentifikation, Backup usw. können nahtlos von externen Quellen integriert werden.
- IT-Fachleute können sich auf das Geschäft konzentrieren statt auf Infrastruktur
- Durch die Verbindung von internen und externen Diensten können völlig neue Arten von Anwendungen entstehen
- Interoperabilität bisher heterogener Plattformen wird deutlich verbessert
 - Insbesondere kleine Endgeräte leicht anbindbar

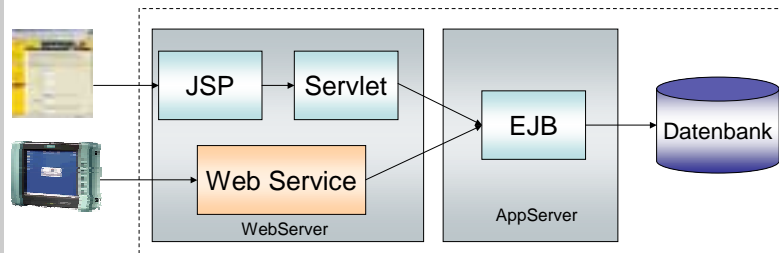
Mobile Web Services

8

© Siemens AG, T. Wieland, 27.01.2002

Wo kann ich Web Services heute nutzen?

- Viele Web-Sites bieten Eingaben über Formulare
- Alternativ zum Formular können die Eingaben auch als Web Service-Aufrufe akzeptiert werden
- Beispiel:
 - Paketdienst bietet Formular, Empfang einer Sendung zu bestätigen
 - Als Web Service kann eine Client-Anwendung auf einem mobilen Endgerät diese Infos direkt zum Server schicken



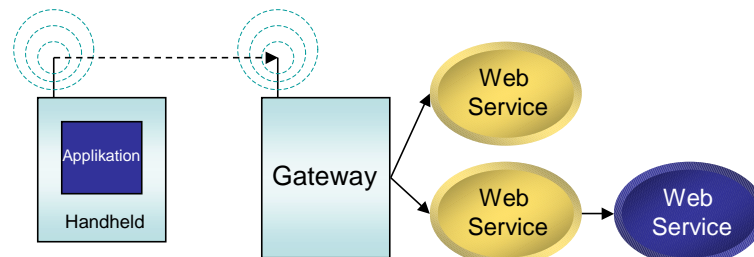
Mobile Web Services

9

© Siemens AG, T. Wieland, 27.01.2002

Nutzung von mobilen Web Services

- Anwendungen werden auf dem Endgerät installiert
 - Evtl. mittels automatischem Deployment (künftig z.B. WebStart oder .NET Component Download)
- Applikationen enthalten nur Koordinationslogik, um eine schnelle Reaktion auf Benutzerinteraktionen zu gewährleisten
- Business-Logik befindet sich in Form von Web Services im Netz verteilt



Mobile Web Services

10

© Siemens AG, T. Wieland, 27.01.2002

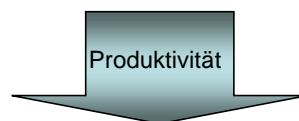
Implementierungen für mobile Geräte



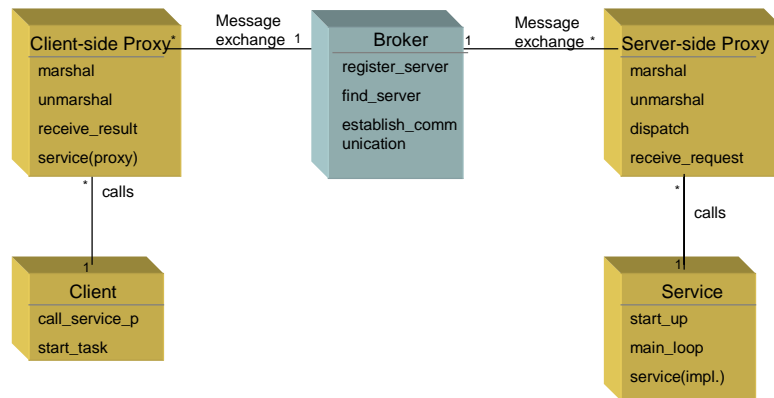
Zutaten für eine SOAP-Umgebung

- **Ein XML-Parser**
 - Dient dazu, die eintreffenden SOAP-Nachrichten auszuwerten
 - Geht auch selbstprogrammiert, ist dann aber nicht so flexibel
- **Ein Listener für den HTTP-Port**
 - Im allgemeinen Port 80
 - Fängt alle Nachrichten auf diesem Port ab
 - Wirft ggf. alle Nicht-SOAP-Nachrichten weg
- **Ein Client-Proxy**
 - Stellt eine Schnittstelle ähnlich wie der Server für lokale Zugriffe bereit
 - Packt die Aufrufe zu einer SOAP-Nachricht zusammen
 - Wertet die eintreffenden Antworten aus

