

# Ubiquitous Computing

(Ubiquitäre Informationstechnologien)

Vorlesung im WS 03/04

---



**Teco**

**Michael Beigl**

Universität Karlsruhe

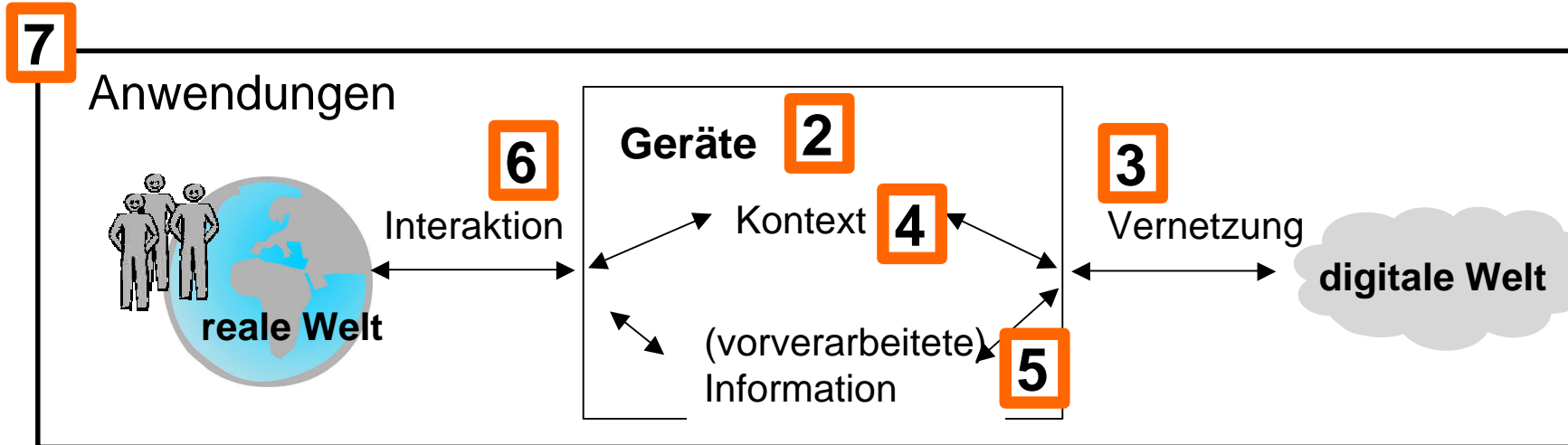
Institut für Telematik

Telecooperation Office

[www.teco.uni-karlsruhe.de](http://www.teco.uni-karlsruhe.de)

# Aufbau der Vorlesung

- 1** Grundlagen
- 2** Geräte
- 3** Vernetzung  
Netzwerke  
Middleware
- 4** Kontext
- 5** Information
- 6** Interaktion
- 7** Anwendungen



# Vernetzung: höhere Schichten

---

- Einführung
- Kommunikationsparadigmen
- Jini Service Infrastruktur
- Bluetooth
- Salutation
- OBEX Object Exchange Protokoll
- HAVi AV Kontrolle
- UPnP Plug&Play

# Einführung: Ubicomp-Netze

---

## Netze für Ubiquitous Computing

- Diversifikation von Endgeräten: mobil, eingebettet, spezialisiert
- Mobilität: mobile Nutzer, mobile Geräte
- Allgegenwart: überall, insbesondere auch im Heimbereich
- Spontaneität: ad hoc Vernetzung von Geräten
- Konvergenz: Daten, Audio/Video, Steuerung

## Entwicklungstrends

- Nutzung „alter Infrastrukturen“ und Schaffung neuer
  - trad. LANs, Funk-LANs, Plug&Play-Busse, Bluetooth, IrDA, Phonenumber, Powerline, ...
- Extrem heterogene Umgebungen: Geräte und Netze
- hohes Maß an Dynamik: Hot&Plug Play, mobile Netze, kurzlebige Verbindungen

# Einführung: Ubicomp-Netze

---

## Typische Szenarien und Herausforderungen

- über Mobiltelefone im Haus bedienen
  - heterogene Geräte und Netze (mobil/Heim)
  - kohärente Sicht auf Dienste im Heim
- Kamera sucht Drucker in fremder Umgebung
  - wie kann ein „geeigneter“ Drucker gefunden werden ?
  - wie unterhält man sich mit einem fremden Gerät ?
- Hausregelung
  - Heizung sucht Thermostat und Bewegungsmelder

## Wichtigste Herausforderung: Komplexität verbergen

- vor allem vor dem Anwender
- keine manuelle Installation / Konfiguration (ad-hoc)
- Abstraktionen für die Anwendungsentwicklung → Middleware

# Middleware

---

## Was ist Middleware ?

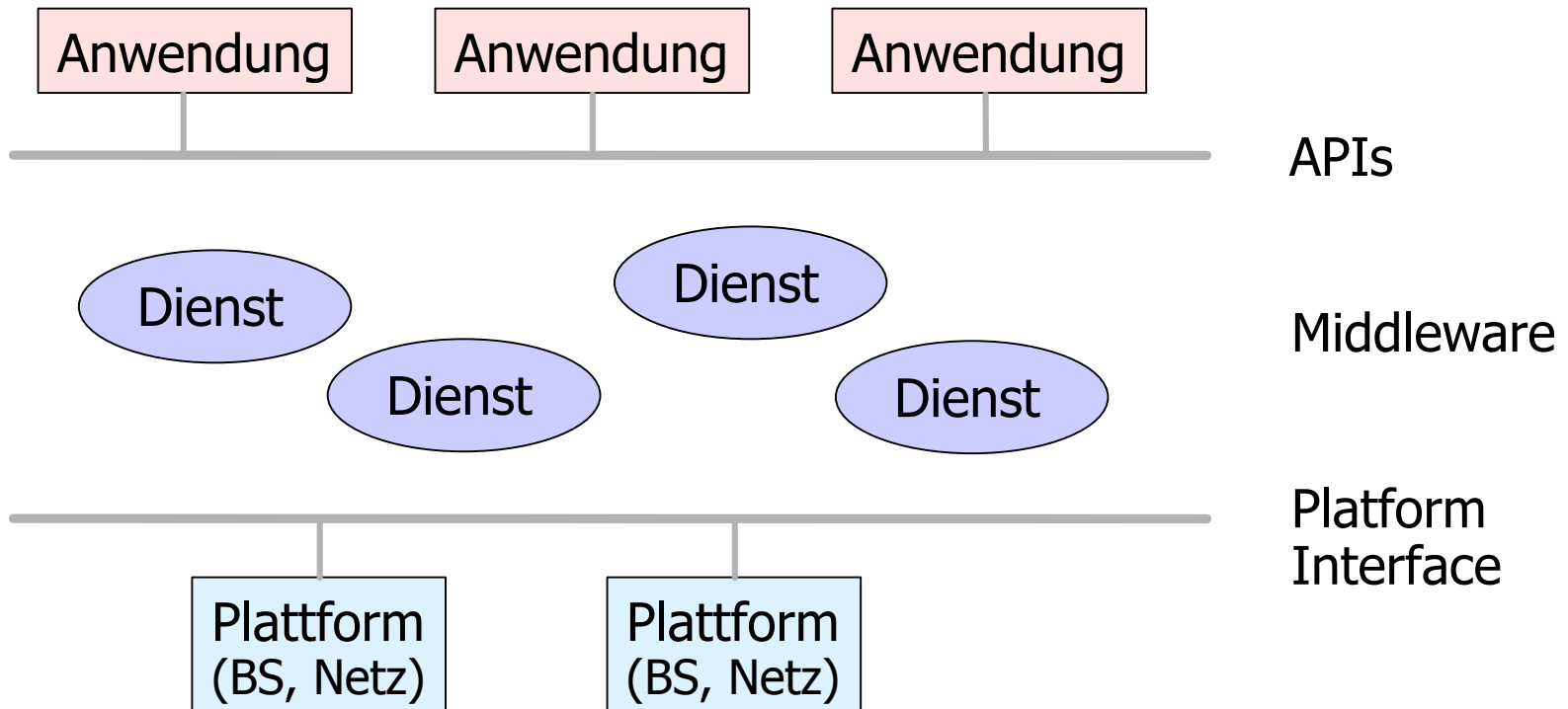
- „der Slash in Client/Server“
- Komponenten für Entwicklung und Einsatz verteilter Systeme
- angesiedelt zwischen Netzwerktechnologie(n) und Anwendung

