„The wireless telegraph is not difficult to understand. The ordinary telegraph is like a very long cat.

You pull the tail in New York, and the cat meows in Los Angeles.

The wireless is the same, only without the cat.“

Albert Einstein

ITSD Consulting GmbH

Anbindung an Serversysteme

Frank Schlinkheider • Enterprise Web & Mobile Solutions • fs@itsd-consulting.de

Worum geht es?

HTTP/HTTPS

Optionale Packages

Serialisierung

CLDC

Middleware

JMS-Provider

Transaktionen

JNDI

?

SAP

JMS

SOAP/XML

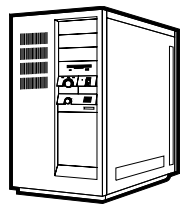
JSR 172

Webservices

CDC

UDDI

Sessionmanagment



J2EE

Agenda

- **Kurze Einführung in Java 2 Micro Edition**
  - Besonderheiten
  - Probleme

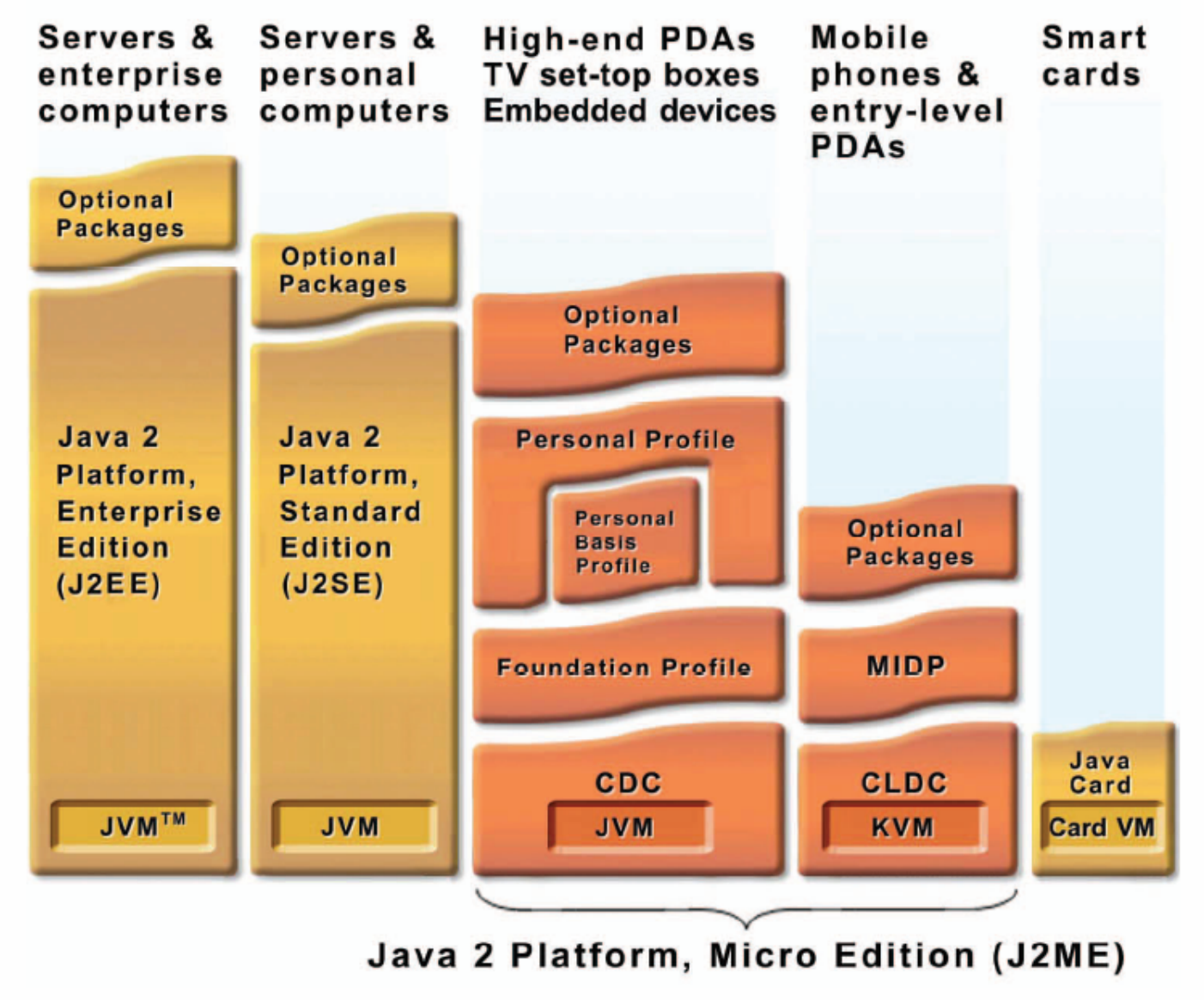
- Anbindungsmöglichkeiten
  - HTTP, kSOAP, J2ME Web Services (JSR 172), jtom

