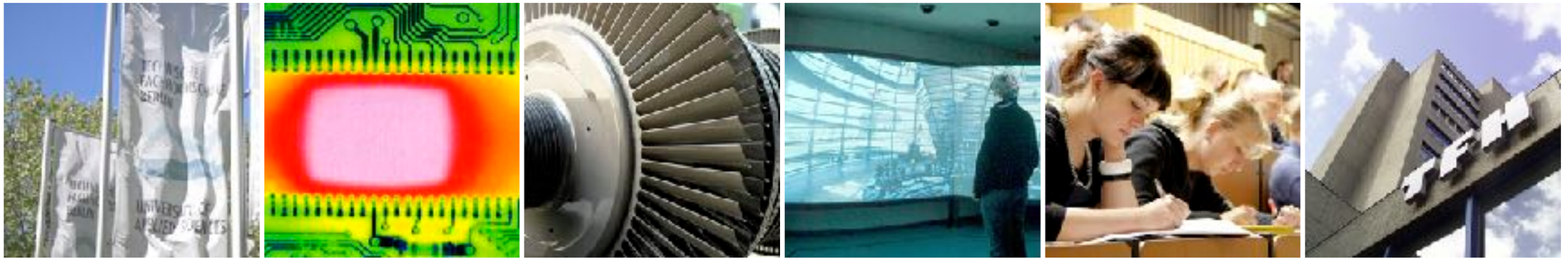
BEUTH HOCHSCHULE FÜR TECHNIK BERLIN
University of Applied Sciences

Summer School
Offene Gebäudeautomation

WebServices, Application Server und weitere Konzepte

aus: Distributed Systems , A. Schill, TU Dresden

EMR



SOAP (Simple Object Access Protocol)

- Protokoll für entfernte Objectaufrufe
- Entwicklung von Microsoft, IBM Lotus und anderen Partnern; Standardisierung durch IETF (Internet Engineering Task Force)
- Kodierung von Aufrufen und Parametern mittels XML (eXtensible Markup Language)
- Realisierung über HTTP, daher über Firewalls übertragbar; speziell für Internet geeignet (im Intranet dagegen kein Firewall-Problem)
- Neue Sicherheitsmechanismen auch auf Basis von XML nutzbar (XML Encryption; XML Signatur)

SOAP

- Unabhängig von speziellen Programmiersprachen, jedoch mittels Mappings von Java, C++ etc. nutzbar
- Aufgrund von Einbettung in HTTP weniger effizient als direkte Kommunikation wie bei RMI / IIOP / .NET
- Keine Referenzparamter, keine automatische Speicherverwaltung (Ziel ist Beschränkung auf minimale Funktionalität)
- Für synchrone Aufrufe auf Objekten nutzbar, aber auch asynchrone Interaktionen unterstützt (z.B. Message Passing)

SOAP: Beispiel Methodenaufruf

```
POST /BankServer HTTP/
Host: www.bank.de
Content-Type: text/xml
Content-Length: nnnn
SOAPMethodNeme: Some-Namespace-URI#getBalance

<SOAP: Envelope xmlns: SOAP="urn:schemas-xmlsoap-org:soap">
    <SOAP: Header>
        <t: Transaction xmlns: t="some-URI" SOAP: mustUnderstand="1">
        328
        </t: Transaction>
    </SOAP:Header>
    <SOAP: Body>
        <m: getBalance "xmlns: m="Some-Namespace-URI">
            <AccountIdentification>
                <AccountNumber>3044005</AccountNumber>
                <pin>****</pin>
                <name>Hans Muster</name>
            </AccountIdentification>
        </m: getBalance>
    </SOAP: Body>
</SOAP: Envelope>
```

SOAP: Beispiel (Antwort des Server)

HTTP 200 ok
Connection: close
Content-Type: text/xml
Content-Length: nnnn

```
<SOAP: Envelope xmlns: SOAP= “urn: schemas-xmlsoap-org: soap“>
    <SOAP: Body>
        <m:getBalance xmlns: m= “Some-Namespace-URI/ “>
            <return> -1350.50 </return>
        </m: getBalance>
    </SOAP: Body>
</SOAP: Envelope>
```

SOAP: Kodierungsstruktur

- Einbettung in HTTP POST Requests und zugehörige Antworten
- Envelope: Definition des Namensraumes und ggf. Spezifikation eigener Kodierregeln für Parametertypen
- Header: Übergabe impliziter Steuerparameter (Hinweis: hier bedeutet „Transaktion“ lediglich Request/Response-Interaktion)
- Body: Eigentliche Kodierung des Aufrufs und der Parameter

SOAP: Kodierungsstruktur

```
<element name= “AccountIdentification“>
    <complexType>
          <element name= “accountNumber“ type=“xsd:int“/>
           <element name= “pin“ type=“xsd:int“/>
           <element name= “name“ type=“xsd:string“/>
    </complexType>
</element>
```

- auch alle anderen wesentlichen Datentypen spezifizierbar (z.B. (variable) Arrays, Enumerations etc.)
- Abbildung auf Datentypen gängiger Programmiersprachen möglich