Nachteil
bei Eigenbau:



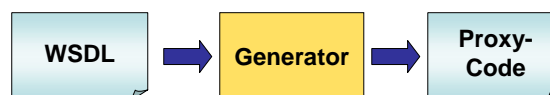
Das Broker-Pattern



Quelle: F. Buschmann et al.: "Pattern-oriented software architecture", Wiley, 1996

Effiziente Entwicklung braucht Generatoren

- **SOAP-Nachrichten müssen geparkt bzw. erzeugt werden**
 - "Marshaling" bzw. "De-Marshaling"
- **Sollte als Proxy-Teilkomponente gekapselt werden**
 - Proxy selbst zu schreiben ist möglich, aber sehr mechanische Aufgabe
- **SOAP allein reicht meist nicht für einen Web Service**
 - Client-Proxy aus bestehender WSDL-Beschreibung notwendig
- **Generatoren sind notwendig, um aus WSDL Client-Proxies oder aus Java/C++/...-Klassen Server-Proxies zu erzeugen!**



- **Daran sollte auch Qualität von Toolkits gemessen werden!**

Verfügbare Toolkits

- **Microsoft Mobile Internet Toolkit for .NET**
 - Bietet Web Forms, die auf dem Server laufen
- **.NET Smart Device Extensions**
 - Enthält das Compact Framework, das .NET für Kleingeräte
 - Derzeit noch im Beta-Stadium
- **PocketSoap**
 - Komplette Open-Source-SOAP-Implementierung für Pocket-PCs (z.B. Compaq iPaq) unter Windows CE
 - Wird auch von Microsoft empfohlen/verwendet
- **kSOAP**
 - Open-Source-Version für die J2ME, z.B. auf Mobiltelefonen



Software &
Engineering
Architecture

Microsoft Mobile Internet Toolkit

- **Zusatz für Visual Studio .NET**
 - Bringt „Mobile Web Forms“
 - Erkennt Endgerät, erzeugt WML 1.1, HTML 3.2 oder cHTML
- **Gleiches Konzept wie bei Web Forms**
 - Controls laufen auf dem Server (Attribut *runat="server"*)
 - Alle Aktionen über ASP.NET-Seiten
- **Web Services können aus diesen Seiten heraus aufgerufen werden**
 - Das mobile Endgerät ruft Web Services nicht direkt auf
 - Die Logik liegt nach wie vor auf dem Server
 - Entspricht einer zusätzlichen Indirektionsstufe zur Content-Adaption



Software &
Engineering
Architecture

Beispiel: Temperatur-Konverter °F -> °C

```
<%@ Page Codebehind="Tempconv.cs" Inherits="TempConv.TempPage"
Language="C#" %>
<%@ Register TagPrefix="mobile"
Namespace="System.Web.UI.MobileControls"
Assembly="System.Web.Mobile" %>

<mobile:Form runat="server">
  <mobile:Label runat="server">Temperature in °F</mobile:Label>
  <mobile:Textbox runat="server" id="TempEdit"/>
  <mobile:Command runat="server" OnClick="Respond" Text="Convert"/>
</mobile:Form >

<mobile:Form runat="server" id="ResponseForm">
  <mobile:Label runat="server" id="INewTemp">0</mobile:Label>
</mobile:Form>
```

Beispiel: Temperatur-Konverter °F -> °C

```
namespace TempConv
{
  public class TempPage : MobilePage
  {
    protected System.Web.UI.MobileControls.Label Label1;
    // ... Weitere Steuerelement-Deklaration
    protected float m_temp;

    protected void Respond(Object sender, EventArgs e)
    {
      m_temp = float.Parse(TempEdit.Text);
      ActiveForm = ResponseForm;
    }

    protected void CreateResponse(Object sender, EventArgs e)
    {
      localhost.TempCalc tc = new localhost.TempCalc();
      INewTemp.Text = "Temperature in °C: " +
        tc.FahrenheitToCelsius(m_temp).ToString();
    }
  }
}
```