- Beispiele: kSOAP und jtom

- Live-Demo

- Zusammenfassung und Diskussion

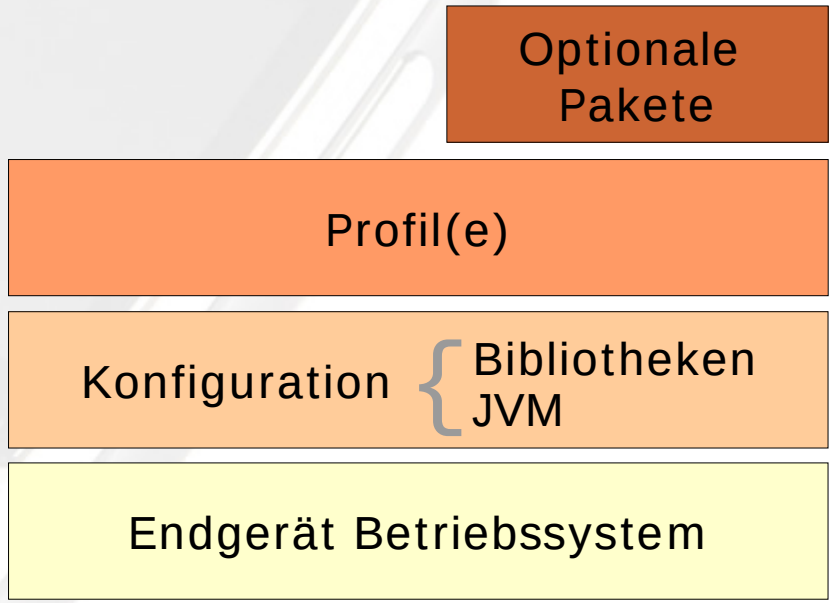
J2ME Einführung - Die Java Familie



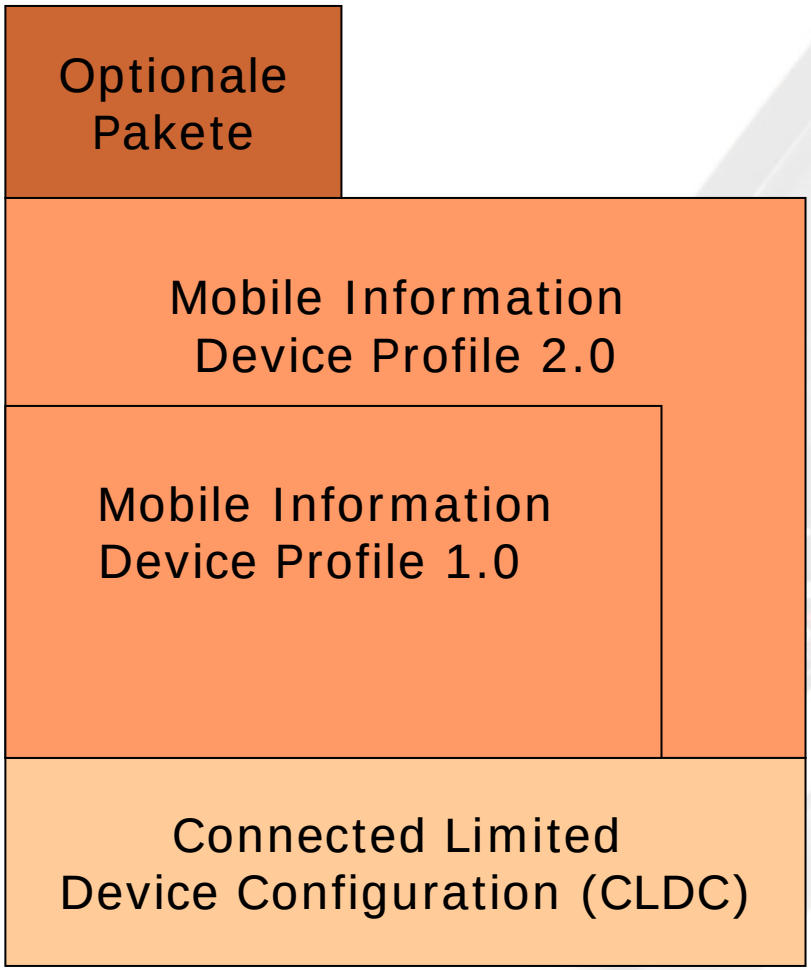
J2ME Einführung – Überblick

- mobile und kleinere Geräte
- ***Konfiguration***
  definieren der Funktionalität
- ***Profil(e)***
  zusätzliche Funktionalität für diese Segmente
- ***Optionale Pakete***
  erweiternde Funktionalität