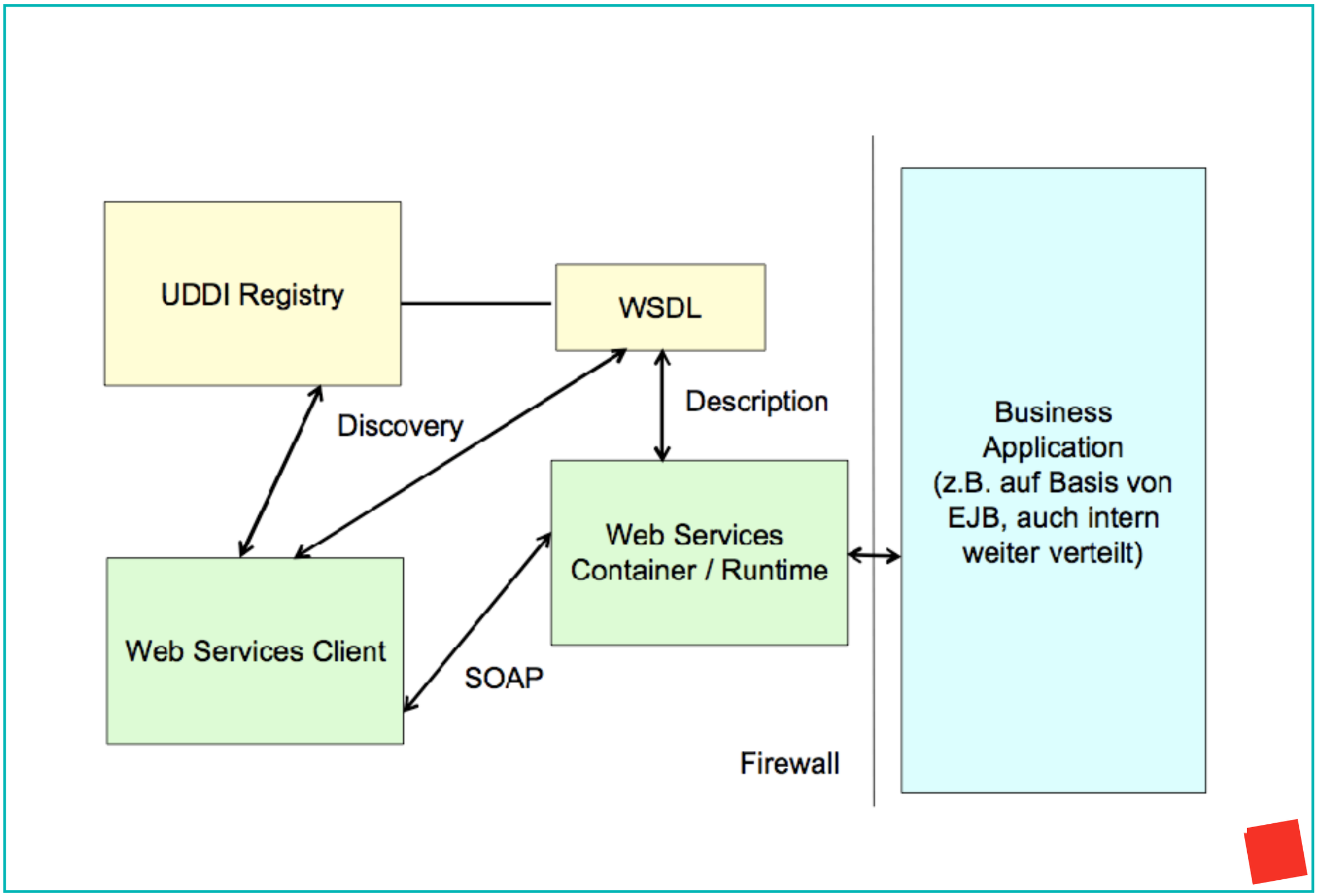
WebServices

- Herstellerübergreifende Initiative für Web-basierte Dienste
- Basis: Standardisierte Protokolle (z.B. SOAP / XML) und Middleware-Plattformen (Application Server)
- Definition: „Eingekapselte, lose gekoppelte Funktionen, die über Standardprotokolle zugänglich sind.“
- Schnittstellenbeschreibungen mittels WSDL (WebServices Description Language)
- Vermittlung von Diensten mittels UDDI (Universal Description, Discovery and Integration); vergleichbar mit Directory Service; http://www.uddi.org

WebServices

- Sicherheitsarchitektur WS(WebServices)-Security: Ergänzung von SOAP um digitale Signaturen (PKI – Public Key Infrastructure) und Verschlüsselung
- Perspektivisch auch Erweiterung von Firewalls um Funktionen zur Überprüfung von Authentisierung und Autorisierung bei SOAP-Kommunikation via „http“ („Port 80“)
- Insgesamt: Rahmenwerk zur Beschreibung netzweiter Dienste, das von Herstellern durch Systemlösungen konkretisiert wird, u.a. IBM, Microsoft, Sun, BEA Systems etc.
- Standardisierung durch W3C und Web Services Interoperability Organisation

WebServices



WDSL (WebServices Description Language)