Beispiel: Temperatur-Konverter °F -> °C



Mobile Web Services



19

© Siemens AG, T. Wieland, 27.01.2002



Software & Engineering Architecture

.NET Smart Device Extensions

- **Portable und kleine .NET CLR für kleine Geräte**
 - Enthält Compact Framework (verschranktes .NET Framework)
- **Bindet Visual Studio .NET ein**
 - Lässt "managed" Applikationen direkt ausführen
 - Debuggen mit Visual Studio .NET
 - Bietet Pocket-PC-Emulator
- **Arbeitet komplementär zum Betriebssystem**
 - Betriebssystem übernimmt Scheduling, Anzeige der Benutzeroberfläche, Eingabeverarbeitung, Ressourcenverwaltung etc.
- **Unterstützt über ADO.NET und SQL Server Datenhaltung mit unterbrochener Verbindung auf dem Gerät**
- **Setzt kein installiertes Compact Framework auf dem Endgerät voraus, sondern kopiert die Laufzeitumgebung beim Deployment gleich mit!**

Mobile Web Services

20

© Siemens AG, T. Wieland, 27.01.2002



Software & Engineering Architecture

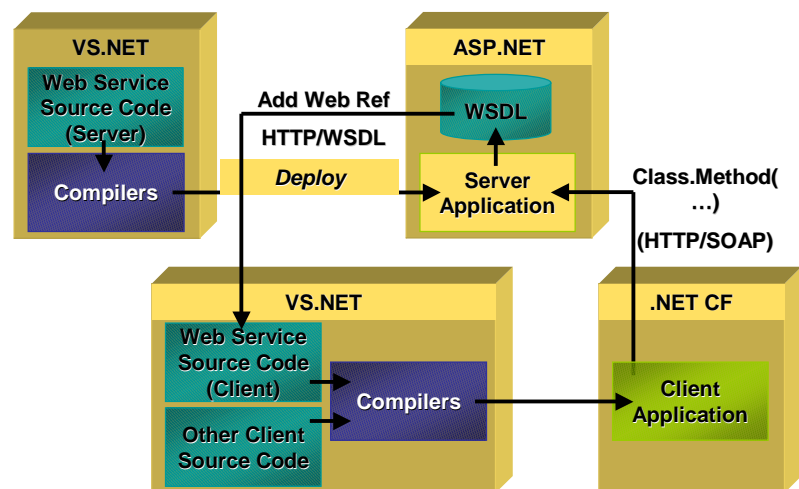
Web Service in SDE

- SDE unterstützt Web Services als Client
- Daten und Benutzeroberfläche lassen sich trennen
 - Ein "rich client" kann auf viele verschiedene Services zugreifen
- Trennung nach Publisher/Consumer Entwurfsmuster
- Benutzung von Web Services wie im Internet
- Client-Anwendung kann Paketierung von Datenübertragungen festlegen
- .NET CF arbeitet mit dem vom Visual Studio .NET erzeugten Proxy-Code
 - Sowohl synchrone als auch asynchrone Schnittstellen werden unterstützt



Software &
Engineering
Architecture

Entwicklung von Web Services



PocketSOAP

- **Open-Source-Projekt unter MPL (Mozilla-Lizenz), entwickelt von Simon Fell**
 - Wird als COM-Komponente installiert, steht so allen Anwendungen zur Verfügung
 - Am einfachsten aus VB anzusprechen, über COM-Interop auch aus .NET (CF)
 - Verwendet den Expat XML-Parser
- **Große Feature-Menge; PocketSOAP (1.2) unterstützt**
 - Einfache Datentypen, base64-Binärdaten, Arrays, multi-dimensionale Arrays, Teil- und dünnbesetzte Arrays, komplex Typen uva.
 - SOAP-Headers
 - HTTP einschließlich Proxies, Authentifizierung, Proxy-Authentifizierung und SSL
- **Läuft unter Windows 9x/ME/2000/XP und CE 3.0/PocketPC 2002**
 - Unterstützt StrongARM-, SH3- und MIPS-Architekturen
- **Weitere Tools vom gleichen Autor verfügbar**
 - WSDL-Proxy-Generator
 - SOAP Server Services
 - Proxy Trace Server, um SOAP-Nachrichten zu tracen