- In Summe => **J2ME API**



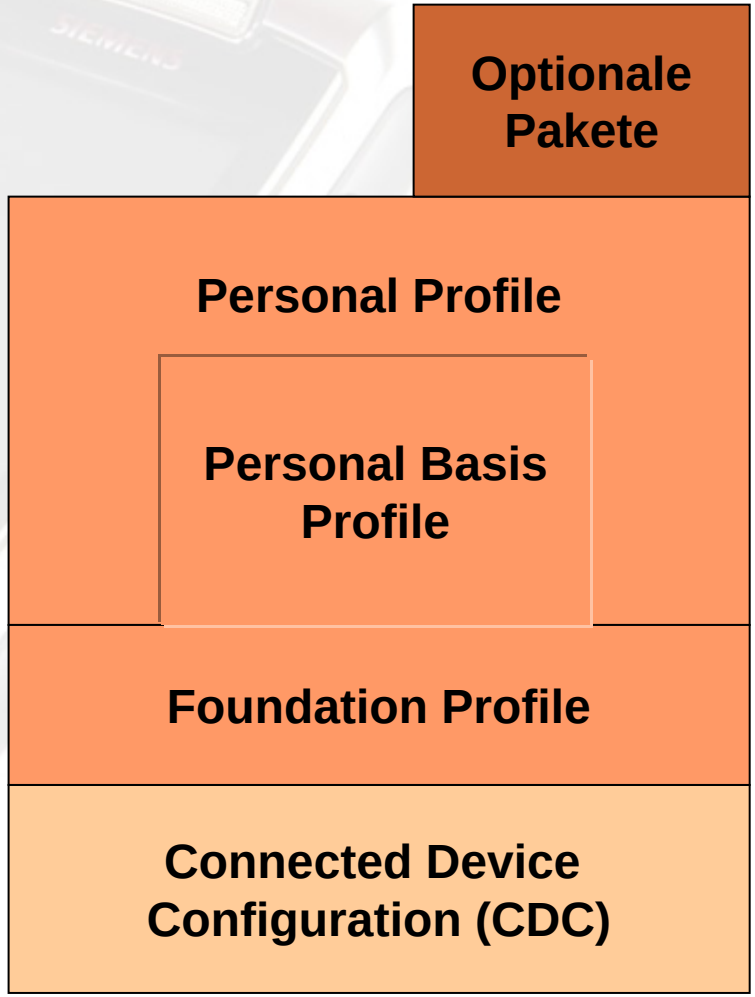
J2ME Einführung – CLDC/MIDP



- **MIDP 1.0**
  - -HTTP
  - Record Management System
  - Benutzungsschnittstelle:
    - portable High Level API
    - direkte Low Level API
- **MIDP 2.0**
  - MIDP 1.0
  - HTTPS
  - Push Registry
- **Optionale Pakete**
  - Bluetooth
  - Webservices
  - SMS (WMA)
  - etc.
- **Schlanke VM (ca. 80 KByte)**

J2ME Einführung – CDC + Profile

- **Foundation Profile**
  - ohne grafische Schnittstelle
  - Optimierte Basisklassen
- **Personal Basis Profile**
  - ohne volle AWT Unterstützung
  - unterstützt xlet Programmiermodell
- **Personal Profile**
  - AWT und
  - Applet Programmiermodell
- **Optionale Pakete**
  - J2ME RMI
  - JDBC



J2ME Einführung – Anbindungsprobleme

- Nur HTTP/HTTPS
- Keine einheitliche Zugriffsmöglichkeit
- Kein JNDI
- Kein RMI
- Keine asynchrone Kommunikation
- Kein Messaging

Zudem

- Unzuverlässige Netzverbindungen
- Keine standardisierte mobile Middleware

Lösung?

Agenda

- Kurze Einführung in Java 2 Micro Edition
  - Besonderheiten
  - Probleme
- **Anbindungsmöglichkeiten**
  - **HTTP, kSOAP, J2ME Web Services(JSR 172), jtom**
- Beispiele: kSOAP und jtom
- Live-Demo
- Diskussion

Anbindungen – HTTP/HTTPS

- Request/Response
- Keine Serialisierung
- Keine Transaktionen
- Session-Management
- Kein Messaging
- Keine „direkte“ Anbindung an Serversysteme

