MMI 2: Mobile Human-Computer Interaction

Mobile Communication

Prof. Dr. Michael Rohs

michael.rohs@ifi.lmu.de

Mobile Interaction Lab, LMU München

Lectures

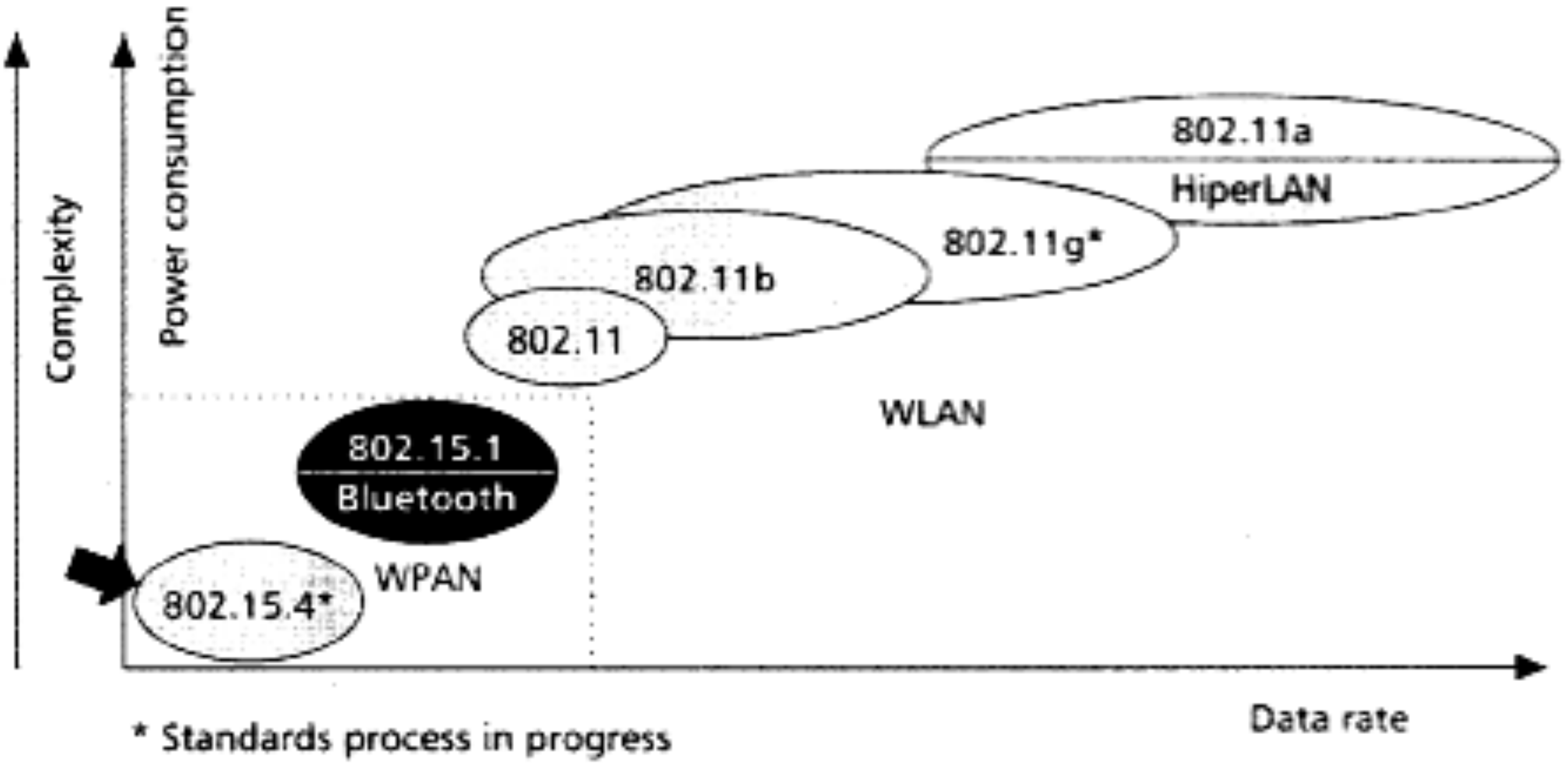
#	Date	Topic
1	19.10.2011	Introduction to Mobile Interaction, Mobile Device Platforms
2	26.10.2011	History of Mobile Interaction, Mobile Device Platforms
3	2.11.2011	Mobile Input and Output Technologies
4	9.11.2011	Mobile Input and Output Technologies, Mobile Device Platforms
5	16.11.2011	Mobile Communication
6	23.11.2011	Location and Context
7	30.11.2011	Mobile Interaction Design Process and Prototyping
8	7.12.2011	Evaluation of Mobile Applications
9	14.12.2011	Visualization and Interaction Techniques for Small Displays
10	21.12.2011	Mobile Devices and Interactive Surfaces
11	11.1.2012	Camera-Based Mobile Interaction 1
12	18.1.2012	Camera-Based Mobile Interaction 2
13	25.1.2012	Sensor-Based Mobile Interaction 1
14	1.2.2012	Sensor-Based Mobile Interaction 2
15	8.2.2012	Exam

Preview

- Wireless mobile communication technologies

- Short range (~10m)
  - Bluetooth
  - ZigBee
- Medium range (~100m)
  - Wireless LAN
- Long range (almost everywhere)
  - GSM, GPRS, UMTS

Operating Space for Wireless Communication Standards



Source: Gutierrez et. al, 2001, IEEE 802.15.4: a developing standard for low-power…

HTTP CLIENTS

HTTP Clients

- Android ships with Apache’s HttpClient
  - http://hc.apache.org/httpclient-3.x/
  - Widely used in J2EE
- Full support for the HTTP protocol
  - GET, POST, HEAD, DELETE, PUT (org.apache.http.client.methods)
- Usage
  - Create HttpClient
  - Instantiate PostMethod or GetMethod
  - Set HTTP parameter name/value pairs
  - Execute the HTTP request
  - Process the HTTP response
- Permissions

```
<uses-permission android:name="android.permission.INTERNET" />
```

HTTP Client Example

```
HttpClient client = new DefaultHttpClient();
HttpGet request = new HttpGet();
request.setURI(new URI("http://code.google.com/android/"));
HttpResponse response = client.execute(request);
BufferedReader in = new BufferedReader(new
        InputStreamReader(response.getEntity().getContent()));
StringBuffer sb = new StringBuffer("");
String line;
String NL = System.getProperty("line.separator");
while ((line = in.readLine()) != null) {
    sb.append(line + NL);
}
in.close();
String page = sb.toString();
```

HTTP Get

- Parameters as part of URL

```
HttpGet method = new HttpGet("http://www.x.com/upload.aspx?
   one=valueGoesHere");
client.execute(method);
```

- Limited length of URL (< 2048 characters)

HTTP Post

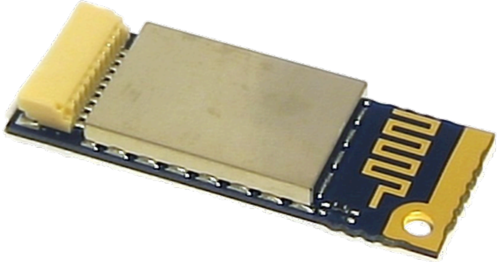
```
HttpClient client = new DefaultHttpClient();
HttpPost request = new HttpPost("http://www.x.com/upload.aspx");

List<NameValuePair> params = new ArrayList<NameValuePair>();
params.add(new BasicNameValuePair("one", "valueGoesHere"));
UrlEncodedFormEntity formEntity =
            new UrlEncodedFormEntity(params);
request.setEntity(formEntity);

HttpResponse response = client.execute(request);

BufferedReader in = new BufferedReader(new
   InputStreamReader(response.getEntity().getContent()));
```