Software &
Engineering
Architecture

Einfache Aufrufe aus VB-Skript

```
dim envelope as PocketSOAP.Envelope
set envelope = CreateObject("PocketSOAP.Envelope")
envelope.methodName = "getQuote"
envelope.URI = "urn:xmethods-delayed-quotes"
envelope.Parameters.Create "symbol", "MSFT"
```

```
dim transp as PocketSOAP.HTTPTransport
set transp = CreateObject("PocketSOAP.HTTPTransport")
transp.Send "http://services.xmethods.net/soap", envelope.serialize
envelope.Parse transp
wscript.echo e.Parameters.Item(0).Value
```



Software &
Engineering
Architecture

SOAP-Server auf dem PocketPC

- **Durch einfache Mittel auch ein SOAP-Server auf einem PocketPC möglich**

- WinSock-Control lauscht auf Port 80
- Listener empfängt und sendet HTTP-Nachrichten
- SOAP-Parser verwendet MSXML
- Response-Funktion baut einfachen Antwort-String zusammen

- **Siehe dazu Christian Forsbergs Artikel**

- <http://www.microsoft.com/MOBILE/developer/technicalarticles/hostingweb.asp>

- **Viele mögliche Anwendungen**

- Kontext-Information über den Benutzer (Büro, Besprechung, Kundenbesuch)
- Statusabfrage
- ...



Software &
Engineering
Architecture

kSOAP



- **Entstanden im Enhydra-Projekt**

- Open-Source-Software unter EPL
- Mittlerweile als Version 0.99 verfügbar (Dez. 01)

- **Untermenge von SOAP 1.1**

- Unterstützung für Datentypen wie große Dezimalzahlen oder Base64-codierte Byte-Arrays kann hinzugefügt werden
- Multi-dimensionale Arrays werden nicht unterstützt.
- Gute Interoperabilität mit anderen Implementierungen

- **Entwickelt für die Java 2 MicroEdition (J2ME)**

- Besonders für Kleinstgeräte wie Mobiltelefone
- Über einen API-Umsetzer (ME2SE) auch mit der Standard-Edition verwendbar



Software &
Engineering
Architecture

Beispiel mit kSOAP

```

HttpTransport transport = new HttpTransport ("http://212.99.131.154:7300/",
      "\http://www.shinkatech.com/Currency/CurrencyConverter.calculateExchangeRate\");

SoapObject convert = new SoapObject (serviceNamespace, "convert");

convert.addProperty ("currency", fromField.getString ());
convert.addProperty ("amount", new Float (amountField.getString ());
convert.addProperty ("toCurrency", toField.getString ());

SoapObject request = new SoapObject (serviceNamespace, "calculateExchangeRate");
request.addProperty ("convert", convert);

ClassMap classMap = new ClassMap();
classMap.prefixMap = new PrefixMap (classMap.prefixMap, "m",
      "http://www.shinkatech.com/CurrencyConverter/message/");
transport.setClassMap (classMap);

String result = transport.call (request).toString ();
result = result.substring(0,result.indexOf(".")+3);

resultItem.setLabel ("amount :");
resultItem.setText (" "+result);

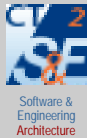
```

Quelle: enhydra.org

Mobile Web Services

27

© Siemens AG, T. Wieland, 27.01.2002



Software &
Engineering
Architecture

SOAP auf dem Linux PDA

- **Agenda VR 3 läuft mit Linux**
 - Eigene Linuxanpassung (basierend auf linux-vr.org), MIPS-Prozessor
 - Konnektivität mittels serieller oder IR-Schnittstelle bzw. eingebautem Modem
 - Graphische Oberfläche mit Fast Light Toolkit (ftk.org)
- **C++-Entwicklung mit GCC als Cross-Compiling**
- **Rund 3 MB Platz im Dateisystem des VR 3**
- **Für SOAP wird libxml2 eingesetzt**
 - Aus Gnome-Projekt (xmlsoft.org)
 - Schneller und komfortabler Parser
 - kleiner Footprint
 - enthält sogar HTTP-Sender und -Listener