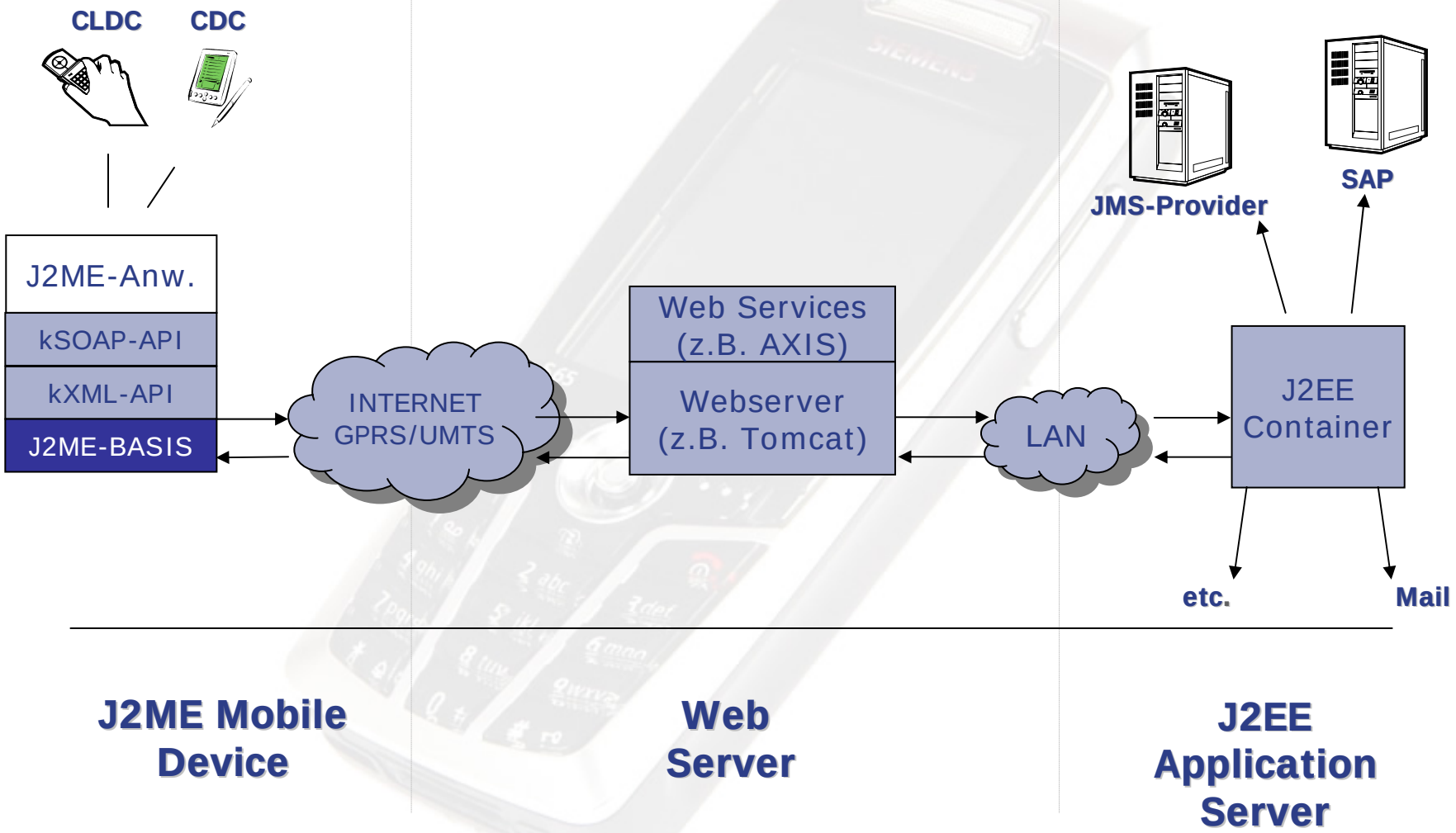
-> für einfachste Anbindung geeignet.

Anbindungen – kSOAP

- Webservice für J2ME
- Open Source
- CDC(J2SE)/CLDC
- Client-Bibliothek < 40 KByte
- Serialisierung
- Kein UDDI
- Keine Transaktionen
- Kein Sessionmanagement
- Keine SOAP Messages nach SOAP 1.1 Encoding
- Keine „direkte“ Anbindung