Beispiel (verkürzt):

```
<binding name="BankServer" type="tns:BankServerPortType">
  <soap:binding style = "rpc" transport = "http://... "/>
  <operation name = "getBalance">
    <soap:operation soapAction = "urn:xmethodsBankServer"/>
    <input> ... </input>
    <output> ... </output>
  </operation>
</binding>
```

Aufrufmodi: oneway ; request-response (Client/Server)
notification ; solicit-response (Server/Client)

WebServices: Einsatzszenarien

- Phase 1: Unternehmensinterne Kopplung von Einzelanwendungen in Ergänzung zu anderen Protokollen wie Java RMI oder CORBA

- Phase 2: Unternehmensübergreifende Kopplung zwischen eng kooperierenden Partnern (z.B. Hersteller / Zulieferer) unter fester Zuordnung von Diensten und klaren vertraglichen Rahmenbedingungen

- Phase 3: Offener Markt von Diensten; Leasing-Software; dynamische Selektion, Zuordnung, Nutzung und Abrechnung von Diensten (jedoch derzeit noch viele Unwägbarkeiten)

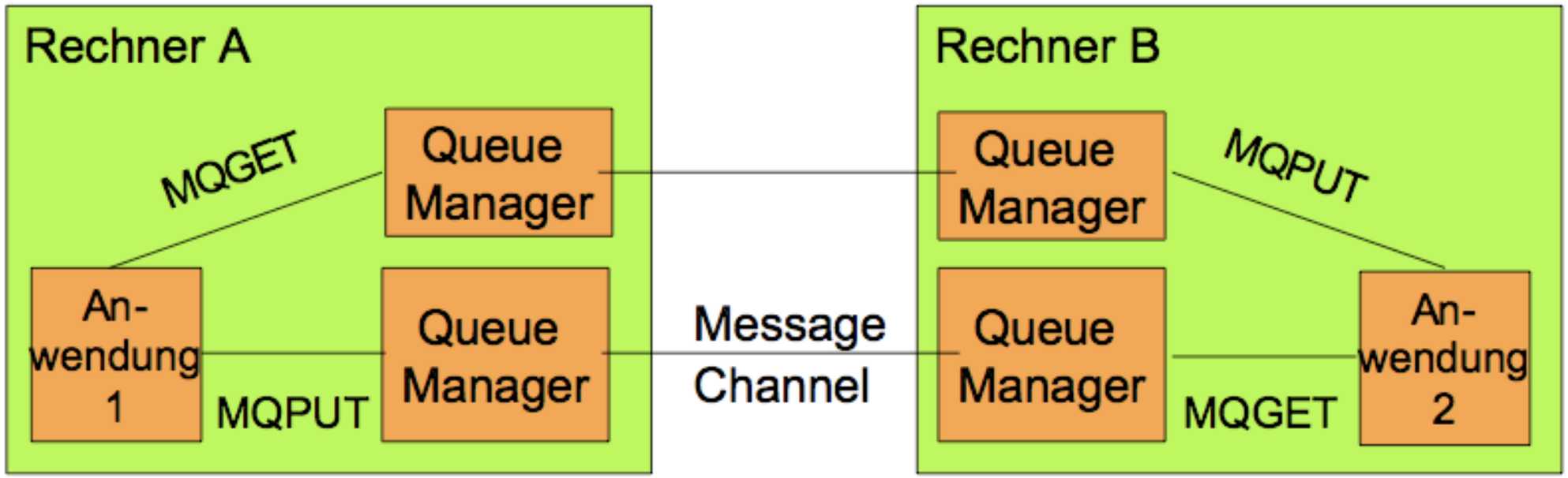
WebServices: Fazit

- Komfortabler, Web-basierter Aufrufmechanismus
- Mittels SOAP / HTTP auch Firewall-fähig
- Weitgehend automatische Generierung der Schnittstellenbeschreibungen aus Entwurfsrepräsentation durch Tools möglich
- **Aber:** Kein Ersatz von EJB oder .NET, sondern Zugangstechnologie vom Client zum Server, insbesondere via Internet

Message Oriented Middleware

- Produktbeispiele: IBM MQ Series, BEA MessageQ, Tibco etc.
- Basis: Messages, Queues mit Queue Manager
- Dynamische Kopplung zwischen Anwendungen und lokalen Queues über An- / Abmeldung
- Nutzung von Queues für Senden oder Empfangen; auch gemischte Nutzung möglich
- Kopplung verteilter Queue Manager über Message Channels
- C++-und Java-Support (konform zu JMS)
- Nutzung von XML (eXtensible Markup Language) zur Beschreibung der übertragenen Inhalte
- Unterstützung wesentlicher Betriebssystemplattformen