BLUETOOTH

Bluetooth slides partially based on slides by Prof. Dr. F. Mattern, ETH Zurich

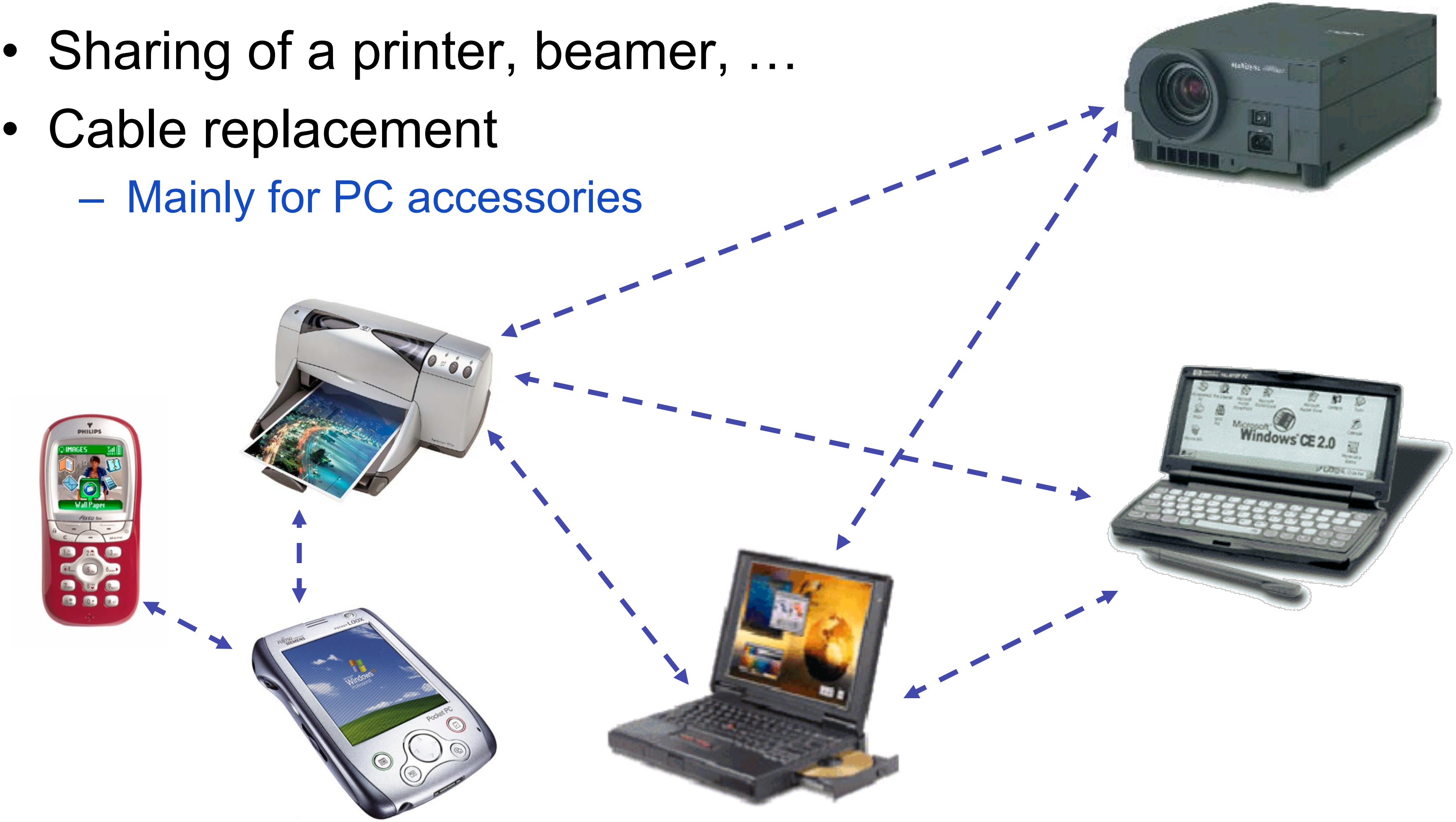
Bluetooth Technology

- Mainly cable replacement for portable devices
  - Seamless connectivity between mobile phones, PDAs, and other electronic devices
  - Simultaneous voice and data
- Ad hoc wireless connectivity
  - "Spontaneous networks", no infrastructure
  - Dynamic discovery of nearby devices and services they offer
- Short-range (10 m)
- Design goals
  - Low cost
  - Small form factor
  - Low power consumption
  - Security



Bluetooth Usage Scenarios: Personal Ad-hoc Networks

- Wireless file transfer
- Sharing of a printer, beamer, …
- Cable replacement
  - Mainly for PC accessories



Bluetooth Usage Scenarios: Proximity Synchronization

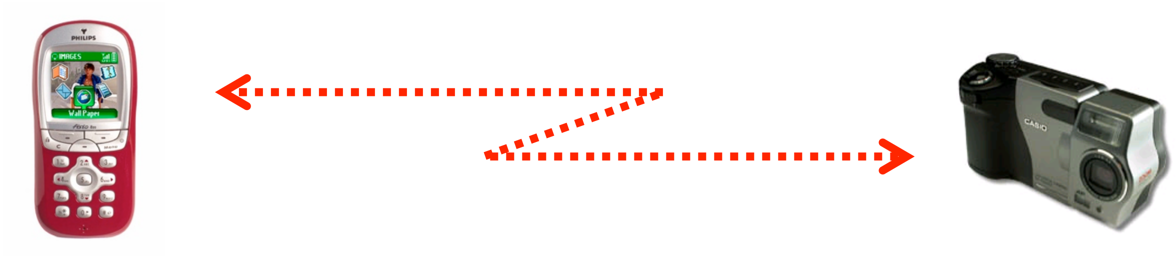
- Synchronize PDAs, cellular phones, mobile PCs, ...
- Personal information management
  - Calendar
  - Phonebook
  - Messages
  - Address list
  - To-do list

- On demand synchronization
  - Business cards
- Automatic synchronization
  - “Hidden computing”

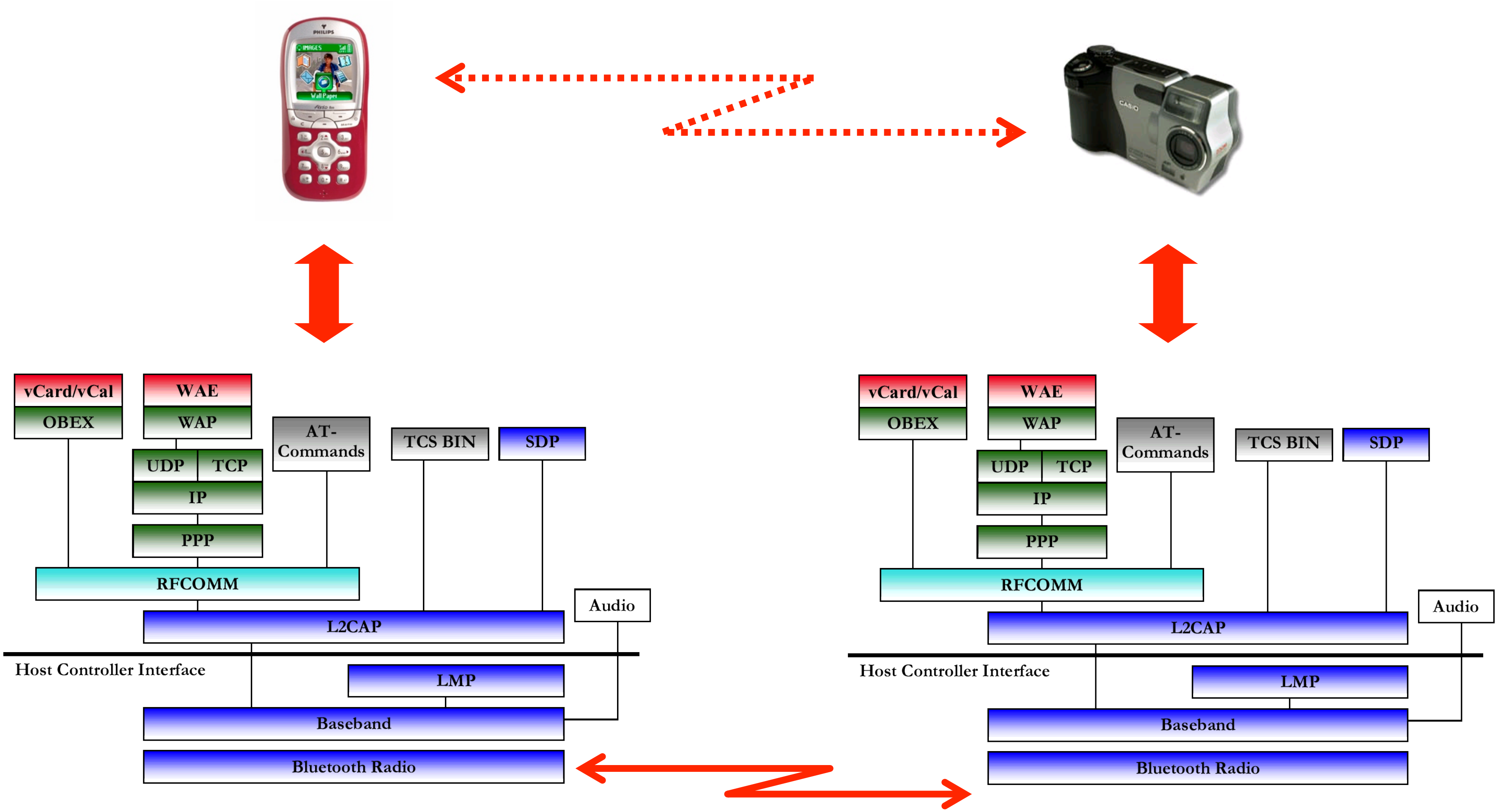
Bluetooth Usage Scenarios: Cordless Headset

- Hand’s free phone calls
- Flexible associations between devices
- Use with a phone
  - Dial by voice
- Use with a PC
  - Write by voice
  - Listen to audio
- Use with a stereo, portable CD player, MP3 player, recording device, ...