-> einfache kostengünstige Lösung

Anbindungen – Architekturbeispiel kSOAP

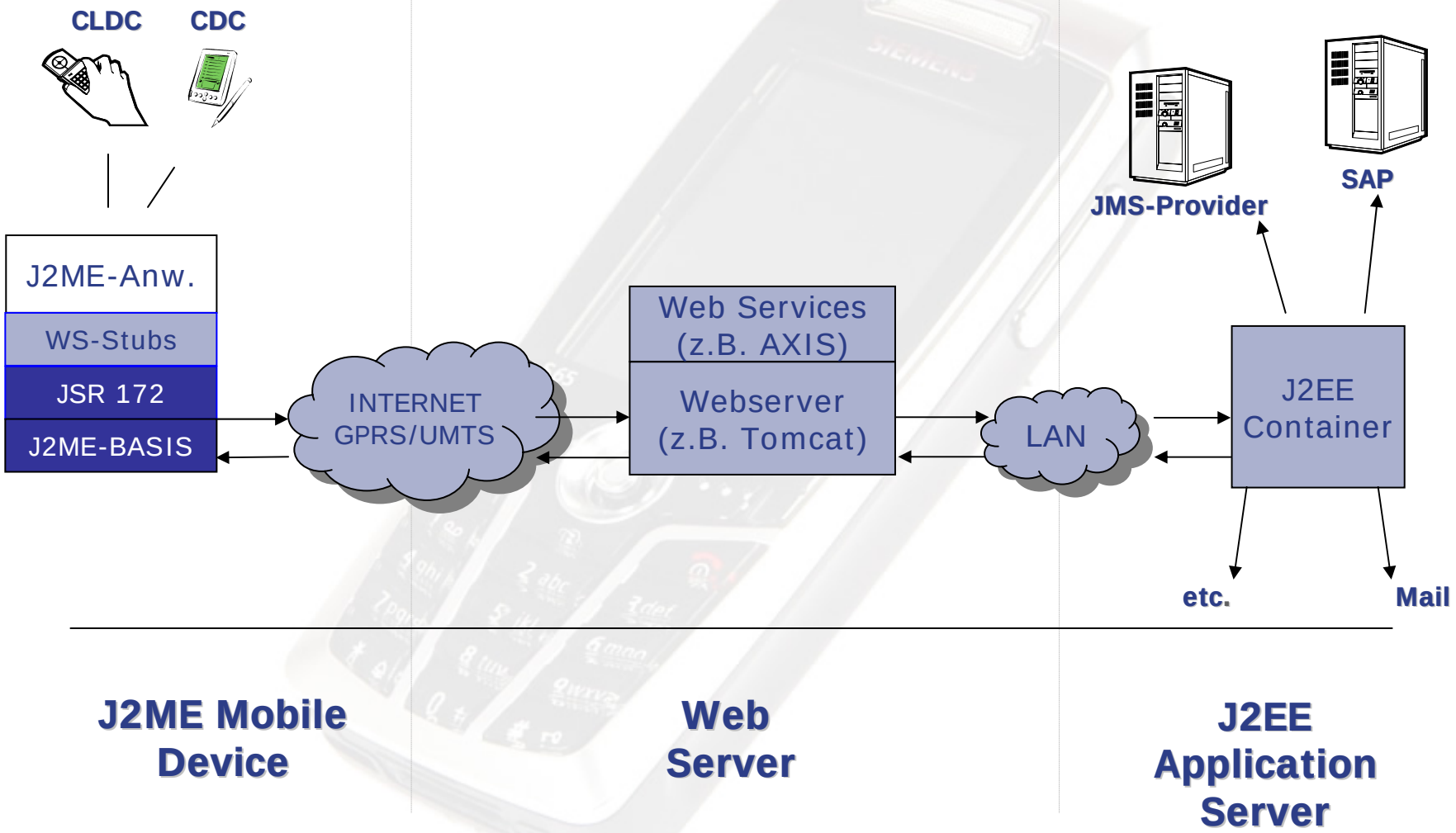


Anbindungen – J2ME Webservice API (JSR172)

- Webservices für J2ME (JSR 172)
- Optionales Package!
- CDC/CLDC
- SOAP 1.1
- Stub-Generator
- Serialisierung
- Keine Transaktionen
- Sessionmanagement
- Keine SOAP Messages mit SOAP 1.1 Encoding
- Kein UDDI
- Keine „direkte“ Anbindung