Beispielszenario



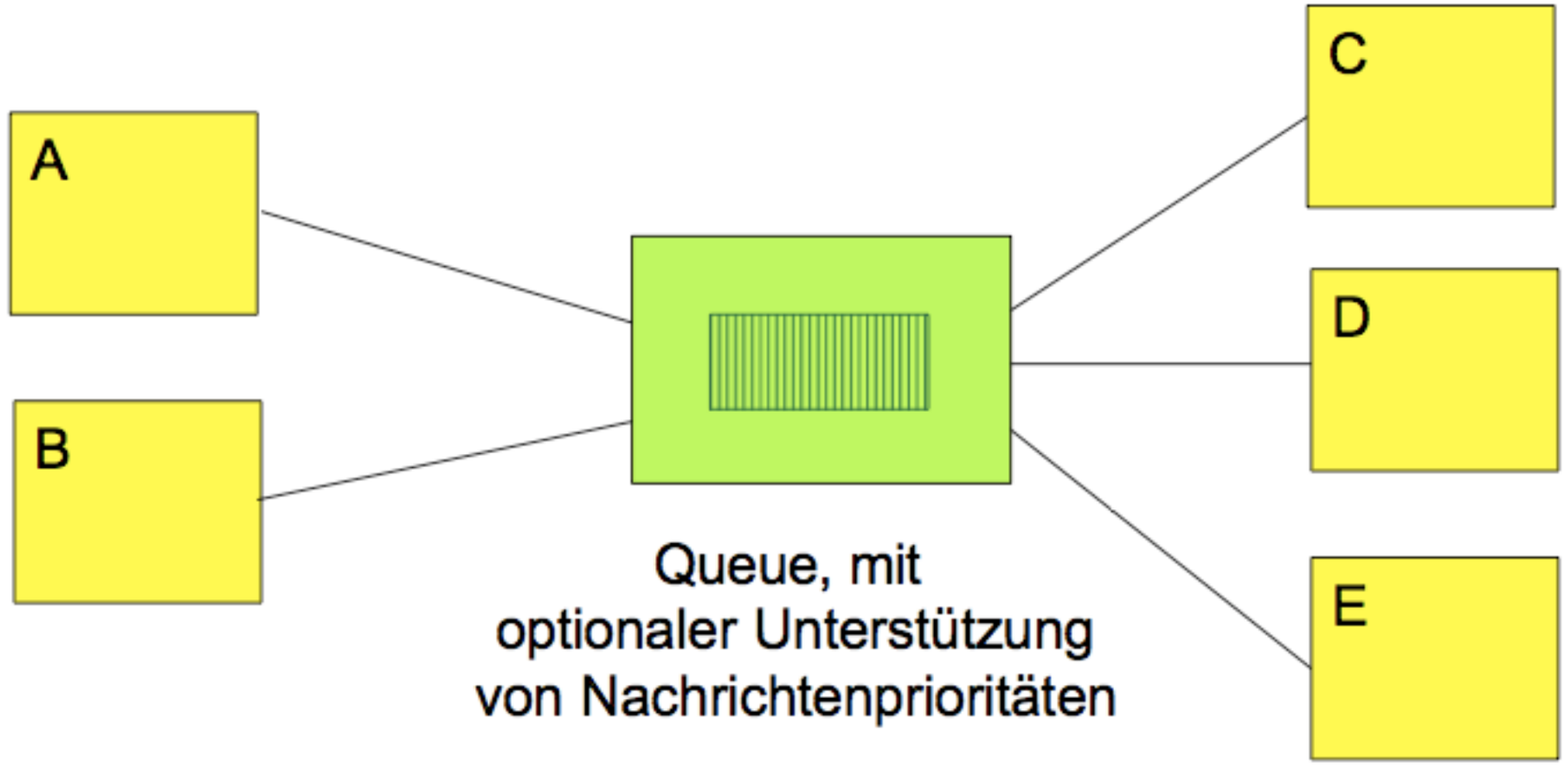
Entkopplung der Anwendungen durch Queue Manager

→ Nachrichtenweiterleitung auch bei nicht laufender Anwendung möglich

N:M - Kommunikation

Zugriff auf Server
durch mehrere Clients

- Lastausgleich
(selektive Auslieferung) oder
- Parallele Bearbeitung
(replizierte Auslieferung)



Message Oriented Middleware: Bewertung

Vorteile:

- Einfache Handhabbarkeit
- Zuverlässige Auslieferung von Messages
- Flexible Einsatzmöglichkeiten (z.B. Lastausgleich, Parallelisierung, Batch-Übertragung von Filialdaten etc.)
- **Gut für lose Kopplung von Programmen, z.B. via Internet oder für Mobile Computing geeignet**