Bluetooth Protocol Overview



Bluetooth Protocol Overview

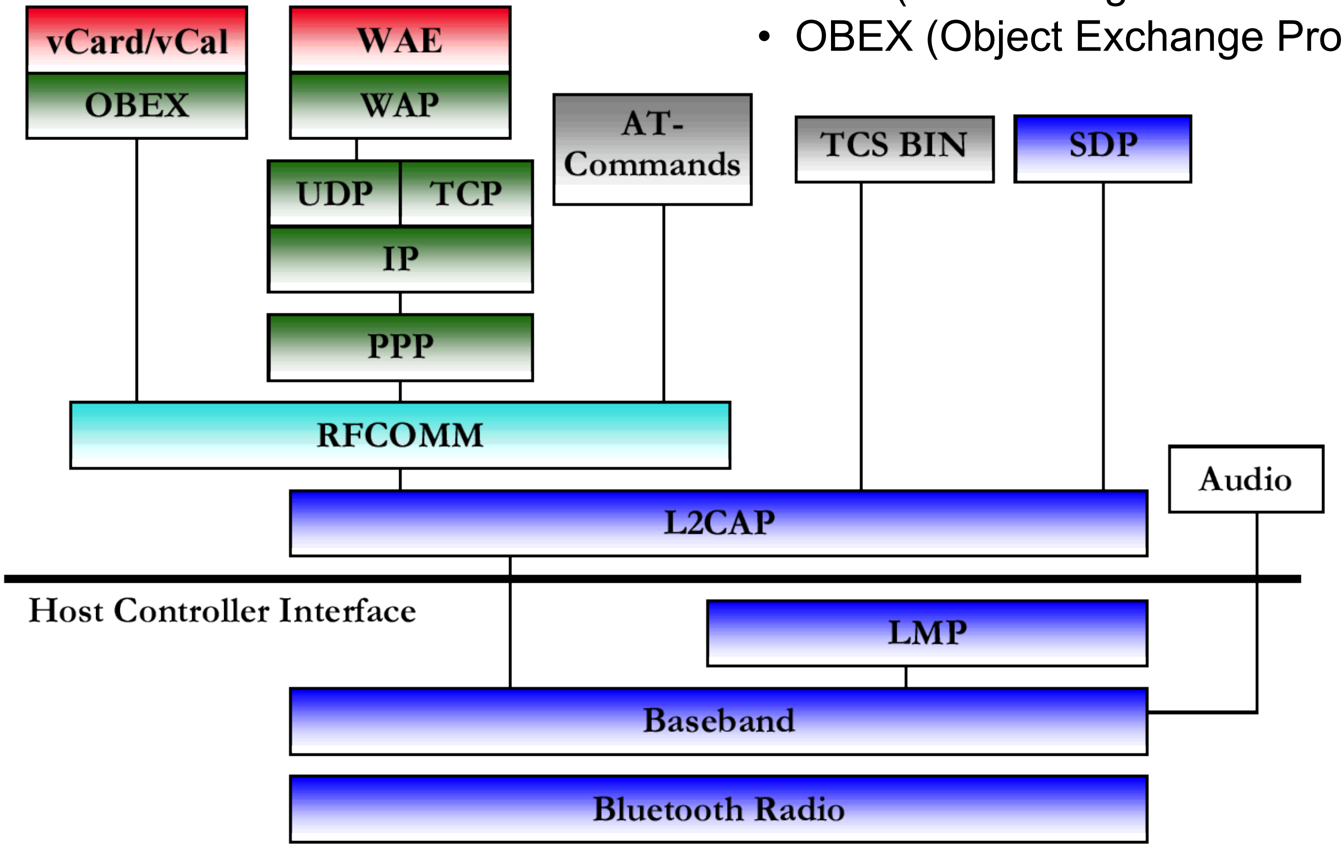


vCard/vCal
OBEX
WAE
WAP
UDP
TCP
IP
PPP
AT-Commands
TCS BIN
SDP
RFCOMM
Audio
L2CAP
Host Controller Interface
LMP
Baseband
Bluetooth Radio

vCard/vCal
OBEX
WAE
WAP
UDP
TCP
IP
PPP
AT-Commands
TCS BIN
SDP
RFCOMM
Audio
L2CAP
Host Controller Interface
LMP
Baseband
Bluetooth Radio

Bluetooth Protocol Stack

- RFCOMM (“RF” Serial Communication protocol)
- SDP (Service Discovery Protocol)
- L2CAP (Logical Link Control & Adaptation Protocol)
- HCI (Host/Controller Interface)
- LMP (Link Management Protocol)
- OBEX (Object Exchange Protocol)