-> gute Lösung, aber nicht für alle Endgeräte vorhanden

Anbindungen – Architekturbeispiel JSR 172

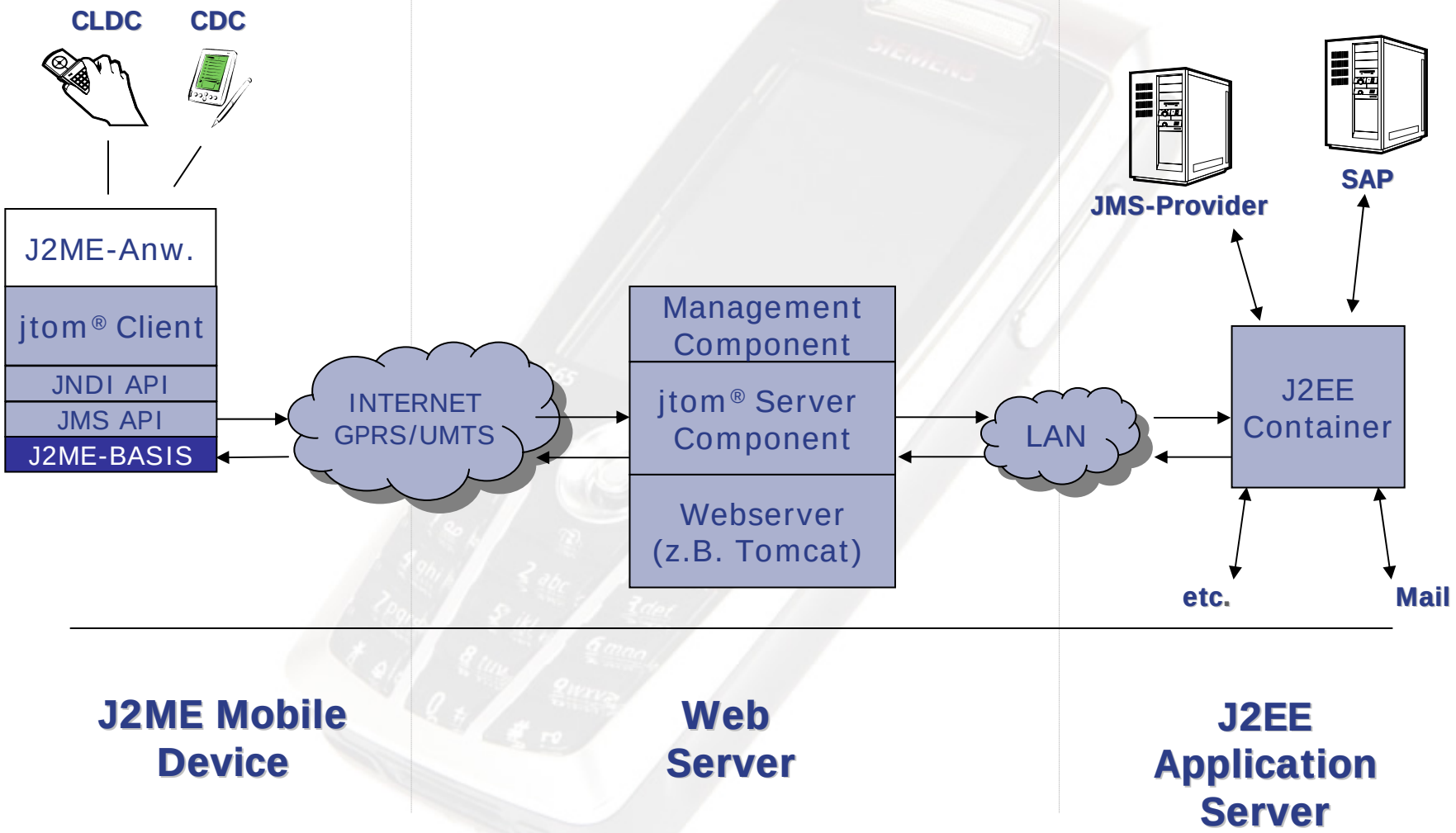


Anbindungen – mobile Middleware jtom

- JNDI und JMS
- Client-Package < 50 KByte (für JNDI < 40 KByte)
- Effiziente Serialisierung
- Transaktionsmanagment
- Sessionmanagement
- Messaging
- „direkte“ Anbindung an Serversysteme
- Unterstützung von CDC/CLDC
- Sockets/HTTP/HTTPS
- JBoss, IBM WebSphere, BEA WebLogic
- Unterstützung jeglicher VM

-> gute Lösung, kommerzieller Einsatz: Lizenzkosten

Anbindungen – mobile Middleware jtom



Agenda

- Kurze Einführung in Java 2 Micro Edition
  - Besonderheiten
  - Probleme
- Anbindungsmöglichkeiten
  - HTTP, kSOAP, J2ME Web Services (JSR 172), jtom
- **Beispiele: kSOAP und jtom**
- Live-Demo
- Diskussion

Anbindungen – kSOAP Example

- Es soll eine einfache Preisinformation eingeholt werden.
- Serverklasse „sample.ksoap.PriceService“
- WSDD (Deployment für AXIS)
- Clientklasse „sample.ksoap.PriceJ2MEService“

Anbindungen – kSOAP Webservice

```
package sample.ksoap;

public class PriceService {

   public String getPrice(String number) {

        //Return the price
        if (number.equals("1"))
                return new String(1500.50f);
        else
                return new String(2000.00f);
   }
}
```

Anbindungen – kSOAP Deployment

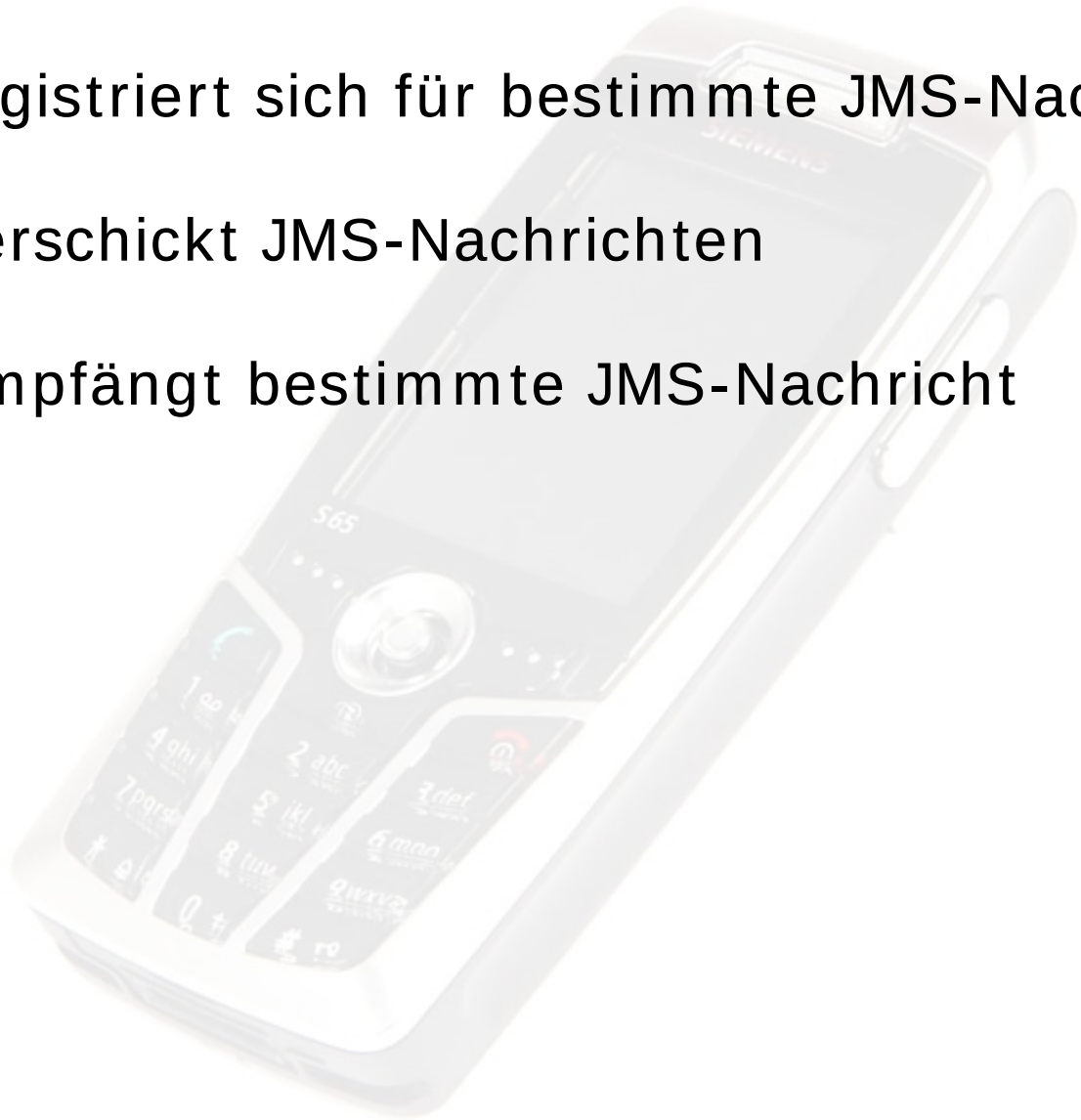
```
<deployment xmlns="http://xml.apache.org/axis/wsdd/"
   xmlns:java="http://xml.apache.org/axis/wsdd/providers/java">

  <service name="PriceService" provider="java:RPC">
      <parameter value="sample.ksoap.PriceService" name="className"/>
      <parameter value="getPrice" name="allowedMethods"/>
  </service>
</deployment>
```