## Wofür ?

- Abstraktion von Netzwerkprogrammierung
- Interoperabilität: Zwischenschicht über unterschiedlichen Geräte- und Netzwerkplattformen
- Komponenten für allgemeine Aufgaben in verteilten Systemen: z.B. Name Service, Security Service,...

# Middleware

---



# Middleware

---

## RPC: Remote Procedure Call (80er Jahre)

- Prozedurales Paradigma
- entfernter Prozeduraufruf analog zu lokalem Aufruf
- Code für die Kommunikation wird automatisch erzeugt

## Objekt-orientierte Middleware (90er Jahre)

- Objekt-orientierte Programmierung
- Kommunikation mit entfernten Objekten über automatisch erzeugte lokale Proxies (z.B. „stubs“ in CORBA)
- Vermittlungsdienste (z.B. Object Broker)

## Java Remote Method Invocation

- Java's Middleware für Methodenaufrufe von Objekten, die in verschiedenen VM ablaufen

# Middleware für UbiComp

---

## Integration heterogener Geräte

- Gateways für kohärenten Zugang zu heterogenen Umgebungen
- Service Paradigma: dynamischer Verbund von Geräten ohne zentrale Komponente; spontane Bildung verteilter Systeme

## Service Gateways

- Bündelung von Diensten über Gateways
- Residential Gateways: Verbindung zwischen Heimnetz und Außenwelt (= Internet)
  - bietet Geräten im Haus Zugriff auf Internet-Dienste
  - bietet externen Diensteanbietern kohärenten Zugang zu Geräten/Infrastruktur im Haus (z.B. für Fernwartung, Sicherheitsüberwachung,...)
  - Administration durch Gateway Operator
  - Standardisierung: Open Service Gateway Initiative (OSGi)

# Vernetzung: höhere Schichten

---

- Einführung
- **Kommunikationsparadigmen**
- Jini Service Infrastruktur
- Bluetooth
- Salutation
- OBEX Object Exchange Protokoll
- HAVi AV Kontrolle
- UPnP Plug&Play

# Kommunikationsparadigmen

## Ablauf einer Kommunikation

- Initiierung: Auswahl der Kommunikationspartner  
→ Wie **Auswahl**
- Durchführung: Austausch von Kommunikation  
→ **Grund Kommunikation**
- Beendigung

### Auswahl

	ID	Dienst	Kontext
Kom.-Grund	Info	HTTP	<b>IrOBEX</b>
Dienst	<b>Jetsend</b>	<b>Jini, UPnP, HAVi</b>	
Kontext			RAUM

# Kommunikationsparadigmen

## Ablauf einer Kommunikation

- Initiierung: Auswahl der Kommunikationspartner  
→ Wie **Auswahl**
- Durchführung: Austausch von Kommunikation  
→ **Grund Kommunikation**
- Beendigung

### Auswahl

	ID	Dienst	Kontext
Info	HTTP		IrOBEX
Dienst	Jetsend	Jini, UPnP, HAVi	
Kontext			RAUM