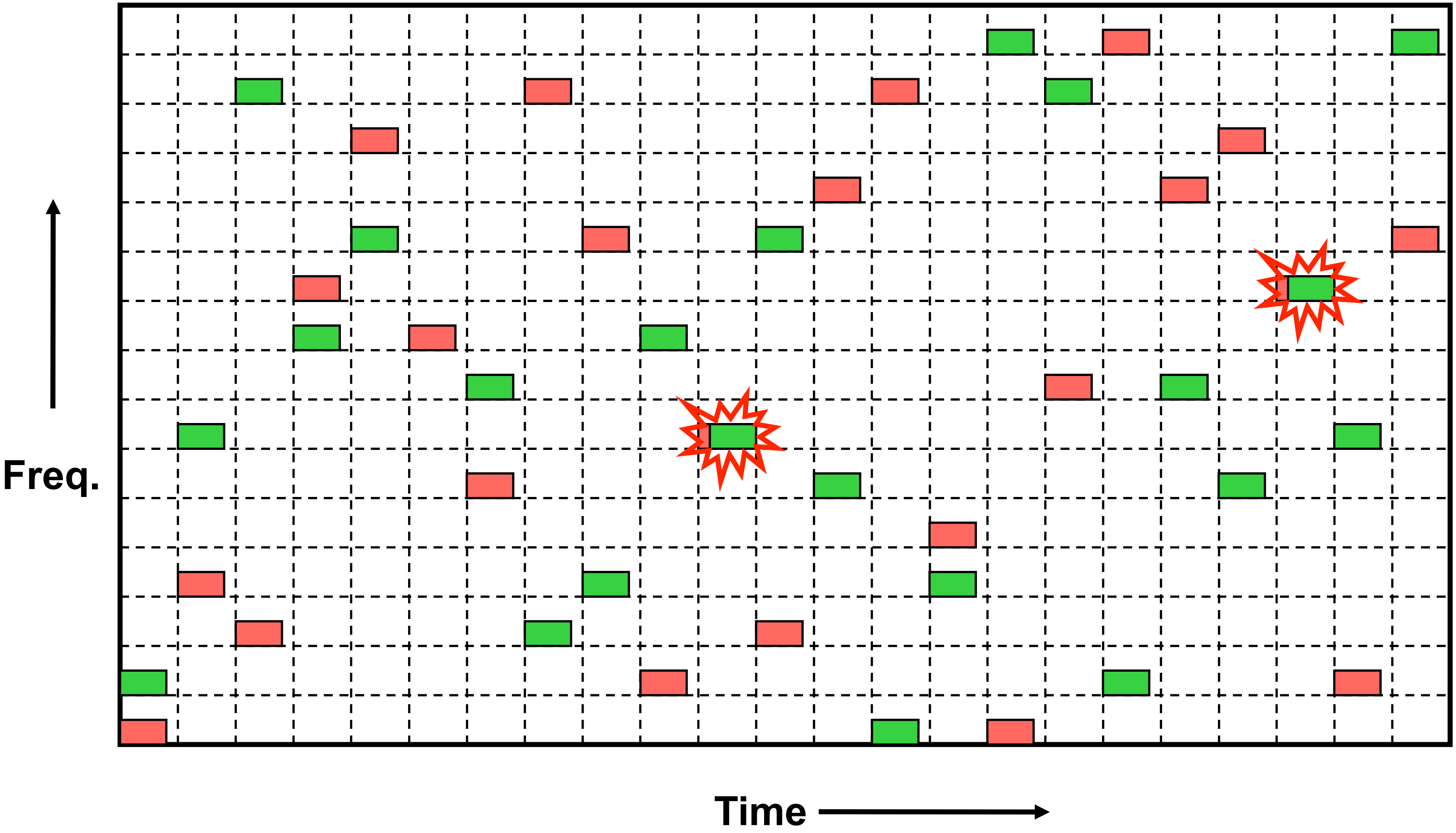


Bluetooth Radio

- Global 2.4 GHz ISM band
  - (2.402 - 2.480 GHz, 79 channels)
  - Frequency hopping 1600 hops/s
- Data rates
  - Version 1.0-1.2: 1 Mbit/s
    - 432 kbit/s (symmetric half duplex)
    - 723.1 kbit/s (asymmetric)
  - Version 2.0 with enhanced data rate (EDR): 3 Mbit/s
    - 2.1 Mbit/s practical transmission rate
  - Version 3.0 + HS: 24 Mbit/s
- Maximum power
  - Class 1: 100 mW (~100 meters)
  - Class 2: 2.5 mW (~10 meters)
  - Class 3: 1 mW (~5 meter)

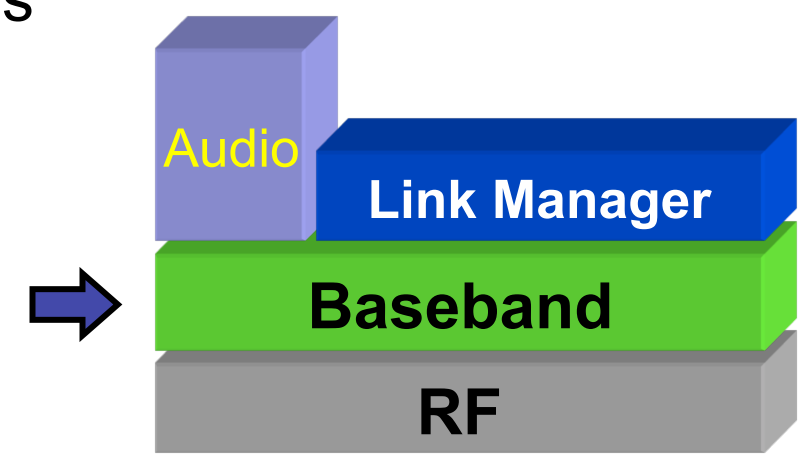
For comparison:
Wireless LAN (WiFi, IEEE 802.11)
11 or 54 Mbit/s (and more)
100 mW

Frequency Hopping

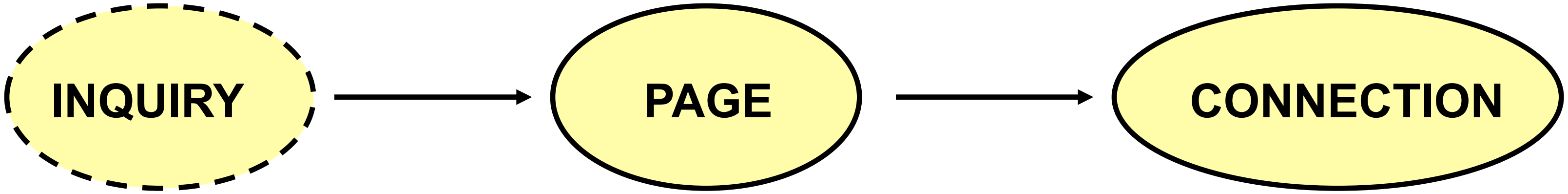


Baseband Layer Responsibilities

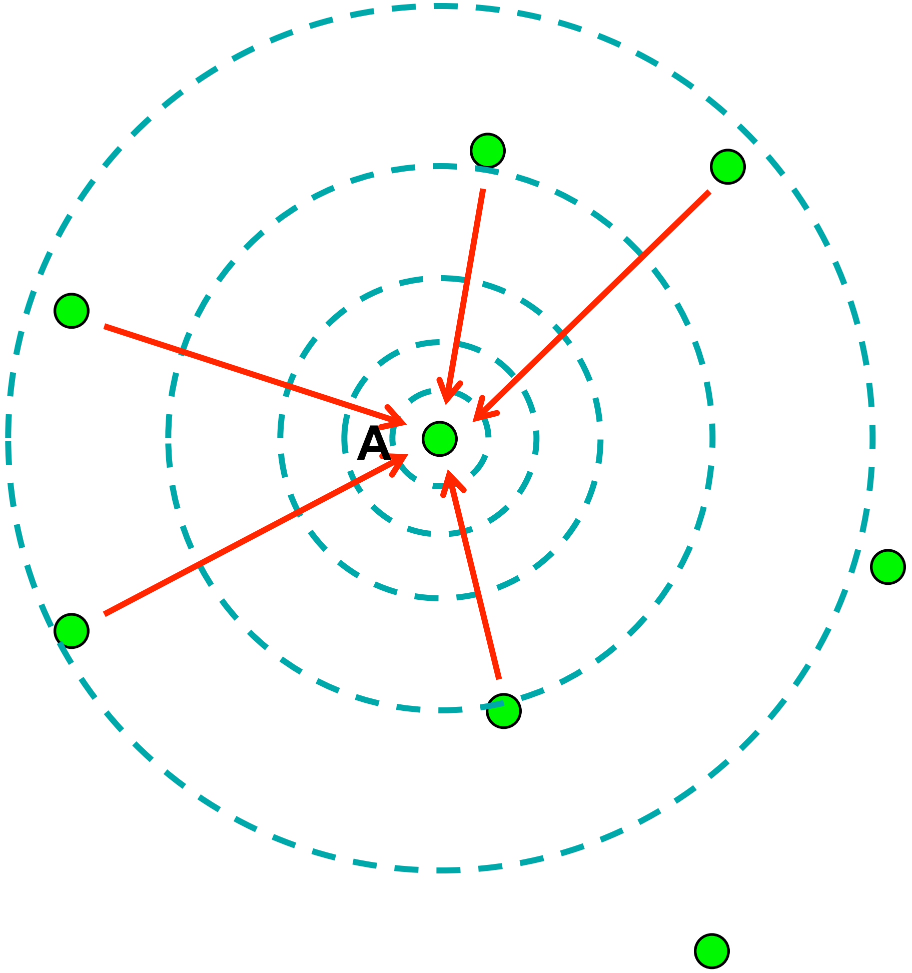
- Synchronization of sender and receiver
  - Time synchronization
  - Frequency synchronization (hopping sequence)
- Searching for and connecting to other devices
- Master and slave roles
- Error handling, retransmissions
- 48-bit Bluetooth device address
  - Compatible to IEEE 802 MAC (e.g., “Ethernet address”)
  - Example: 00:04:3E:23:46:C0