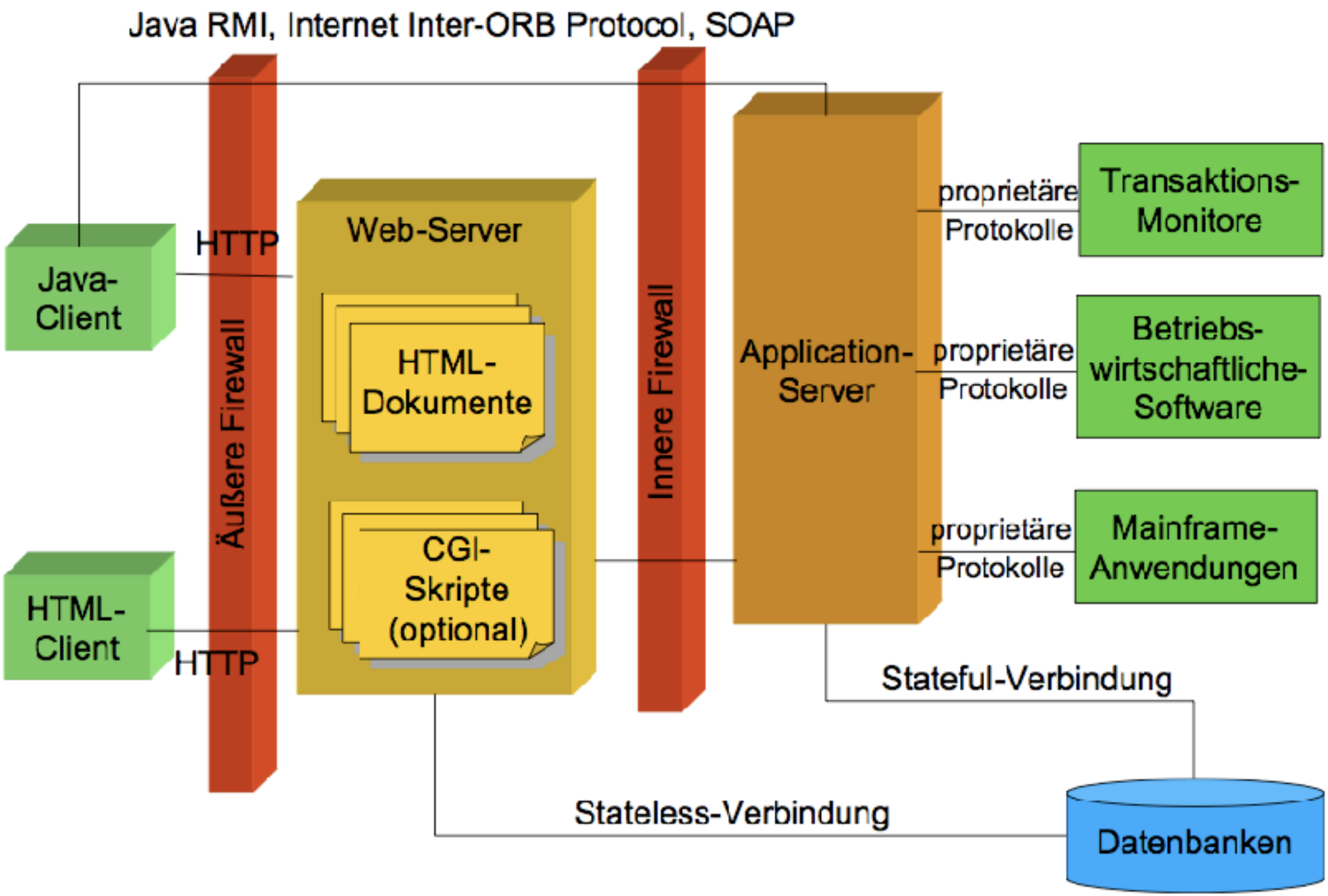
Nachteile:

- Eingeschränkte Kommunikationssemantik
- Interaktionsmodell anders als bei Prozedur- / Methodenaufrufen
- Begrenzte Verfügbarkeit höherer Services
- Bisher proprietäre Lösungen, erst schrittweise Standardisierung