# Service Paradigma

---

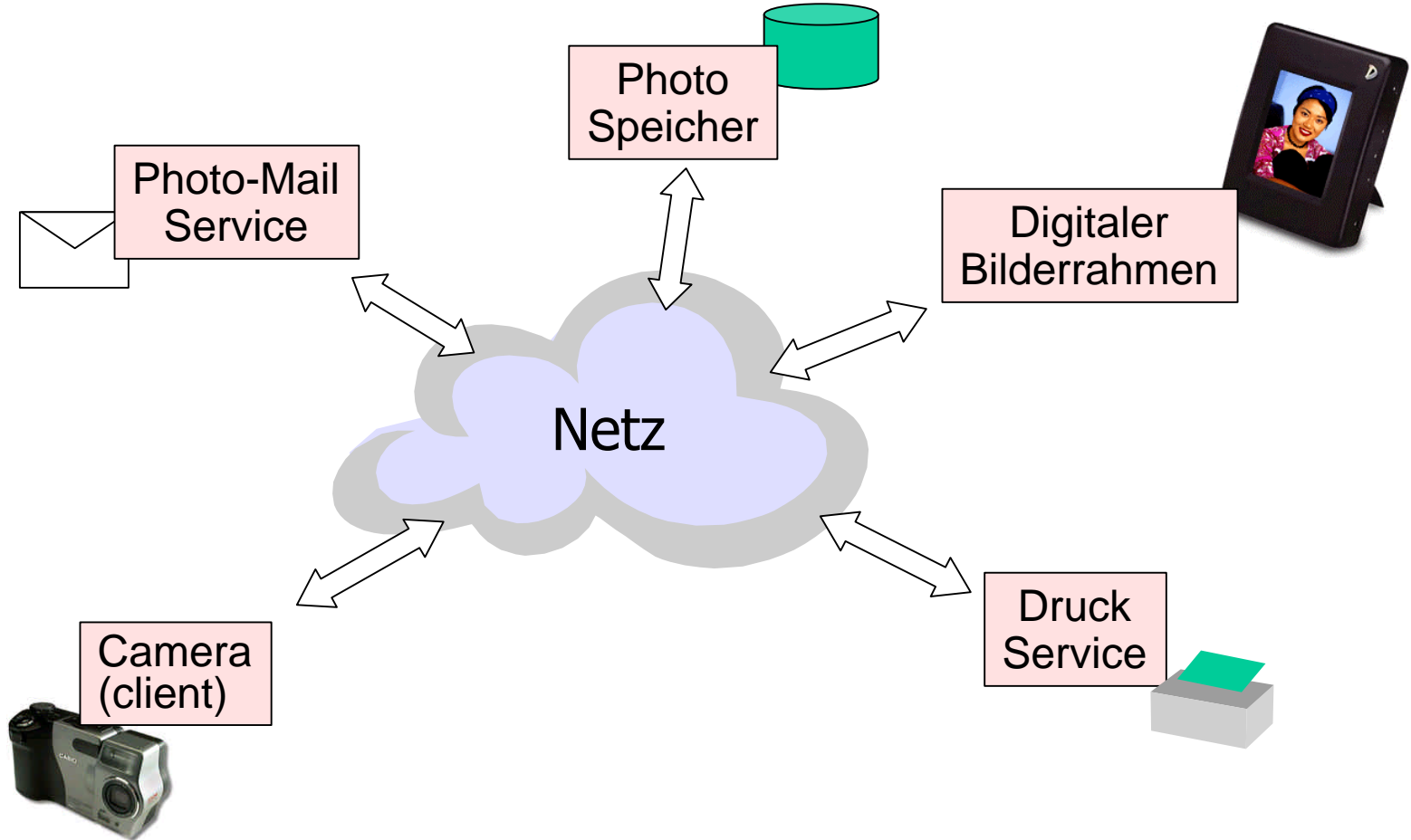
## Everything is a Service

- Geräte ebenso wie Software
- vgl. Objekt-orientierung: „everything“ is an object
- Services werden durch Interfaces definiert, über die sie ihre Funktionalität zur Verfügung stellen
- Services werden beschrieben durch Typ und Attribute
- Services können sich zu Systemen verbünden („federation“)

## Beispiele für Services

- Kamera, Drucker, Fax, Scanner, Speicher, Rechenleistung
- Türöffner, Beleuchtung, Alarmanlage, Stromzähler, ...
- Rechtschreibprüfung, Formatkonvertierung, ...
- Online Banking, Aktienhandel, ...
- Hotelführer, Stadtplan, ...

# Service Federation: Beispiel



# Service Paradigma

---

## Netzwerk-zentrisch

- „the network is the computer“
- Netzwerk = Hardware und Softwareinfrastruktur für Dienste
- Sichtweise: „Netzwerk, an das Geräte angeschlossen sind“ (statt „Geräte, die vernetzt werden“)
  - Netzwerk existiert immer, Geräte/Dienste sind transient
  - Komponenten und Kommunikationsbeziehungen kommen und gehen

## Spontane Vernetzung

- Services finden sich in der offenen Netzwerkkumgebung zu zeitweiligen Verbundsystemen zusammen
- müssen sich dazu nicht a priori kennen
- typisches Szenario: Client wacht auf und fragt nach Diensten in der lokalen Umgebung

# Service Paradigma

---

## Spontane Vernetzung von Services

- wie werden Services aufeinander aufmerksam ?
- wie können bestimmte Services in einer fremden Umgebung gefunden werden ?
- wie verständigen sich Services, wenn sie sich gefunden haben ?
- Werden Dienstinfo. in Infrastruktur gehalten oder ad-hoc ermittelt?

## Infrastruktur für Service Discovery

- „Registry“: Verzeichnis/Vermittlung von Services
- Protokolle zum Registrieren und zum Anfragen von Services
- Protokolle für Client-Zugriff auf Service, und für die Nutzung von Services durch Clients
- z.B. Sun's Jini aufbauend auf Java/RMI, Microsoft's UPnP (Universal Plug & Play)
- z.B. HAVi (Home Audio/Video interoperability) aufbauend auf IEEE.1394 für Home Entertainment Dienste

# Vernetzung: höhere Schichten

---

- Einführung
- Kommunikationsparadigmen
- **Jini Service Infrastruktur**
- Bluetooth
- Salutation
- OBEX Object Exchange Protokoll
- HAVi AV Kontrolle
- UPnP Plug&Play

# Jini Service Infrastruktur

---

## Hauptkomponenten

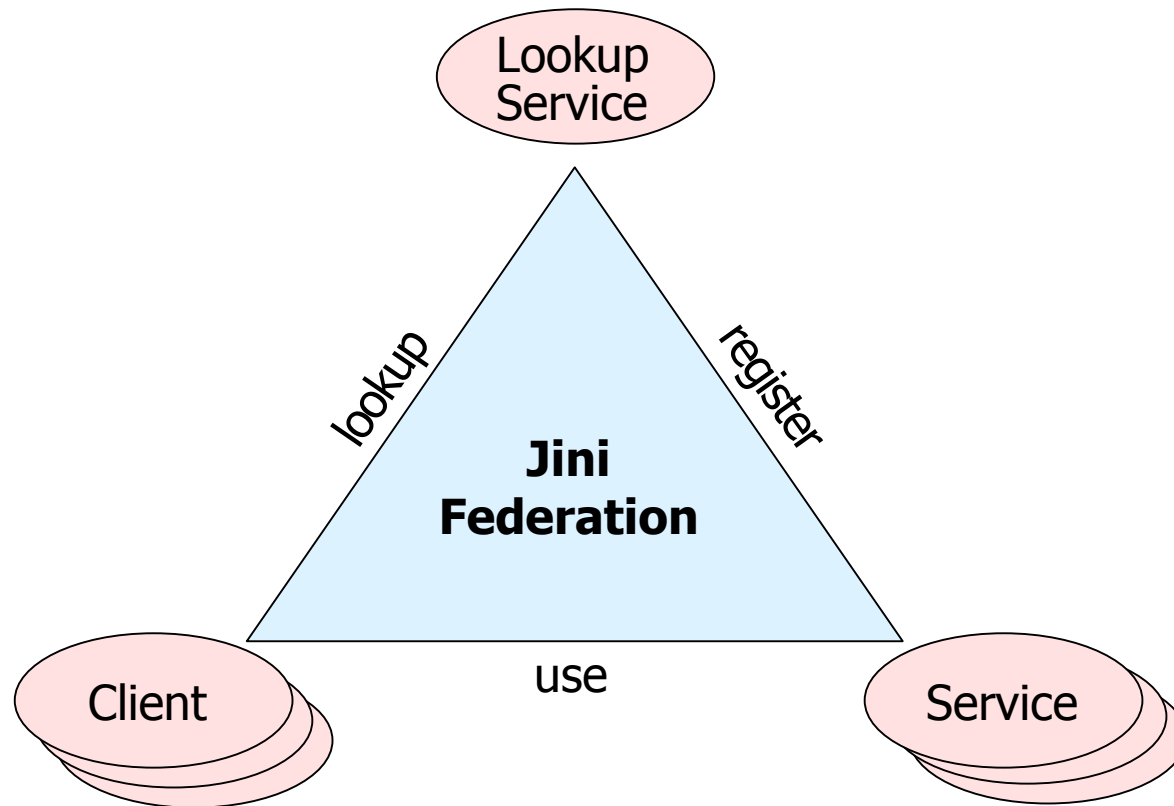
- Lookup Service (LUS): „Registry“ für Services
- Protokolle basierend auf TCP/UDP/IP
  - Discovery & Join, Lookup von Services
- Proxy Objekte
  - als lokale Vertreter für Services