Connection Establishment

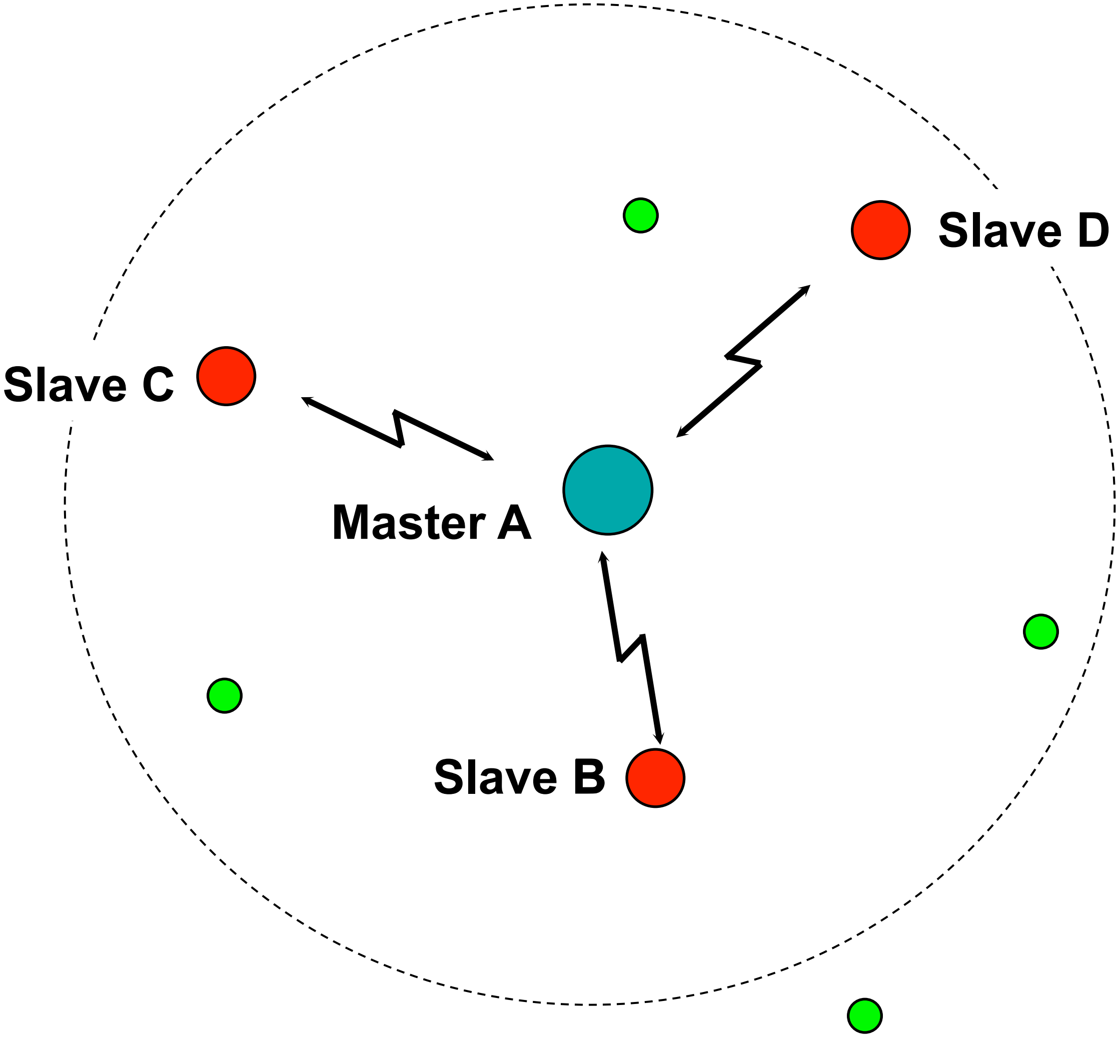


Inquiry: Looking for Nearby Devices



- Responses include:
  - Device Address
  - Class of Device

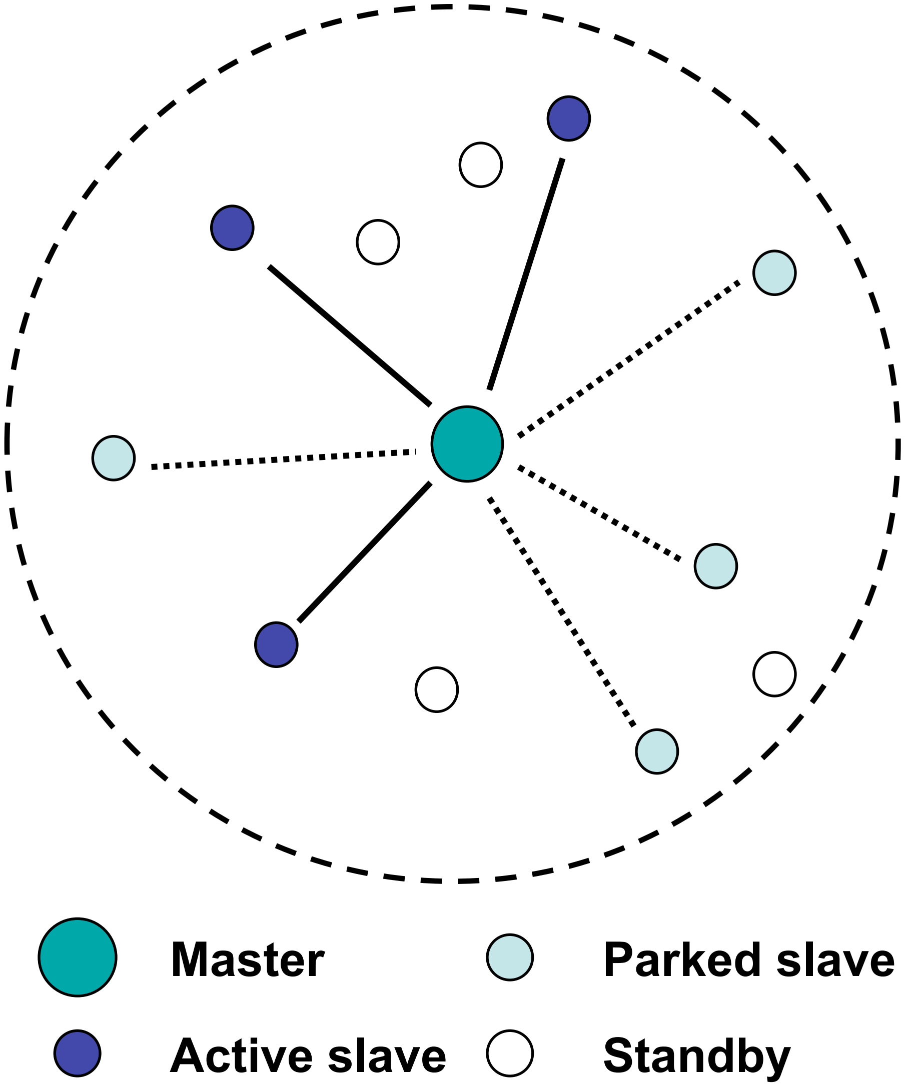
Paging: Establishing a Connection



- Done for each device independently
- Paging device becomes master

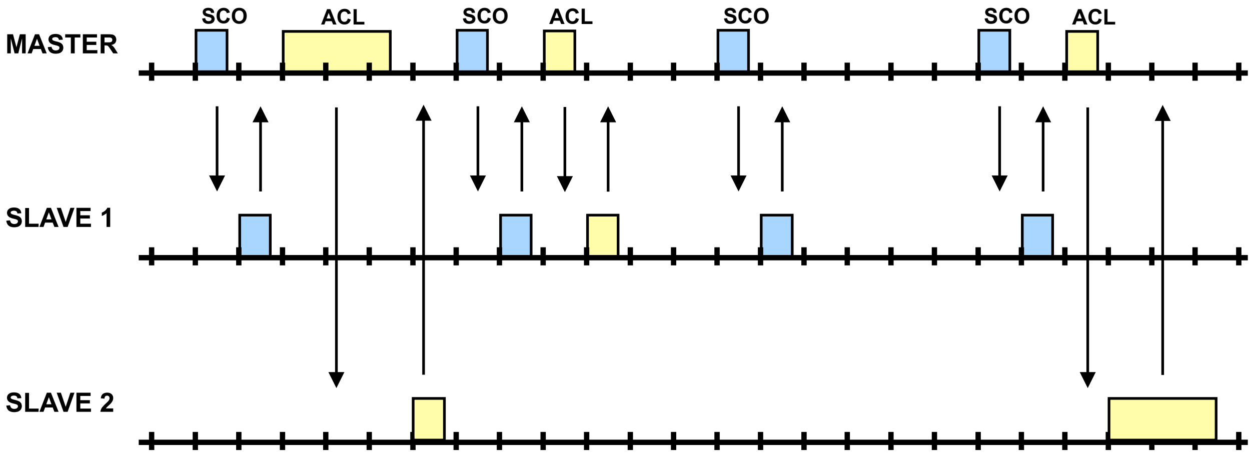
Piconet

- Star Topology
  - 1 master
  - up to 7 active slaves
  - up to 255 parked slaves
- Master
  - Determines hopping scheme and timing
  - Administers piconet
- Logical Channels
  - Asynchronous, packet oriented
  - Synchronous, connection-oriented (voice)