Anbindungen – kSOAP Client

```
...
   try {
       String symbol = "1";
       SoapObject rpc = new SoapObject("urn:PriceService","getPrice");
       rpc.addProperty ("price", symbol);

       SoapSerializationEnvelope envelope =
                       new SoapSerializationEnvelope
                           (SoapEnvelope.VER10);
       envelope.bodyOut = rpc;

       HttpTransport ht = new HttpTransport
                   ("http://localhost:80/axis/services/PriceService");

       ht.call("urn:PriceService#getPrice",envelope);
       return envelope.getResult();
   }
   catch (Exception e) {
...
```

Anbindungen – jtom Example

- Client registriert sich für bestimmte JMS-Nachrichten
- Client verschickt JMS-Nachrichten
- Client empfängt bestimmte JMS-Nachricht



Anbindungen – jtom – Registrierung

```
/* register a MessageListener with a message provider */
public void registerReceiver(String url) {
  try {
    // open connection to JMS-server
    MessageConnection mConn = (MessageConnection)Connector.open(url);

    // "this" is registered as MessageListener for JMS-messages
    // When receiving a message the method
    // "this.notifyIncomingMessage()" is called
    mConn.setMessageListener(this, "userid = '4711'",true);

    mConn.setExceptionListener(this);
  } catch (java.io.IOException ioEx) {
      // handle Exceptions as required
  }
}
```

Anbindungen – jtom – URL

```
// req: jtom protocol identification
    String url = "wma2jms://"

// req: connection url for service
    +"http://localhost:8080/jtomServer/MIDPService;"

// req: key for accessing WMA-to-JMS mapping bean
    +"jtom/Connection;"

// req: JMS connection factory
    +"factory=ConnectionFactory;"

// req: message destination name
    +"name=testTopic;"

// req: used messaging model (topic|queue)
    +"type=TOPIC;"

// opt: JMS user and password
    +"username=john;password=needle";
```

Anbindungen – jtom – JMS senden

```
/* Send JMS-messages. */
public void sendMessage(String message, String url) {
  try {
    // open connection to JMS-server
    MessageConnection mConn = (MessageConnection)Connector.open(url);

    // create new TextMessage
    TextMessage msg = (TextMessage)mConn.
      newMessage(MessageConnection.TEXT_MESSAGE);

    // set Property
    msg.setMessageProperty("userid", "4711");
    // set text to send
    msg.setPayloadText(message);
    // send message
    mConnection.send(msg);

  } catch (java.io.IOException ioEx) {
    // handle Exceptions as required
  }
}
```