## Lookup Service

- Verzeichnis ähnlich RMI Registry
- Aufgabe: „Helpdesk“ für Services/Clients
  - Registrierung von Services, die angeboten werden
  - Verteilung von Diensten an anfragende Clients

# Jini Lookup Service

---



# Discovery: Finden eines LUS

---

## Finden eines Lookup Services

- ... ohne a priori Kenntnis des Netzwerks
- Service Provider sucht LUS um Service anzumelden (register)
- Client sucht LUS um einen Service anzufragen (look up)

## Discovery Protokoll

- Multicast-Anfrage an bekannten Port
- Lookup Service lauscht auf entsprechendem Port und antwortet mit Proxy Objekt (interface **ServiceRegistrar**)
  - Proxy Objekt wird in die anfragenden Service geladen
  - Kommunikation mit LUS dann über den Proxy
- weitere Discovery-Protokolle
  - Unicast: Service kann LUS direkt ansprechen, wenn er die IP-Adresse schon kennt
  - Multicast Announcement: LUS meldet sich per Multicast, z.B. nach Ausfall

# Join: Registrieren eines Services

---

## Join Protokoll

- Service Provider hat einen Proxy des LUS für die Kommunikation empfangen
- Provider registriert über den Proxy seinen Service: **register()**
- Provider übergibt dabei dem LUS
  - den eigenen Service Proxy
  - Attribute, die den Dienst beschreiben (z.B. „600 dpi“, „version 21.1“, ...)
- Service tritt mit dem Join in den Jini-Verbund ein
  - Provider kann nun über den LUS gefunden und von anderen Services genutzt werden

# Lookup: Suchen von Services

---

## Lookup Protokoll

- Client sucht bestimmten Service
- kennt LUS und verfügt über Proxy für die Kommunikation (via Discovery Protokoll)
- sendet Anfrage an LUS in Form eines „Service Template“
  - ID, Typ, Attribute
- LUS antwortet mit keinem/einem/mehreren passenden Services
  - ggf. Auswahl im Client
- Client erhält vom LUS Proxy des vermittelten Services
- Client nutzt Proxy für direkte Kommunikation mit dem Provider
  - beliebiges Protokoll
  - Proxy: Gateway zu Service-Funktionalität beim Provider
  - Proxy kann aber auch (Teil der) Service-Funktionalität implementieren, d.h. Ausführung beim Client

# Lookup: Service Matching

---

## Service Template

- Service ID (Registration Number): kann angegeben werden, falls Dienst bereits bekannt ist (durch frühere Nutzung)
- Service Typ: definiert durch die Schnittstelle
- Attribute (sog. Entries), die den Service beschreiben

## Service Matching

- Übereinstimmung via Attribute: Mehrwert gegenüber traditionellem Naming Service: Service-Auswahl über beschreibende Merkmale
- aber nur exakte Übereinstimmung, kein „größer als“, keine Query-Sprache
  - z.B. Anfrage an „600dpi“ Drucker stimmt nicht mit registriertem „1200dpi“ Drucker überein

# Service Paradigma

---

## Spontane Vernetzung von Services: Jini Konzepte

- wie werden Services aufeinander aufmerksam ?
  - Jini: Lookup Services als Vermittlungsstelle, Registrierung von Services über Discovery & Join
- wie können bestimmte Services in einer fremden Umgebung gefunden werden ?
  - Discovery von Lookup-Services als Verteiler in fremder Umgebung
  - Lookup anhand von Service Templates, insbesondere auch anhand beschreibender Attribute
- wie verständigen sich Services, wenn sie sich gefunden haben ?
  - Service Proxy wird in den anfragenden Client geladen
  - beliebiges Protokoll, kein spezifisches Aushandlungsverfahren

# Vernetzung: höhere Schichten

---

- Einführung
- Kommunikationsparadigmen
- Jini Service Infrastruktur
- **Bluetooth**
- Salutation
- OBEX Object Exchange Protokoll
- HAVi AV Kontrolle
- UPnP Plug&Play

# Bluetooth Service/Profiles

---

## Profile

- geben an, welche Funktionalität implementiert ist
- Wichtige Profile sind:
  - GAP Profile: Generic Access Profile for discovery and link management
  - SDAP Profile: Service Discovery Application Profile for discovering services and information retrieval
  - SPP Profile: Serial Port Profile for emulating serial cable connections
  - GOEP Profile: Generic Object Exchange Profile (OBEX)
  - CTP Profile: Cordless Telephone Profile for telephony features.
  - IP Profile: Intercom Profile for intercom functionality also referred to as the "walkie-talkie" usage
  - HS Profile: Headset Profile
  - LAP Profile: LAN Access Profile for LAN access using PPP

# Bluetooth: L2CAP, PSM

## L2CAP

- logische Verbindung, in Software (nicht auf BT-Chip)
- Segmentierung von Paketen
- Dienstgütespezifikation

## Protocol und Service Multiplexer (PSM)

- dient der Ermittlung des Dienstes z.B.
- Service Discovery Protocol (SDP)
- RFCOMM
- Telephony Control Protocol Specification (TCS)