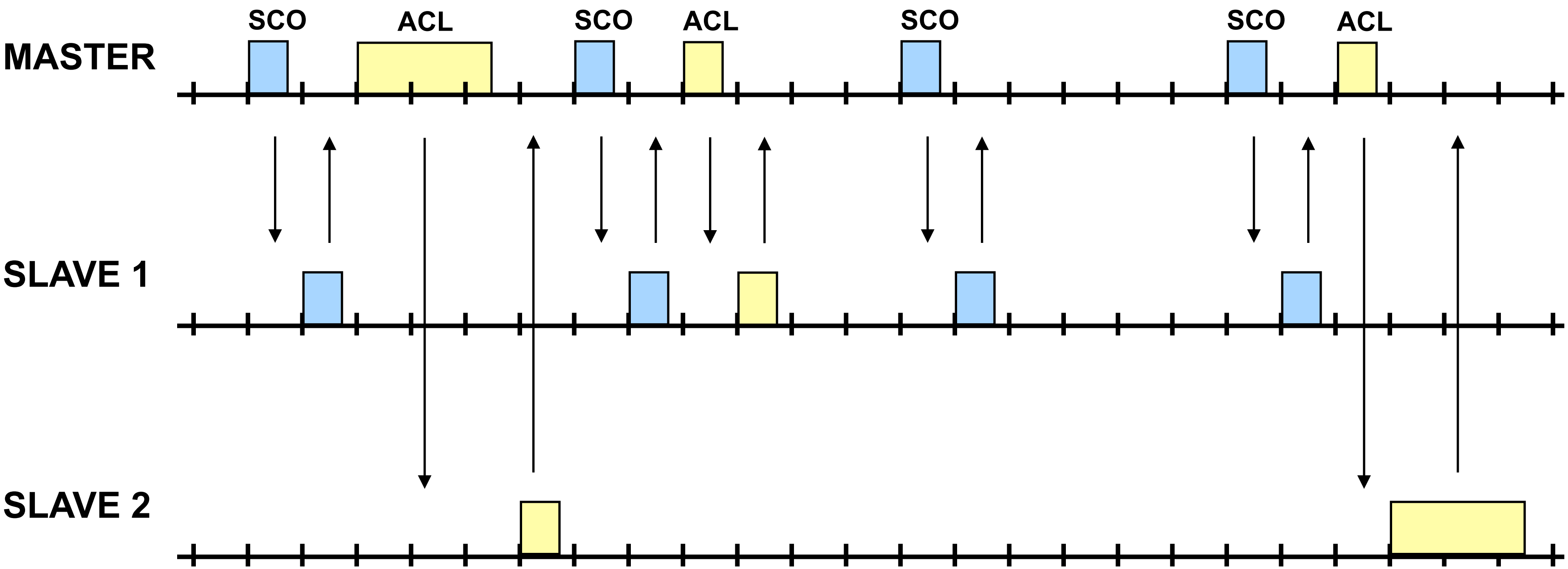


Baseband Link Types

- Synchronous connection-oriented (SCO) link
  - “Circuit-switched”: periodic slot assignment
  - Typically for voice
- Asynchronous connection-less (ACL) link
  - “Packet switching” with acks
  - Variable packet size (1-5 slots)



Mixed Synchronous / Asynchronous Communication



SCO: Synchronous Connection-Oriented link

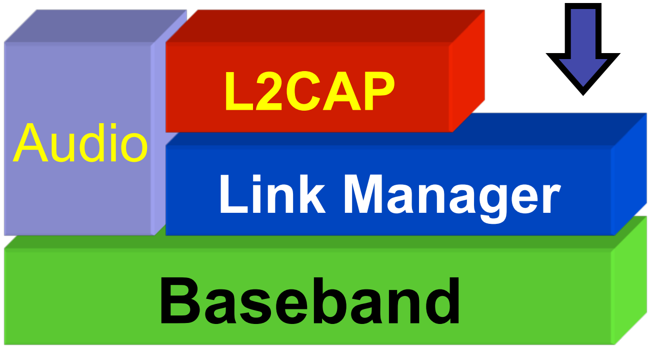
- uses reserved time slots

ACL: Asynchronous Connection-Less link

- uses remaining time slots

Link Manager Responsibilities

- Authentication
  - Only accept connections from trusted devices
- Encryption
- Management of the piconet
  - Master-slave switch
  - Allocating AMA addresses
  - Tearing down connections when slaves leave piconet
  - Exchange of control signals with link managers of other devices (LMP: Link Management Protocol)
- Handling of low power modes (sniff, hold, park)
  - Only listen for synchronization packets

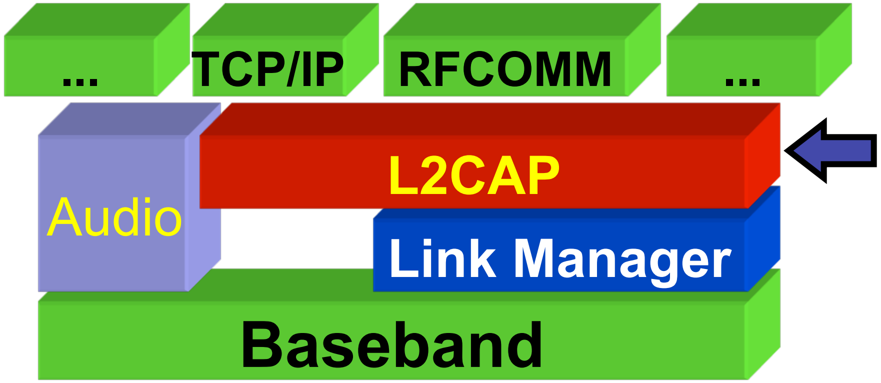


Bluetooth Security

- Important in ad hoc, wireless, RF environments
  - Fast frequency hopping (79 channels)
  - Low transmit power (range < 10m for Class 3)
- Baseband Specification defines security procedures to
  - Authenticate devices (mandatory feature)
  - Encrypt data on link (optional feature)
- Pairing
  - Establish a trusted relationship between two devices by establishing a shared secret
- Link layer encryption
  - Symmetric stream cipher
  - Both SCO and ACL packets can be encrypted

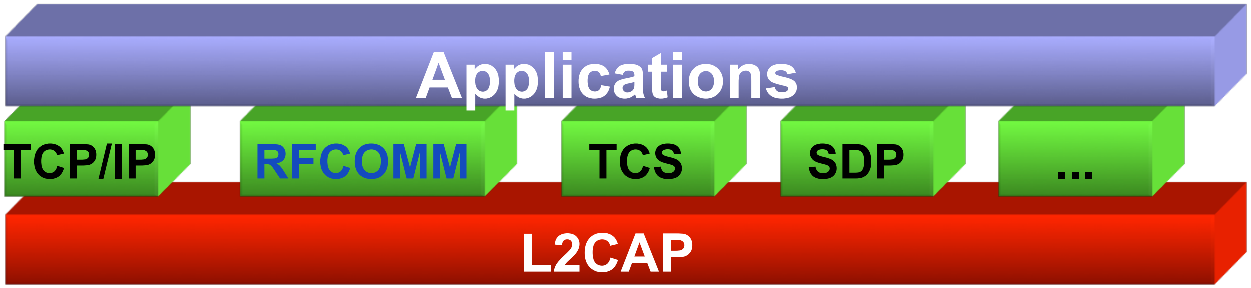
Link Layer Control & Adaptation (L2CAP)

- Data link protocol on top of the baseband
- Upper layers usually do not see the master-slave roles, but use peer-to-peer communication



- Channel abstraction
- Protocol multiplexing for a single “air interface”
- Connection-oriented & connectionless data services
- Packet segmentation and reassembly

RFCOMM



- Emulates a serial port (RS-232 protocol)
- In-sequence, reliable delivery of serial stream
- Enables cable replacement scenarios
- Allows multiple “channels” between two devices (multiplexing via L2CAP)

Android Bluetooth API (since 2.0)

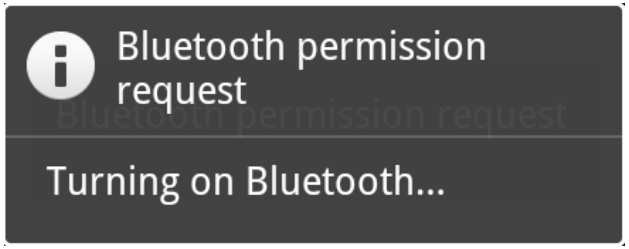
- Package android.bluetooth
- Discover devices and use their services
  - BluetoothAdapter: startDiscovery()
- Local adapter and remote devices
  - BluetoothAdapter represents local device
    - Adding service records to the service database
  - BluetoothDevice represents remote device
    - Querying remote services using UUIDs
- Communication
  - Client: BluetoothSocket
  - Server: BluetoothServerSocket