Software &
Engineering
Architecture

Mobile Web Services

28

© Siemens AG, T. Wieland, 27.01.2002

Beispiel: Request erzeugen

```
xmlDocPtr getRequest(xmlChar* term, xmlChar* srcLang, xmlChar* trgLang) {
    xmlDocPtr doc;
    xmlNodePtr tree, subtree, subsubtree;
    xmlNsPtr SOAP, xsd, xsi, ns1;
    doc = xmlNewDoc((xmlChar*)"1.0");
    doc->children = xmlNewDocNode(doc, NULL, (xmlChar*)"Envelope", NULL);
    SOAP = xmlNewNs(doc->children, (xmlChar*)"http://schemas.xmlsoap.org/soap/envelope/",
        (xmlChar*)"SOAP-ENV");
    xmlSetNs(doc->children, SOAP);
    tree = xmlNewChild(doc->children, SOAP, (xmlChar*)"Body", NULL);
    subtree = xmlNewChild(tree, NULL, (xmlChar*)"getTranslations", NULL);
    ns1 = xmlNewNs(subtree, (xmlChar*)"urn:demo1:translate", (xmlChar*)"ns1");
    xmlSetNs(subtree, ns1);
    xmlSetNsProp(subtree, SOAP, (xmlChar*)"encoding-style",
        (xmlChar*)"http://schemas.xmlsoap.org/soap/encoding/");

    subsubtree = xmlNewChild(subtree, NULL, (xmlChar*)"term", term);
    xmlSetNs(subsubtree, NULL);
    xmlSetNsProp(subsubtree, xsi, (xmlChar*)"type", (xmlChar*)"xsd:string");

    subsubtree = xmlNewChild(subtree, NULL, (xmlChar*)"srcLanguage", srcLang);
    xmlSetNs(subsubtree, NULL);
    xmlSetNsProp(subsubtree, xsi, (xmlChar*)"type", (xmlChar*)"xsd:string");

    subsubtree = xmlNewChild(subtree, NULL, (xmlChar*)"targetLanguage", trgLang);
    xmlSetNs(subsubtree, NULL);
    xmlSetNsProp(subsubtree, xsi, (xmlChar*)"type", (xmlChar*)"xsd:string");
    return doc;
}
```

Quelle: K. Banke: "Webservices auf dem PDA von Agenda", Linux Enterprise, 12/2001

Wozu braucht man mobile Web Services?

Erfolgskriterien für mobile Web Services

- **Mobile Web Services sollten den Kontext des Benutzers verwenden**
 - Beim Zusammenspiel verschiedener Services sollte Kontextinformation übertragen werden
- **Infrastrukturservices sollten global zur Verfügung stehen**
 - Wie bei EJBs werden auch für Web Services Dienste für Sicherheit, Persistenz und Transaktionen benötigt
- **Web Services sollten den Workflow des Geschäfts abbilden**
 - Mobile Web Services müssen auch die mobilen Möglichkeiten nutzen
 - Nicht nur einen anderen Zugriffsweg bereitstellen!
- **Existierende Systeme müssen integriert werden**
 - Lokal und im Backend ist Standardmiddleware (J2EE, CORBA 3, COM+) viel effizienter
 - Konnektoren müssen möglichst automatisch generierbar sein

Erfolgskriterien für mobile Web Services (2)

- **Mobile Web Services müssen sich schrittweise vom Client/Server-Modell lösen**
 - Web Services können auch für Peer-to-Peer (P2P)-Lösungen eingesetzt werden
 - Damit ist ein automatischer Informationsabgleich zwischen mobilen Endgeräten möglich
- **Verfügbarkeit und Wartbarkeit muss gewährleistet sein**
 - Anwendungen, die Web Services benötigen, sind auf deren permanente Verfügbarkeit angewiesen
 - Neue Stufe der gegenseitigen Abhängigkeit (Service Level Agreements erforderlich!)
 - Betreiber von Services müssen auf Ausfallsicherheit und Lastverteilung achten, ggf. auch auf Quality of Service (Streaming)
 - Versionierung muss Abwärtskompatibilität sicherstellen
- **Entwicklung muss einfach sein, nicht schwieriger als für den Server**
 - Dazu brauchen wir flexible Entwicklungsumgebungen (unabhängig von Produktfamilien) mit Generatoren und Debugging-Möglichkeiten

Herausforderung: Context awareness

- **Fähigkeit eines Dienstes, sich an einen (verändernden) Kontext selbstständig anzupassen**
- **Definition (nach Dey und Abowd):**
 - "Kontext ist eine Information, welche die Situation einer Entität beschreibt. Die Gesamtheit aller Konfigurationsmerkmale dieser Entität ist ihr Kontext.
 - Ein System heißt 'context aware', wenn es den Kontext verwendet, um relevante Informationen und/oder Dienste dem Benutzer zur Verfügung zu stellen, wobei die Relevanz von der Rolle des Benutzers abhängt."