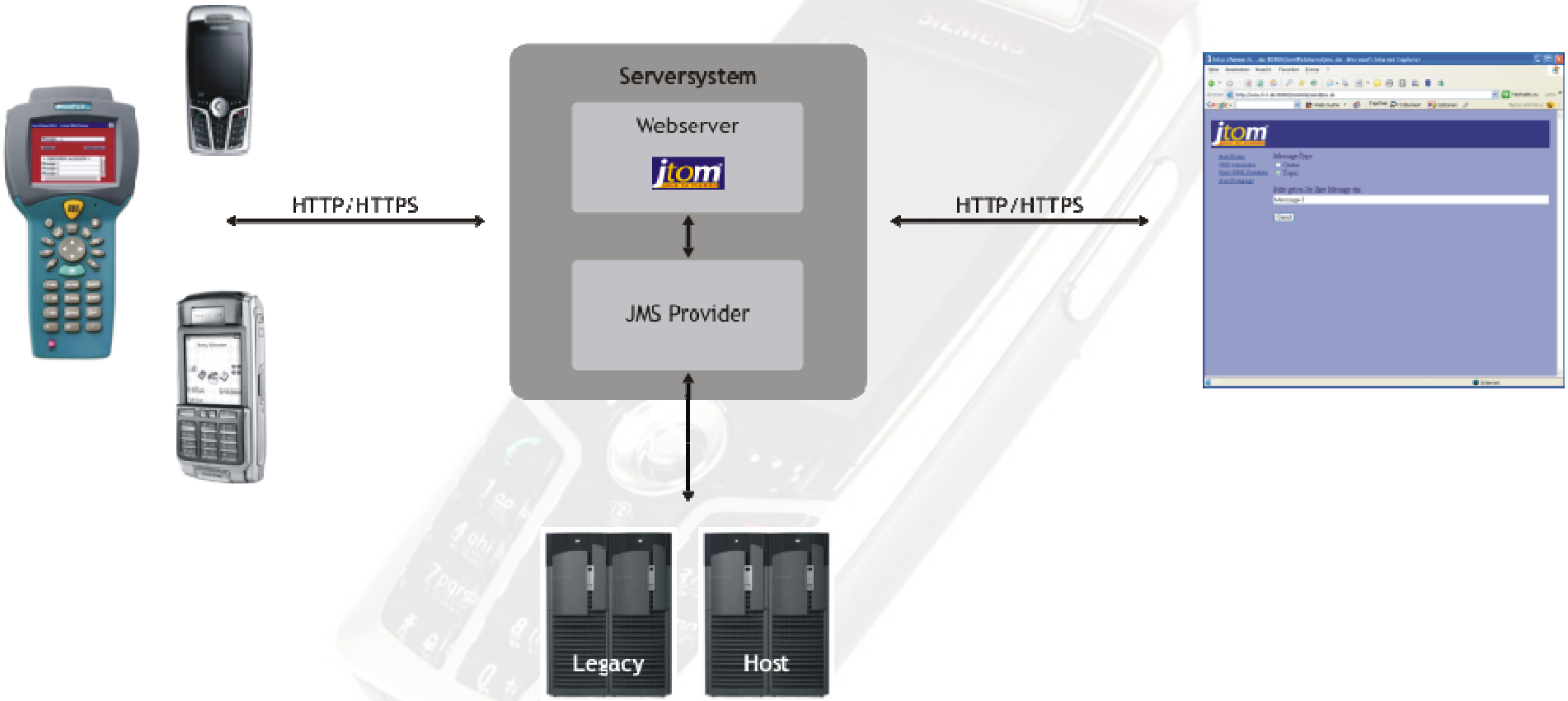
Anbindungen – jtom – JMS empfangen

```
/* This method is called when a message arrives. */
public void notifyIncomingMessage(MessageConnection conn) {
  try {
    // fetches message from connection
    Message msg = (Message) conn.receive();

    // type based handling of messages
    if(msg instanceof BinaryMessage) {
      // BinaryMessage
      byte[] bytes = ((BinaryMessage )msg).getPayloadData();
      // do other stuff with the payload here

     } else {
       // TextMessage
       String string = ((TextMessage )msg).getPayloadText();
       // do other stuff with the payload here
     }
   } catch (java.io.IOException ioEx) {
     // handle Exceptions as required
   }
}
```

Agenda

- Kurze Einführung in Java 2 Micro Edition
  - Besonderheiten
  - Probleme
- Anbindungsmöglichkeiten
  - HTTP, kSOAP, J2ME Webservice (JSR 172), jtom
- Beispiele: kSOAP und jtom
- **Live-Demo**
- Zusammenfassung und Diskussion

jtom Live-Demo



Zusammenfassung

- HTTP/HTTPS
  - immer vorhanden
  - aufwendige Umsetzung
  - keine direkte Anbindung an das Backendsystem
- kSOAP
  - klein und flexibel
  - Open Source
  - erhöht die Client-JAR-Größe
  - CDC und CLDC
- J2ME Web Services (JSR 172)
  - optionales Package
  - CDC und CLDC
  - Stub-Generator
- jtom
  - direkte Verbindung
  - JNDI/JMS
  - Messaging zum J2ME-Client
  - Sessionmanagement
  - effiziente Serialisierung

Zusammenfassung

- Verwenden Sie bereits Webservices im Unternehmen?
- Benötigen Sie ein Sessionmanagement für die Anbindung Ihrer mobilen Endgeräte?
- Können Sie die zu benutzenden Endgeräte vorgeben?
- Kennen Sie alle zu benutzenden Endgeräte?
- Benötigen Sie eine JMS-Funktionalität für Ihre Applikation?
- Ist die Netzbandbreite gering? (GPRS oder GSM)

Wichtige URLs

- J2ME Web Services API (JSR 172)
  http://www.jcp.org/en/jsr/detail?id=172
- kSOAP
  http://www.ksoap.org
- J2ME
  http://developers.sun.com/techtopics/mobility
- Java Web Services
  http://java.sun.com/webservices
- jtom
  http://www.jtom.de
- SOAP v1.1
  http://www.w3.org/TR/SOAP
- WebServices AXIS
  http://ws.apache.org/axis

ITSD Consulting GmbH

Vielen Dank!

Frank Schlinkheider • Enterprise Web & Mobile Solutions
fs@itsd-consulting.de
www.itsd-consulting.de und **www.jtom.de**