TCS  
SDP  
IP  
OBEX  
RFCOMM

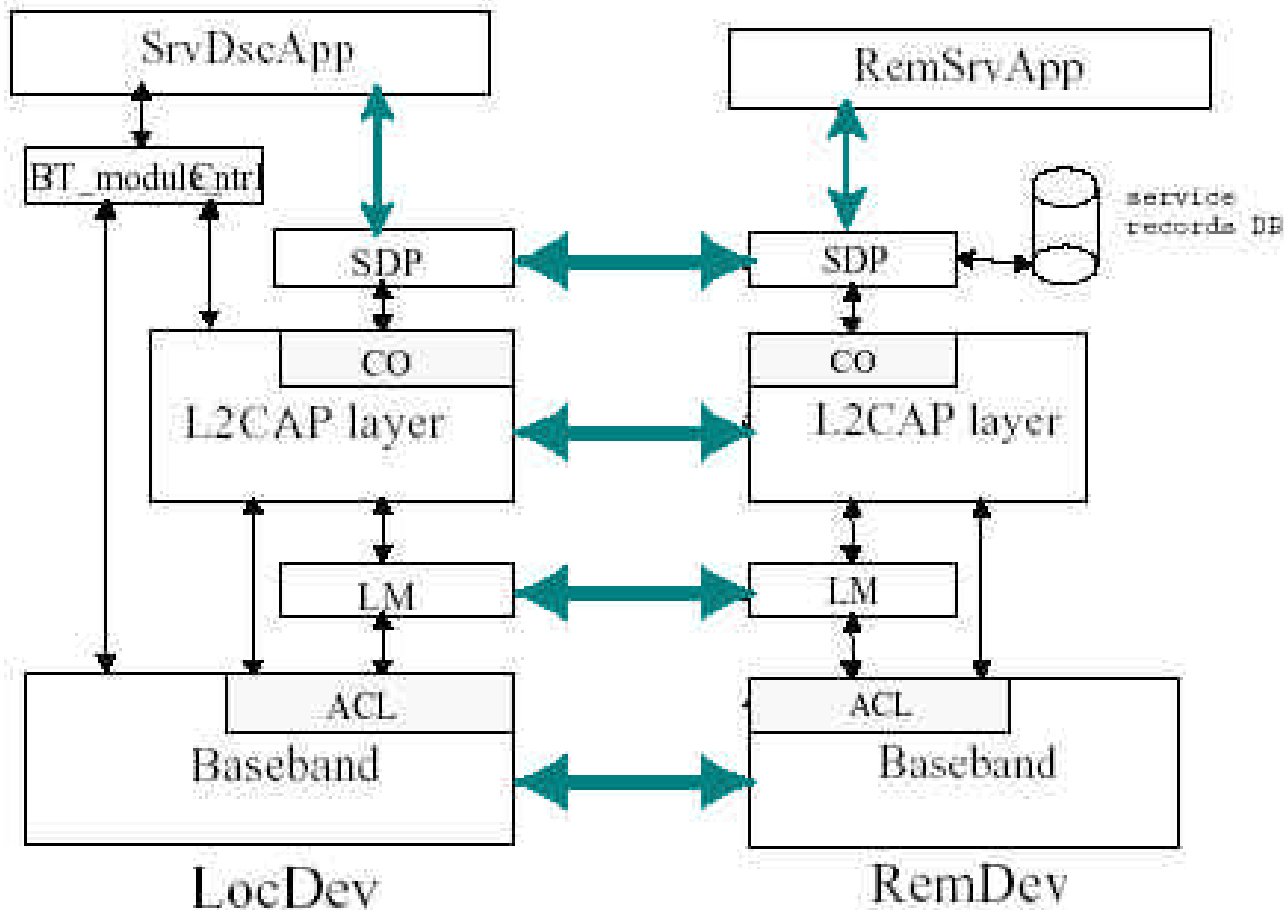
L2CAP

HCI —————

Link, Baseband  
RF (Hardware)

# Bluetooth SDP

## Service Discovery Protocol (SDP)



# Service Discovery Protocol

---

## SDP

- dezentrale Dienstnachfrage, kein Repository in einer „Infrastruktur“, nur Dienste des eigenen Gerätes
- Diensterkennung, Dienstvermittlung
- Dienstkommunikation wird vom Dienst selbst durchgeführt
- keine Zugriffskontrolle
- Interne Datenbank (Service Record DB) besteht aus AttributID und Attributwert Paaren
- Beinhaltet Beschreibung/ID des Dienstes, Name, Charakteristik
- Protokoll ist in Attribut ProtocolDescriptorList beschreiben
- Suche via „Search Pattern“ = Liste von UUIDs die irgendwo in den Attributen auftauchen müssen (UND verknüpft)

# Vernetzung: höhere Schichten

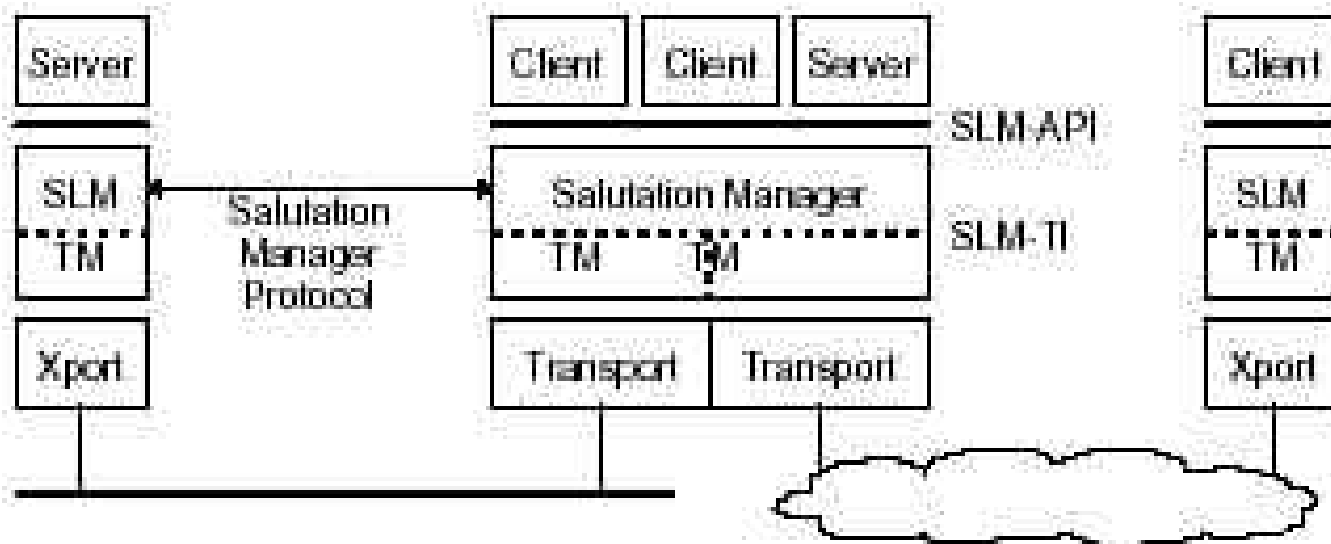
---

- Einführung
- Kommunikationsparadigmen
- Jini Service Infrastruktur
- Bluetooth
- **Salutation**
- OBEX Object Exchange Protokoll
- HAVi AV Kontrolle
- UPnP Plug&Play

# Salutation

## Salutation

- Verwendet existierenden Transport, z.B. TCP/IP, Bluetooth, Irda
- Implementierungsvorschläge vorhanden
- Ergänzt System um sehr flexible Aushandlung von Dienstparametern und Auswahl von Diensten