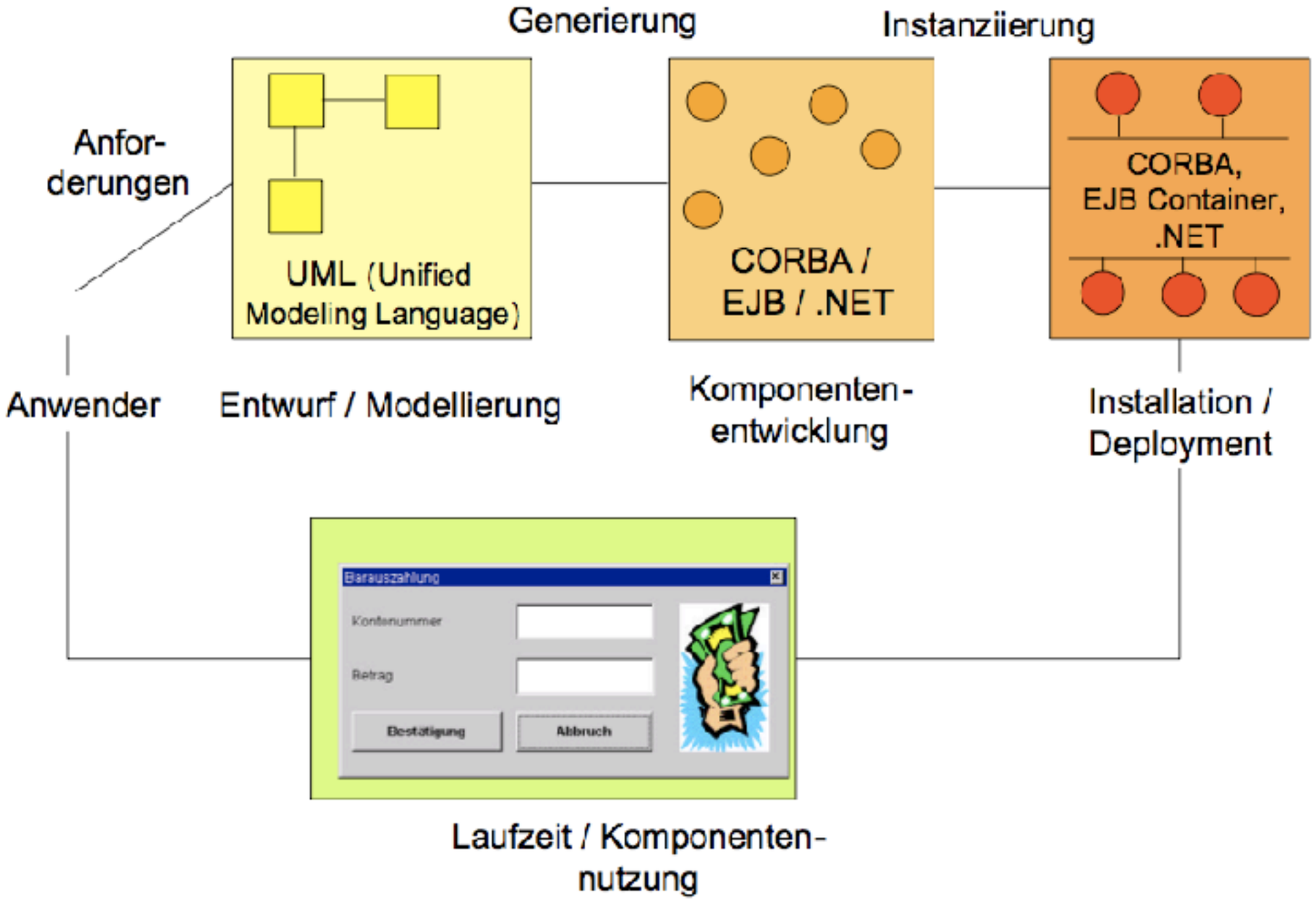
Message Oriented Middleware: Bewertung

- Schnittstellen-Server zwischen Web/Java-Client und Diensten der Unternehmens-DV („middle-tier“)
- Aufgaben:
  - Daten- und Aufrufanpassung
  - Legacy-Integration; Transaktionen
  - Zugriffsschutz
  - Lastverteilung

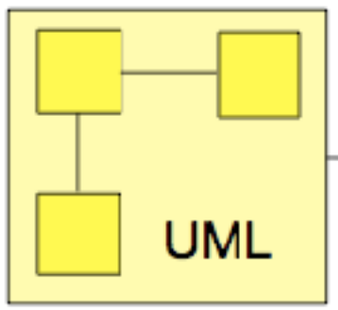
Architekturprinzip



Entwicklungsprozess



Modellierung und Generierung

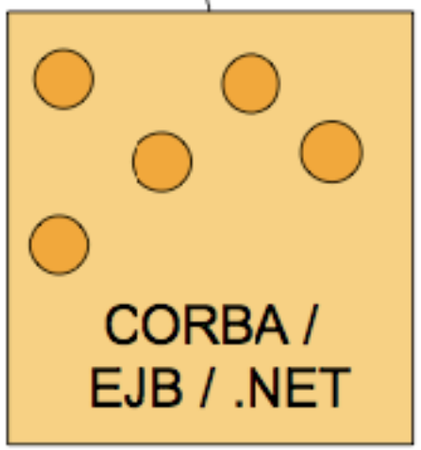


Entwurf /
Modellierung

```
<?xml version="... "?
<component name="Bank">
<interface name="Bank">
```

Komponentenbeschreibung

Generierung



Komponenten-
entwicklung

XML (eXtensible Markup Language) als Zwischenrepräsentation:

- Standardisierung, Portabilität
- Formalisierung (DTD - Document Type Definition bzw. XML Schema)
- Tool-Unterstützung

Interface-Beschreibung in XML (“XMI”)

```
<interface name="Bank">
  <superclass> General </superclass>
  <operation name="TransferRequest">
     <visibility> public </visibility>
     <returnType> long </returnType>
     <oneway> false </oneway>
  </operation>
  <attribute name="Description">
     <type> string </type>
     <visibility> public </visibility>
     <isReadonly> true </isReadonly>
  </attribute>
</interface>
```

Application Server

Wesentliche Funktionalität:

- Entwicklung und Verteilung von Java Anwendungen (“Three-Tier”)
- Skalierbarkeit (>100 Server, >10000 Clients): Multithreading, Connection-Reuse etc.
- Komponentenmodell (Enterprise JavaBeans, .NET etc.)
- Transaktionsunterstützung
- Zugang zu verbreiteten Datenbanken (Oracle, MS SQL Server, Sybase, DB2)
- Sicherheit (Authentisierung, Zugriffskontrolle)
- Unterstützung aktueller Java APIs (JDBC, JNDI, JMS etc.)
- Replikation und Lastverteilung
- Integration von Entwicklungsumgebungen (z.B. IBM Websphere Studio, Borland JBuilder, BEA Weblogic Workshop, MS Visual J++ / J#, C#, Rational Rose, Arcstyler etc.)
- Unterstützung von WWW-Diensten (z.B. Installation von HTML, Servlets etc.)