Class BluetoothAdapter

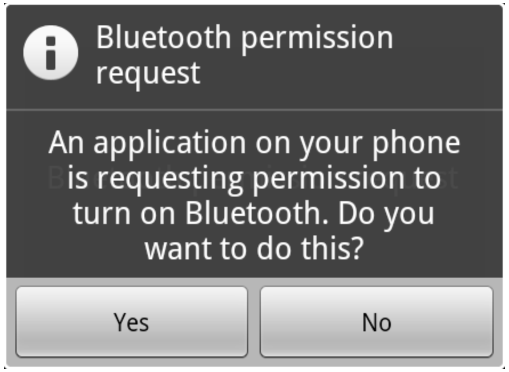
- BluetoothAdapter represents local Bluetooth adapter
  - BluetoothAdapter.getDefaultAdapter()
  - getName(), getAddress(): local name and adress
- Get remote devices
  - BluetoothDevice getRemoteDevice(String macAddress)
  - Set<BluetoothDevice> getBondedDevices()
- Device discovery
  - startDiscovery(): start to find nearby devices
  - Register for ACTION_FOUND intent to be notified as remote Bluetooth devices are found
- Permissions
  - android.permission.BLUETOOTH
  - android.permission.BLUETOOTH_ADMIN

Android Bluetooth Initialization

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);
    ...
    adapter = BluetoothAdapter.getDefaultAdapter();
    if (adapter == null) finish(); // no Bluetooth → end activity

    if (adapter.isEnabled()) testBluetooth();
    } else {
        Intent i = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
        startActivityForResult(i, REQUEST_ENABLE_BT);
    }
}
```


Bluetooth permission request

Turning on Bluetooth...

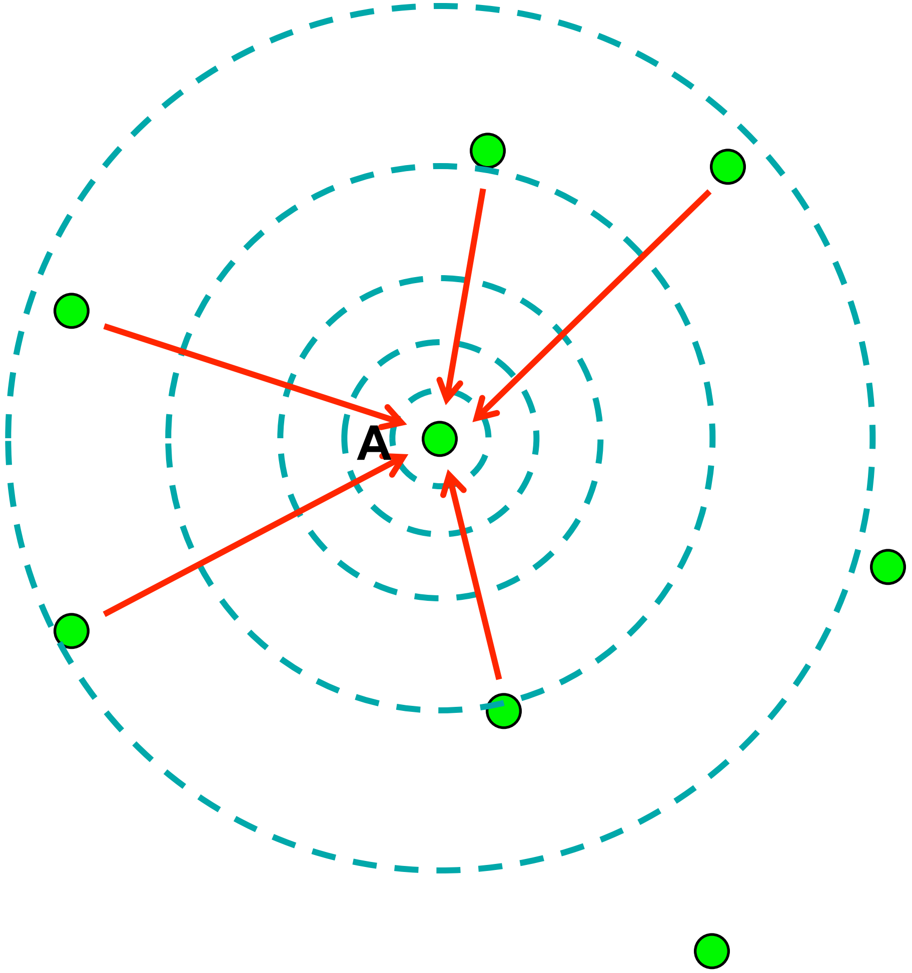

Bluetooth permission request

An application on your phone is requesting permission to turn on Bluetooth. Do you want to do this?

Result of Bluetooth Enable Activity

```
Intent i = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
startActivityForResult(i, REQUEST_ENABLE_BT);
...

protected void onActivityResult(
  int requestCode, int resultCode, Intent data)
{
  if (requestCode == REQUEST_ENABLE_BT) {
    if (resultCode == RESULT_OK) {
      testBluetooth();
    } else {
      finish(); // failed → end activity
    }
}  }
```

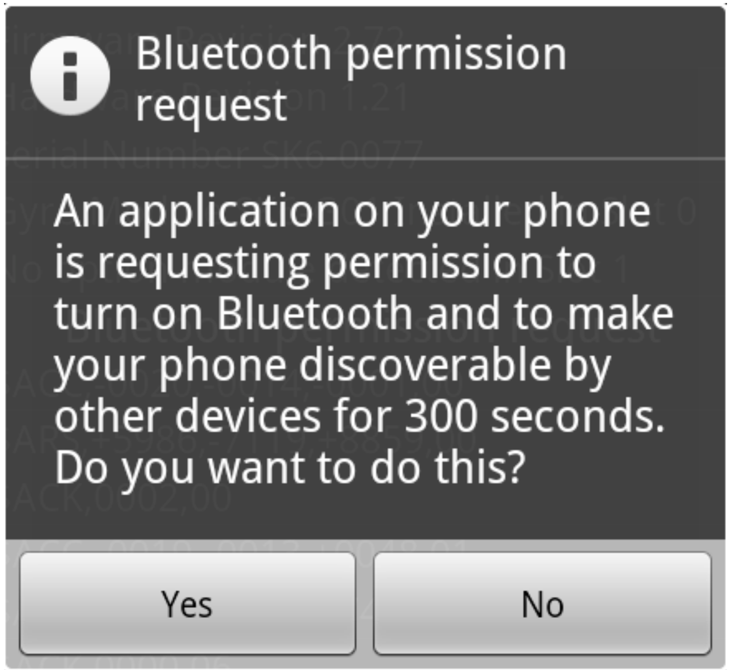
Inquiry: Looking for Nearby Devices



- Responses include:
  - Device Address
  - Class of Device

Make Yourself Discoverable

```
Intent intent = new
   Intent(BluetoothAdapter.ACTION_REQUEST_DISCOVERABLE);
intent.putExtra(
   BluetoothAdapter.EXTRA_DISCOVERABLE_DURATION, 300);

startActivity(intent);
```



Bluetooth permission request

An application on your phone is requesting permission to turn on Bluetooth and to make your phone discoverable by other devices for 300 seconds. Do you want to do this?

Discover Other Devices

- Asynchronously, multiple seconds
- Register “Broadcast Receiver” for discovery events

```
IntentFilter filter = new IntentFilter(BluetoothDevice.ACTION_FOUND);
registerReceiver(deviceFoundReceiver, filter);

filter = new IntentFilter(BluetoothAdapter.ACTION_DISCOVERY_STARTED);
registerReceiver(deviceFoundReceiver, filter);

filter = new IntentFilter(BluetoothAdapter.ACTION_DISCOVERY_FINISHED);
registerReceiver(deviceFoundReceiver, filter);


adapter.startDiscovery();
```

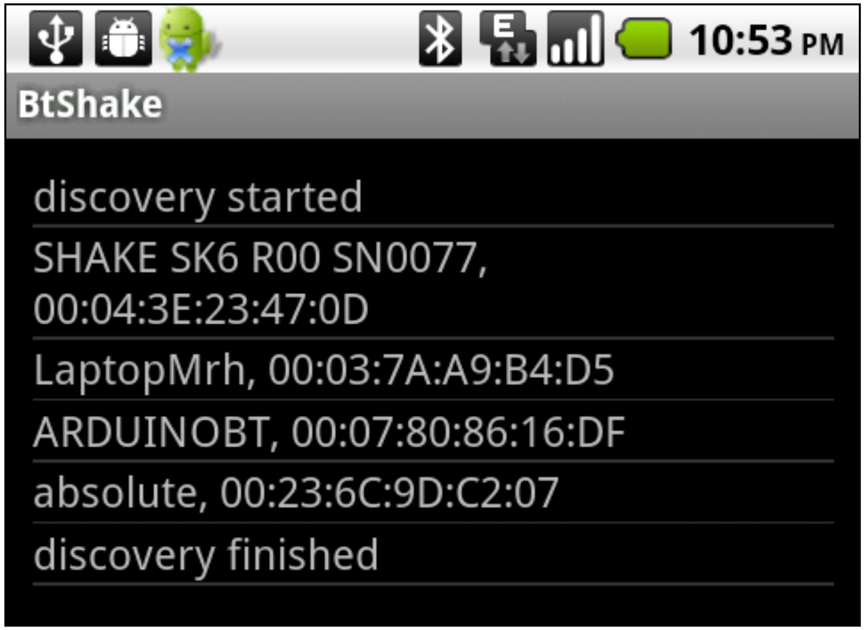
Broadcast Receiver for Discovery Events