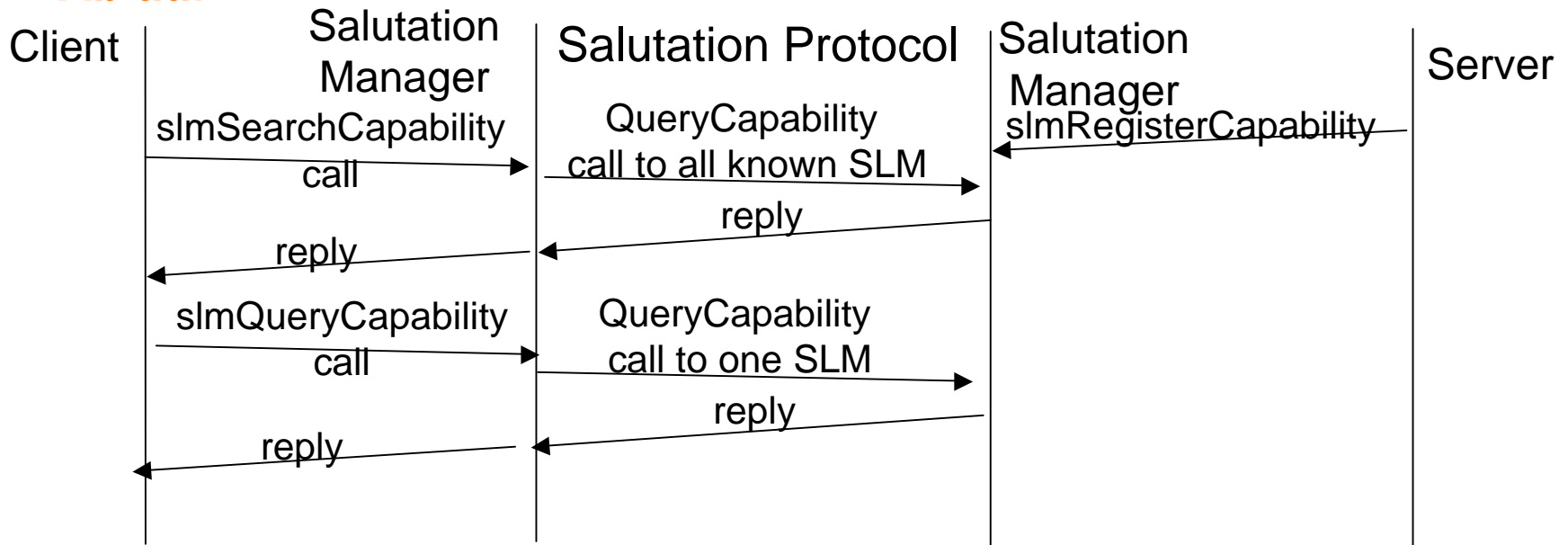


# Salutation Service Discovery

## Aufgaben Salutation Manager (SLM)

- Service Registry
- Service Discovery
- Service Availability
- Service Session Management

## Ablauf



# Salutation Service Discovery

---

## Service Discovery

- slmSearchCapability wird mit Parametern-Muster aufgerufen
- komplette Übereinstimmung oder Aufruf einer „Compare Function“
- Compare-Function muß bei Server bekannt sein
- logische Verbindung (AND,OR) im Ausdruck unterstützt
- Vordefinierte Attribute für Standard-Anwendungen: Drucker, Fax, Voice Message, Personal Information Managment, ....

# Vernetzung: höhere Schichten

---

- Einführung
- Kommunikationsparadigmen
- Jini Service Infrastruktur
- Bluetooth
- Salutation
- **OBEX Object Exchange Protokoll**
- HAVi AV Kontrolle
- UPnP Plug&Play

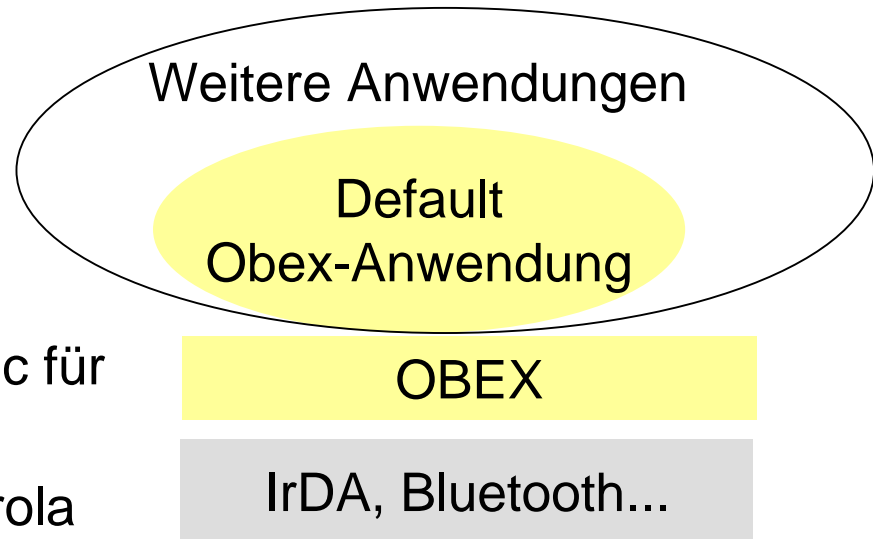
# (Ir)OBEX: IrDA Object Exchange

## Beliebige „Dinge“ austauschen

- Protokoll für den Austausch, das Abfragen, die Anforderung von Objekten und den damit verbundenen Diensten
- Bei Bluetooth: OBEX, setzt auf RFCOMM auf
- Dienste primär Datenübertragungsdienste
- Spezifikation besteht aus zwei Teilen:
  - Beschreibung der Objekte
  - Kommunikationsprotokoll
- Einfaches Protokoll, deshalb:

## SyncML

- Darauf aufsetzende Standard-Sync für PDA, Mobiltelefone
- Palm, Ericsson, IBM, Psion, Motorola