Enterprise Application Integration (EAI)

Ziel:

- Integration unterschiedlicher Anwendungen (Backend)
- Beispiele:
  - Enterprise Resource Planning (ERP)
  - Customer Relationship Management (CRM)
  - Supply Chain Management (SCM)

Technologische Basis:

- Middleware und Application Server (z.B. von IBM, BEA, Forte etc.)
- Zusätzliche produktspezifische Adapter

Lösungsansätze:

- Datenintegration
- Schnittstellen-basierte Integration (API-Integration)
- Workflow- / Prozess-orientierte Integration (bei komplexen Abläufen über mehr als ca. 5 Anwendungen)

Produktbeispiele:

- BEA Systems eLink - Adapter für SAP R/3 Integration
- Delta Software Technology, Wien: SCORE/ Integration Suite
- WRQ VeraStream EAI Suite: SAP R/3 Integration
- Sybase Integration Services:

  SAP, Peoplesoft,
  Oracle (ERP)
  Siebel, Vantive (CRM)

Application Server: Produktbeispiel

- BEA Weblogic
- IBM Websphere
- Borland Application Server
- IONA Orbix
- Oracle Application Server / .Now
- Sybase Enterprise Application Server
- Sun: Open Net Environment (One) / iPlanet
- Software AG EntireX
- Microsoft .NET

Open Source, u.a.:

- Enhydra
- Jonas
- Jboss
- Zope etc.

3.4 BACnet Over Web Services (Annex N, Annex H6)

The XML working group of ASHRAE SSPC 135 has introduced Addendum c to BACnet-2004 that specifies a Web Services interface to building automation and control systems. The addendum is in two parts:

- Annex N to BACnet defines the BACnet Web Services interface, BACnet/WS. BACnet/WS is a connectionless protocol using a Simple Object Access Protocol 1.1 interface (http://www.w3.org/2002/ws/) over HTTP (RFC 2616). The model and primitives exposed in Annex M are independent from the underlying protocol and could apply to any building automation protocol (LonWorks, KNX, ModBus …).
- Annex H6, Combining BACnet Networks with Non-BACnet Networks, that prescribes the gateway mapping specifically to and from BACnet messages. Annex H6 exposes a specific profile for Web Services access to underlying BACnet objects.

The Internet of Things: Key Applications and Protocols
von Olivier Hersent,David Boswarthick,Omar Elloumi, 2010

3.4.1 The Generic WS Model

The BACnet/WS fundamental data structure is a tree of *nodes* under a single *root node*. In order to allow more flexibility in the tree structure, BACnet/WS has a notion of *reference node* that can serve as an alias to a *referent node*.

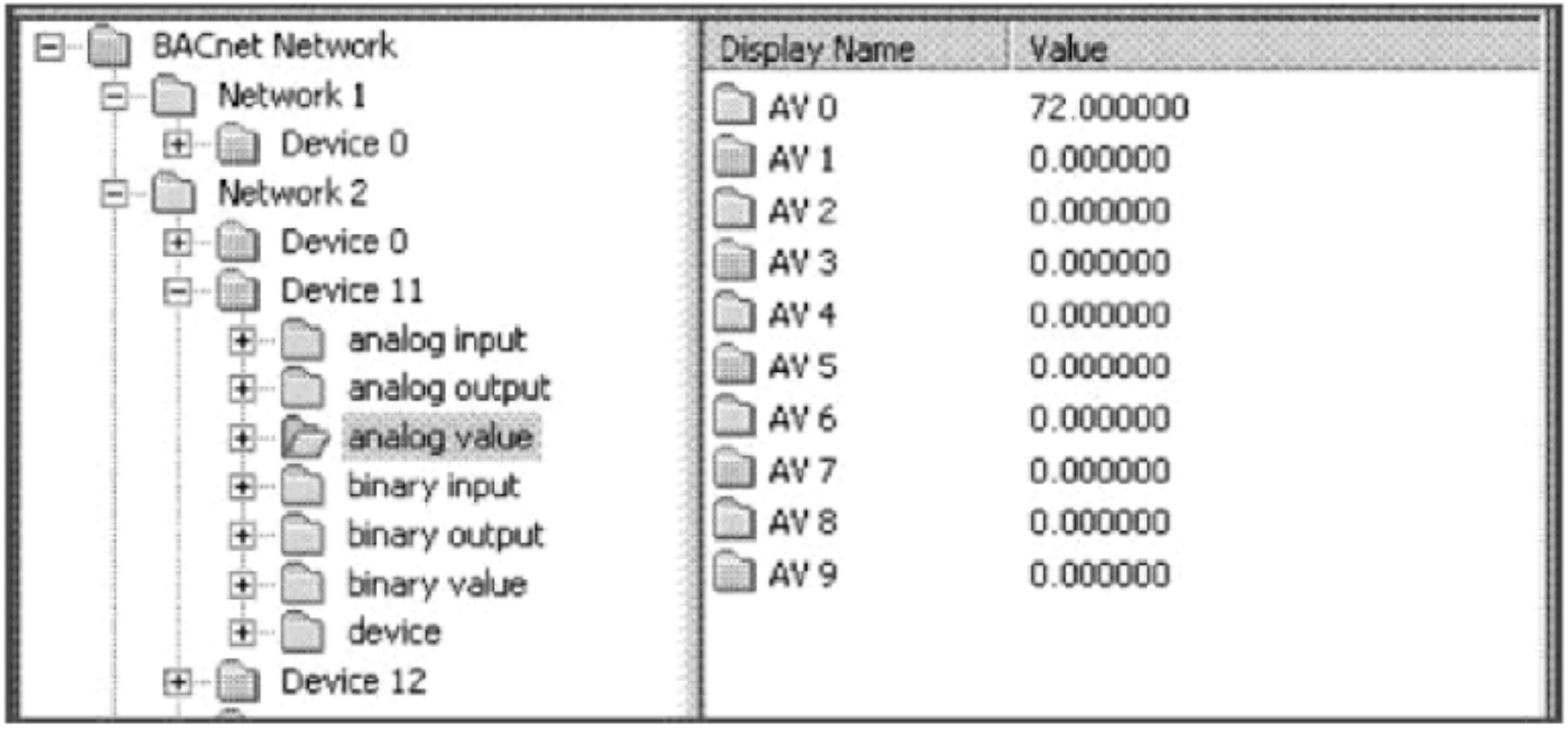
Each node and each property is identified by a path:

- /Floor2/Room3/Discharge Temp identifies a node;
- /Floor2/Room3/Discharge Temp:inAlarm identifies property inAlarm of node DischargeTemp;
- Another format commonly used by BACnet WS vendors (Figure 3.4) is the following /[Network]/[Device]/[ObjectType]/[Instance], for instance **/2/11/2/0** is a path to **network 2, device 11, analog value (object type 2) instance 0**;

The Internet of Things: Key Applications and Protocols
von Olivier Hersent,David Boswarthick,Omar Elloumi, 2010

BACnet WebServices

Figure 3.4 Snapshot of the SCADA engine BACnet web service.



The Internet of Things: Key Applications and Protocols
von Olivier Hersent,David Boswarthick,Omar Elloumi, 2010

- Special value <Path>:Children indicates all the nodes one level below the current node. For instance /:Children returns all the children of the root path. /1/1/0:Children returns all paths to Analog Inputs on Device 1.

The network visible state of each node is exposed as the node value, and a collection of attributes (Figure 3.5). BACnet uses Primitive attributes (native XML types), Array attributes (arrays of Primitive values), Enumerated attributes (choice of XML strings specified by BACnet/WS). Only the Value attribute is writable with the services defined by the standard.

BACnet/WS defines a small set of 5 standard nodes required in any implementation, for example: 10.5}

Figure 3.5 BACnet/WS mandatory and most common attributes.

Attribute	XML type	Description
NodeType	String	Hint about a node content. One of Unknown /System /Network /Device /Functional /Organizational /Area

The Internet of Things: Key Applications and Protocols
von Olivier Hersent,David Boswarthick,Omar Elloumi, 2010

BACnet WebServices?

```
import BAC0
bacnet = BAC0.connect(ip='10.15.20.1')
myController = BAC0.device('10.15.20.120', 33992, bacnet)
for i on range(len(myController.points)):
    print(myController.points[i])
```

```
/02/001/00/00/P.03/CAI/b : 31.25 percentRelativeHumidity
/02/000/00/00/SY_Datsi.01/procBar : 100.00 percent
/02/101/00/03/P.07/CAO/Y : 0.00 percent
/02/000/01/00/S238.01/5110 : 0.00 percent
/02/000/01/00/S238.01/5111 : 0.00 percent
/02/000/01/00/S238.01/5112 : 0.00 percent
/02/000/01/00/S238.01/5100 : 18.00 degreesCelsius
/02/000/01/00/S328.01/5705 : 26.00 minutes
/02/000/01/00/S328.01/5700 : 80.00 hours
Absolute Feuchte : 0.89 gramsOfWaterPerKilogramDryAir
Taupunkt : -17.36 degreesCelsius
/02/000/00/00/SY_Datsi.01/freeMem : 95.00 percent
/02/000/00/00/SY_Datsi.01/state : Bereit
/02/000/00/00/SY_Datsi.01/report : kein Fehler
/02/000/00/00/SY_Datsi.01/command : Bereit
/02/000/00/00/SY_Modul.01/State/F017.02/State : Anlage EIN
/02/002/00/00/P.01/CDI/k : True
/02/002/00/00/P.02/CDI/k : False
/02/002/00/00/P.06/CDO/K : False
/02/002/00/00/P.07/CDO/K : False
/02/002/00/00/P.08/CDO/K : False
/02/002/00/00/P.09/CDO/K : False
/02/000/01/00/M.01/1 : True
/02/000/01/00/S118.01/C.01/BACInfo : True
/02/000/00/03/T.02/t : False
```

With this new version, BACnet/WS ceases to be a simplified interface serving limited purposes: it really becomes a fully functional interface that provides access to the full functionality of BACnet.

The Internet of Things: Key Applications and Protocols
von Olivier Hersent,David Boswarthick,Omar Elloumi, 2010