Bezug des Kontextes

- **Kontext kann sich auf vielerlei Eigenschaften beziehen**
 - Ort: absolut oder relativ zu einem anderen Dienst/Endgerät
 - Bandbreite/Netz: z.B. Umschaltung zwischen GSM/GPRS, UMTS und WLAN
 - Benutzerparadigma: Desktop vs. Mobiltelefon
 - Unbekannte Zielplattformen
 - Rolle des Benutzers: Kunde, Reporter, Projektleiter, Familienvater etc.
 - Vorlieben/Interessen des Benutzers



Herausforderung: Sitzungsmanagement

- **Sitzungen können unterbrochen werden**
 - Benutzer verlässt Empfangsbereich des Netzes (und betritt evtl. Bereich eines anderen Netzes)
 - Endgerät wird abgeschaltet
 - Durch Benutzer: Telefonat, Besprechung, Ereignis in Umgebung
 - Aufgrund von Energiemangel
 - Benutzer wechselt Endgerät
 - Mobiltelefon ⇔ PDA ⇔ Desktop-PC
- **Daher muss der Server aktiv Sitzungen verwalten**



Anwendungsszenarien

- **Routenplanung**
 - Planungssoftware kann Wetter- und Stauinformationen als Web Service nachfragen und in die Planung einbeziehen
- **Krankenhausvisite, Notarzteinsatz**
 - Arzt kann auf Patientendaten zugreifen
- **Mobile Anlagenwartung**
 - Gesamtstatus der Anlage, letzte Kalibrierdaten etc. können vom Service-Techniker herangezogen werden
- **Vertreter auf Kundenbesuch**
 - Aktuelle Rabattsätze, Außenstände etc. können für Angebotserstellung verwendet werden
- **Spiele**
 - Interaktion mit mehreren anderen Benutzern
- **Spontane P2P-Kommunikation**
 - Austausch von Adressen, Interessen etc., z.B. auf Messen



Zusammenfassung



Fazit

- **Web Services eignen sich auch für mobile Endgeräte**
 - Sie stellen neue Herausforderungen bezüglich
 - Netzanbindung
 - Kontextabhängigkeit
 - Sitzungsverwaltung
- **Die aktuellen Toolkits (PocketSOAP, kSOAP) unterstützen Web Services erst rudimentär**
 - Unzureichende WSDL-Integration
 - Keine Proxy-Generierung
- **SOAP-Server auf Endgeräten nur elementar möglich**
 - Bisher nicht in Programmierumgebungen enthalten
 - Aber: Großes Potenzial für Enterprise-Einbindung und P2P

Links

- **Microsoft**

- Web Services: <http://msdn.microsoft.com/webservices>
- Mobile Web Forms: <http://msdn.microsoft.com/vstudio/nextgen/technology/mobilewebforms.asp>
- PocketSOAP: <http://www.microsoft.com/mobile/developer/technicalarticles/pocketsoap.asp>
- Smart Device Extensions: <http://msdn.microsoft.com/vstudio/nextgen/device/smartdev.asp>

- **PocketSOAP**

- <http://www.pocketsoap.com>

- **kSOAP**

- <http://ksoap.enhydra.org>

- **Phone Emulator**

- http://www.openwave.com/products/developer_products/sdk/



Software &
Engineering
Architecture

Links (2)

- **Agenda Handheld**

- <http://www.agendacomputing.com>
- Linux VR: <http://www.linux-vr.org>
- Fast Light Toolkit: <http://www.fltk.org>
- XMLLIB: <http://www.xmlsoft.org>

- **Siemens CT**

- <http://www.ct.siemens.de>
- <http://www.plug-n-play-technologies.com>



Software &
Engineering
Architecture

Links (3)

- **“Towards a better understanding of context and context-awareness”**
 - Anind Dey and G.D. Abowd. Techn. Report, GeorgiaTech, 1998.
- **“Context-Aware Computing Applications”**
 - Bill N. Schilit, Norman I. Adams, and Roy Want. In Proceedings of the Workshop on Mobile Computing Systems and Applications, Santa Cruz, CA, December 1994. IEEE Computer Society.
<ftp://ftp.parc.xerox.com/pub/schilit/wmc-94-schilit.ps>

Danke für Ihre Aufmerksamkeit

Fragen oder Anmerkungen?



Download der Folien: <http://www.dr Wieland.de>