# (Ir)OBEX

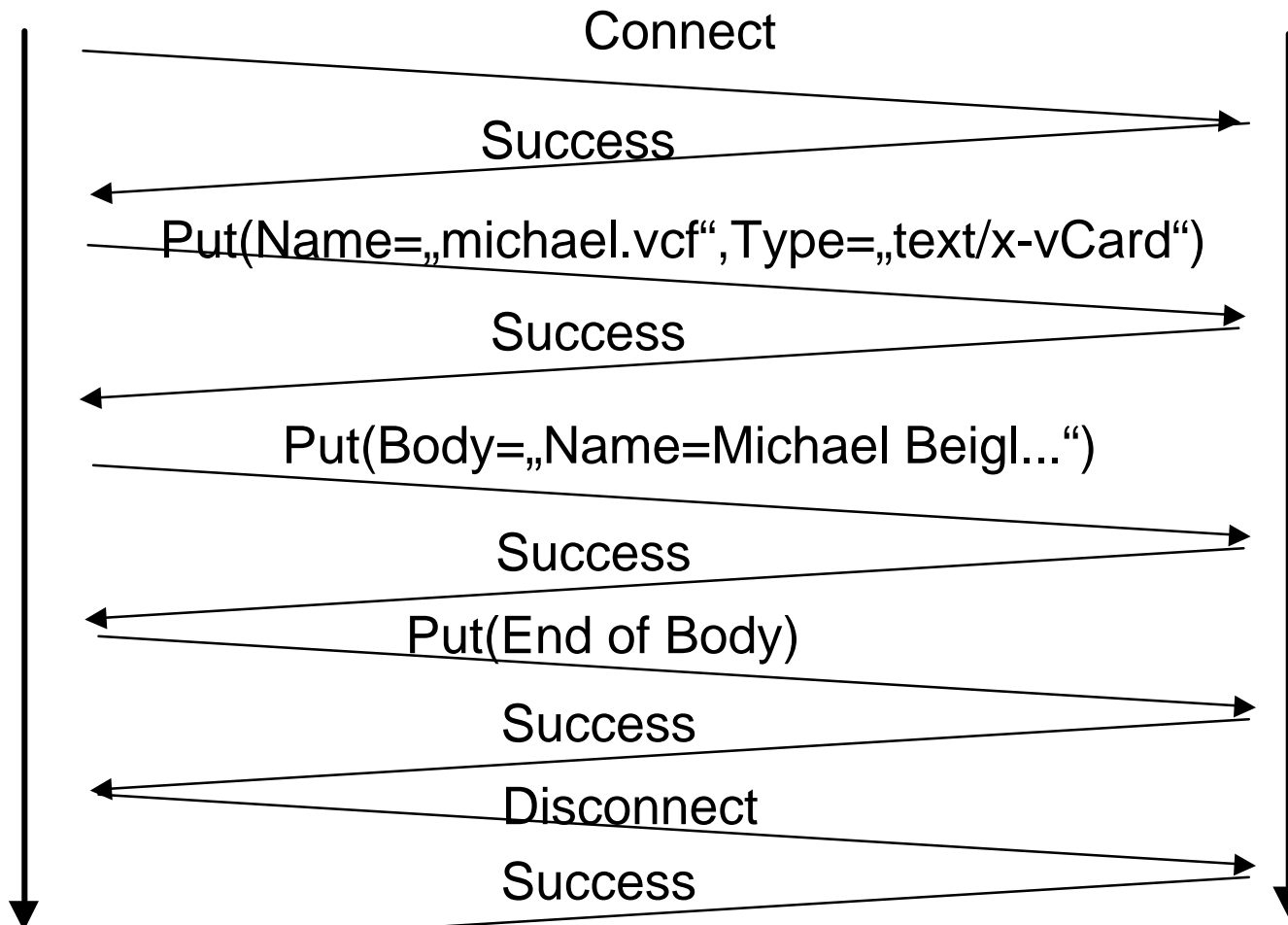
---

## Ablauf Kommunikation

- Geräte bieten Dienste in Form von Objekten an
- Client erfragt einen Dienst (request) und erhält vom Server eine Antwort (response)
- Sitzungsorientiertes Protokoll
- Bsp: Client erfragt, ob er eine vCard ablegen darf
  
- Paket: Opcode len Objekte
- Objektmodell definiert Objektbezeichner und Objekte
- Modell besteht aus einer Liste Tupeln der Form <Bezeichner&Format><Objekt>
- Tupel sind einfach zu parsen
- „Byte“-Kodierung statt Text-Kodierung orientiert sich an leistungsschwachen Geräten

# OBEX

## Vereinfachter Ablauf einer Sitzung



# Vernetzung: höhere Schichten

---

- Einführung
- Kommunikationsparadigmen
- Jini Service Infrastruktur
- Bluetooth
- Salutation
- OBEX Object Exchange Protokoll
- **HAVi AV Kontrolle**
- UPnP Plug&Play

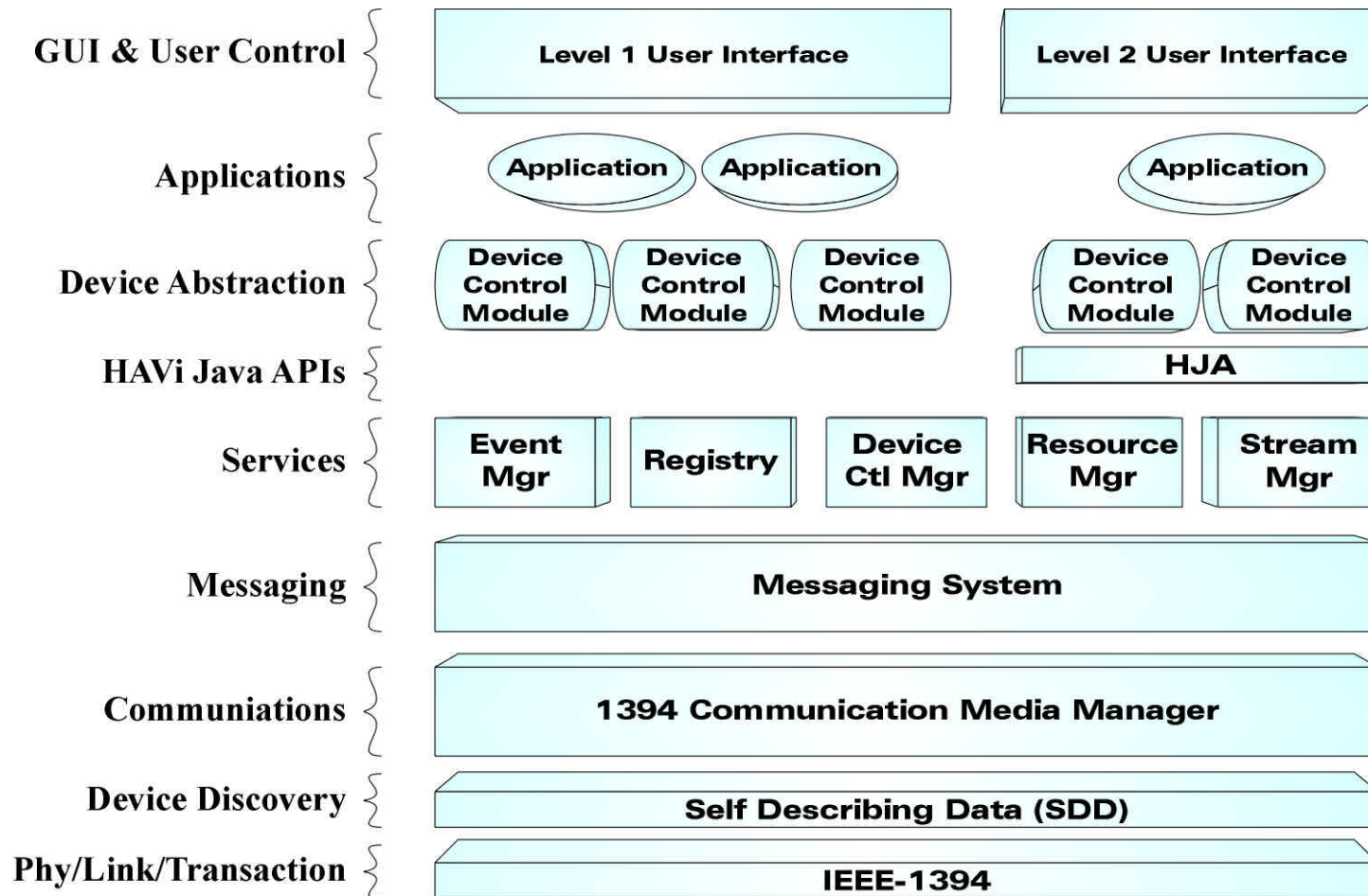
# HAVi

## Eigenschaften HAVi

- Ein Medium für Kontrolle und Daten, Standard von Hitachi, Philips, Sony, Toshiba....
- Verbindungslose Kommunikation basiert auf IEEE1394 vi Communication Manager
- Geräte = Objekte, Methoden = Funktionen, „RFC“ über Pakete
- IAVs (Intermediate AV devices) (native implementierung)
- FAVs (Full AV devices): Java Runtime
- BAV (Base Audio/Video Devices): nur bytecode upload
- LAVs (Legacy AV devices): Aufrufe müssen von FAV umgesetzt werden
- Device Control Module (DCM) und Functional Component Module (FCM) repräsentieren Device, Funktion des D.
- Java AWT 1.1 und spezielle Klassen, UI Programmierung (Havlets) auf Anwendungsebene



# HAVi Architektur



Quelle:Sharp

# HAVi FCM

---

## Functional component Modules (FCM)

- 1+ innerhalb des Device Control Module (DCM)
- Vordefinition von APIs zu FCM in HAVi Spezifikation
- FCMs setzen abstrakte Info in gerätespezifische Info/Kommando um
- Von dort wird das Kommando an Hardware weitergesendet.
- FCM Beispiele:
  - Tuner FCM: Setzen und Erhalten von Kanälen, Auswahl von Attributen (Musiksender...)
  - VCR FCM: PLAY, REC, REW..., Uhrzeit setzen

# Vernetzung: höhere Schichten

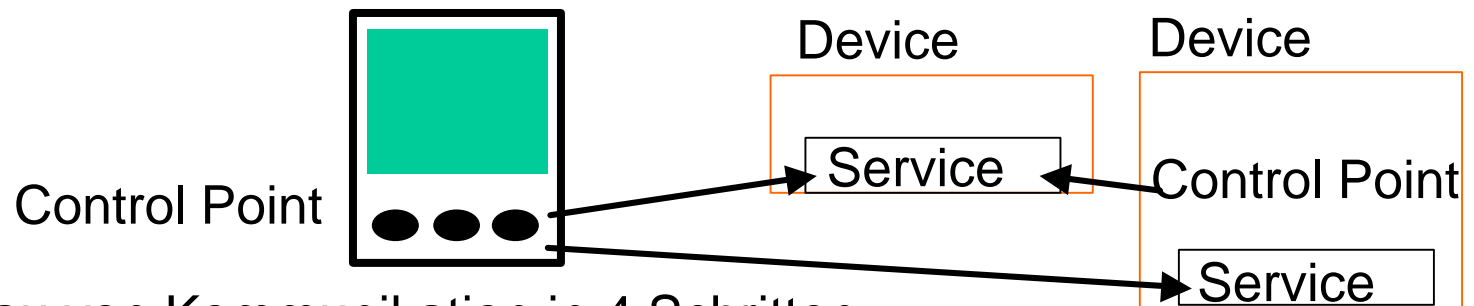
---

- Einführung
- Kommunikationsparadigmen
- Jini Service Infrastruktur
- Bluetooth
- OBEX Object Exchange Protokoll
- HAVi AV Kontrolle
- UPnP Plug&Play

# Universal Plug and Play (UPnP)

## Charakteristik

- Netz zur Ad-hoc Vernetzung von Geräten wie PNP Geräte
- Eingesetzt derzeit vor allem für „NAT Traversal“ in Firewalls/DSL
- Basiert auf IP, benötigt DHCP, XML, HTTP



- Aufbau von Kommunikation in 4 Schritten
- 1) Adressierung, 2) Discovery, 3) Description, 4) (Control, Eventing, Presentation)

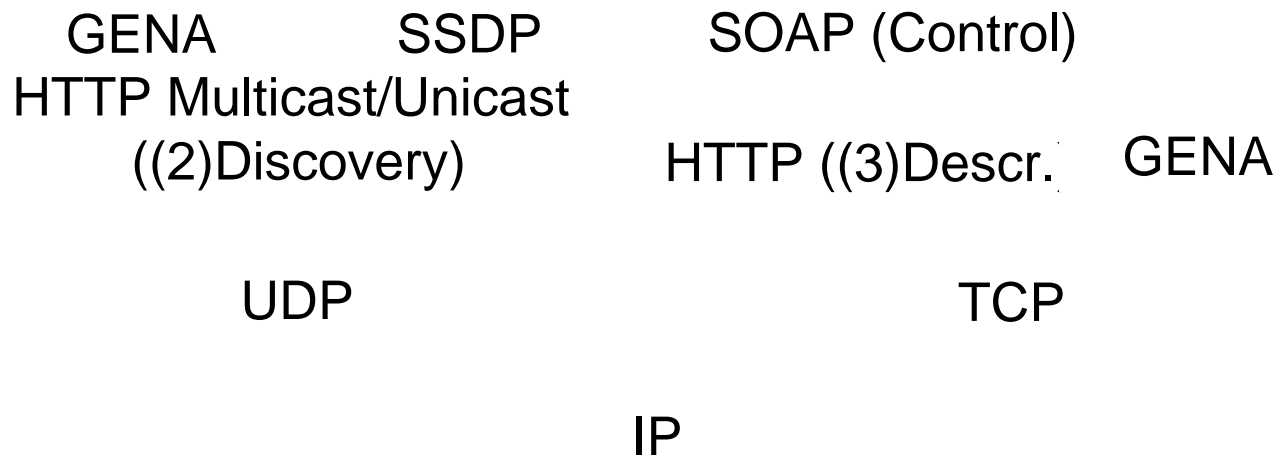
# UPnP Architektur

---

UPnP Vendor Defined

UPnP Forum Defined

UPnP Device Architecture (DCP)(Presentation)



# UPnP Schritte

---

## Adressierung

- DHCP oder Auto IP

## Discovery von Geräten

- SSDP (Simple Service Discovery Protocol): Meldung der Anwesenheit (Announce) oder über vorheriges Search
- LAN Broadcasts oder direktes Ansprechen eines Verzeichnisdienstes (Proxy)
- HTTP/XML basierte verbindungslose Kommunikation z.B. als UDP
- Melden der eigenen Fähigkeiten durch ANNOUNCE-Kommando, enthält zudem URL zu XML Beschreibungsdatei
- Abfrage durch OPTIONS Kommando

## Zudem: Broadcast-DNS Abfrage, DHCP Erweiterung

# UPnP Schritte II

---

## Description

- Beschreibung des Gerätes im XML Format
- Vor allem ID, URL, Hersteller...

## Control

- Simple Object Access Protocol (SOAP)
- Beschreibung Dienste/Aktionen und Parameter in XML Format
- „RPC über HTTP“

## Event

- IETF Draft General Event Notification Architecture (GENA)
- 3 Kommandos/Events:  
Subscribe / Unsubscribe / Notify von Meldungen

# Universal Plug and Play (UPnP)