```
BroadcastReceiver deviceFoundReceiver = new BroadcastReceiver() {
public void onReceive(Context context, Intent intent) {
String a = intent.getAction();
if (BluetoothDevice.ACTION_FOUND.equals(a)) {
     BluetoothDevice device;
     device = intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
     listAdapter.add(device.getName() + ", " + device.getAddress());
} else if (BluetoothAdapter.ACTION_DISCOVERY_STARTED.equals(a)) {
   listAdapter.add("discovery started");
} else if (BluetoothAdapter.ACTION_DISCOVERY_FINISHED.equals(a)) {
   listAdapter.add("discovery finished");
} } };
```

Broadcast Receiver for Discovery Events

- Output events as list:

```
private ListView listView = null;
private ArrayAdapter<String> listAdapter = null;
...
listView = (ListView) findViewById(R.id.list_view);
listAdapter = new ArrayAdapter<String>(this, R.layout.list_item);
listView.setAdapter(listAdapter);

...

listAdapter.add(device.getName()
    + ", " + device.getAddress());
```



```
discovery started
SHAKE SK6 R00 SN0077,
00:04:3E:23:47:0D
LaptopMrh, 00:03:7A:A9:B4:D5
ARDUINOBT, 00:07:80:86:16:DF
absolute, 00:23:6C:9D:C2:07
discovery finished
```

Service Discovery Protocol (SDP)

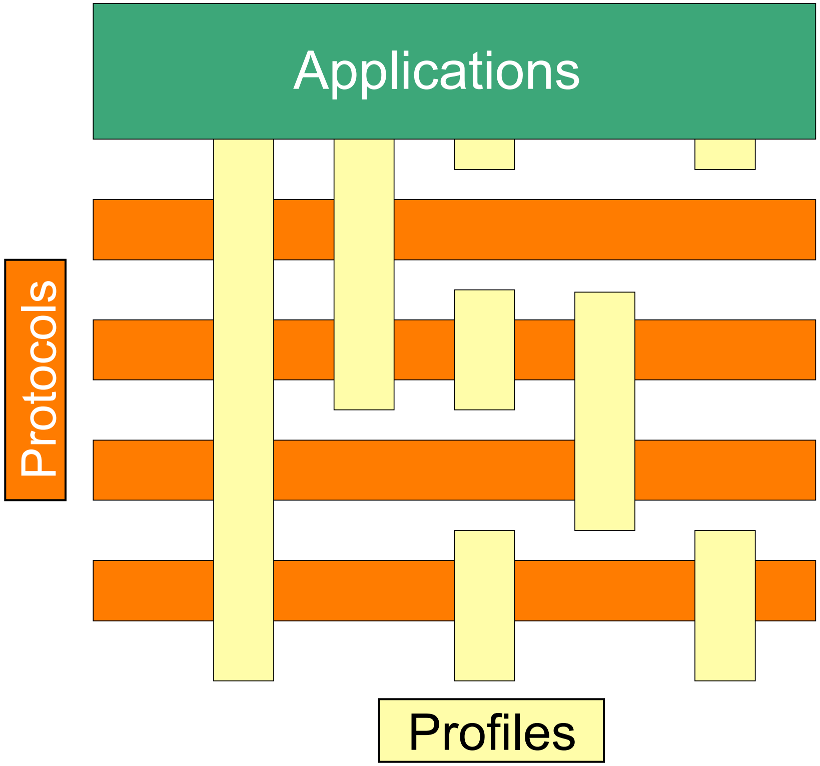
- Devices may spontaneously join / leave a network
  - Goal: self configure without manual intervention
  - Devices should discover each other, negotiate their needs
- SDP defines an inquiry/response protocol for discovering services
  - Searching for services
  - Browsing services
- SDP has no notification service
  - No automatic notification of new services or services becoming unavailable

SDP: Service Description

- Each service is represented by a service record
- Attributes in the service record describe the service
- Attributes represent
  - Unique identifier
  - Service class information (e.g., “printer” or “audio service”)
  - Access protocol information
  - Human-readable service description
- Attribute values
  - Universally unique identifiers (UUIDs), strings, booleans, integers, URLs, etc.
- 128-bit UUID example (Serial Port Profile, SPP):
  - 00001101-0000-1000-8000-00805F9B34FB

Bluetooth Profiles

- Vertical slice through the protocol stack
- Specification for interoperable applications
- A Bluetooth device supports one or more profiles
- Example profiles
  - **Serial port**
  - LAN access
  - File transfer
  - Headset
  - Dial-up networking
  - Fax
  - Cordless telephony



Bluetooth RFCOMM in Android

- Server (device A)
  - Publish serial port service in local SDP database
  - Specify UUID for serial port service
  - Specify name for service
  - System assigns RFCOMM channel number

- Client (device B)
  - Client knows device address of server (discovery)
  - Specifies UUID of required service
  - Client adapter requests RFCOMM channel number from server

Bluetooth RFCOMM in Android

- Server (device A)

```
BluetoothAdapter ba = BluetoothAdapter.getDefaultAdapter();
UUID uuid = UUID.fromString("00001101-0000-1000-8000-00805F9B34FB");
BluetoothServerSocket bss =
    ba.listenUsingRfcommWithServiceRecord("mysvc", uuid);
BluetoothSocket bs = bss.accept(); // blocks until connection established
InputStream is = bs.getInputStream();
```

- Client (device B)

```
BluetoothAdapter ba = BluetoothAdapter.getDefaultAdapter();
BluetoothDevice bd = ba.getRemoteDevice("00:08:1B:CA:D6:38");
UUID uuid = UUID.fromString("00001101-0000-1000-8000-00805F9B34FB");
BluetoothSocket bs = bd.createRfcommSocketToServiceRecord(uuid);
bs.connect();
OutputStream os = bs.getOutputStream();
```

Bluetooth RFCOMM in Android

- BluetoothServerSocket bss = ba.listenUsingRfcommWithServiceRecord("mysvc", uuid);
  - Creates service record with
    - Name = mysvc
    - UUID = <uuid>
    - RFCOMM channel = <auto-assigned RFCOMM channel>
  - Enters service record in SDP database of local device

- BluetoothSocket bs = bd.createRfcommSocketToServiceRecord(uuid);
  - Queries remote SDP server using <uuid>
  - Obtains matching SDP service record
  - Connects to remote service using RFCOMM channel

SHAKE SK6 / SK7 Sensor Module

- Movement sensing and vibrotactile feedback
- Targeted at human-computer interaction
  - Linear and rotational movements
  - Absolute orientation / direction
  - Human body proximity
  - Haptic sensations (programmable vibrotactile display)
- Characteristics
  - Very low noise sensors
  - Simple ASCII protocol
  - Programmable (selection of sensors and filters)
- Communication via Bluetooth RFCOMM
  - Easy connection to other hardware

SHAKE SK6 / SK7 Sensor Module

- Sensors
  - 3-axis accelerometer (±2g or ±6g, resolution 1mg)
  - 3-axis gyroscope (±500deg/s, resolution 0.1deg/s)
  - 3-axis magnetometer (±2 Gauss, resolution 1mGauss)
  - 2 analog inputs (0-2.75V, resolution 1mV, >12 bits)
  - 2 capacitive sensors (<10mm body proximity)
  - 3-position jog dial
- Actuators
  - Vibrating motor with braking capability
- Internal filters for smoothing sensor data
- Real-time clock for precise time stamping

Problem: SDP not always available

- Elegant, but no way to specify RFCOMM channel number explicitly
- Some hardware (SHAKE SK6, Arduino) does not fully implement SDP
  - Use fixed RFCOMM channel number 1

- Use non-official API and Java reflection

```
BluetoothDevice device = adapter.getRemoteDevice("00:04:3E:23:47:0D");
Method m = device.getClass().getMethod(
        "createRfcommSocket", new Class[] { int.class });
socket = (BluetoothSocket) m.invoke(device, Integer.valueOf(1));
```

Blocking I/O

- Bluetooth I/O calls are blocking → separate thread
- Problem: separate thread cannot update GUI → Handlers
- Worker Thread (handler created by main thread):

```
String s = in.readLine(); // from Bluetooth InputStream, blocking
Message msg = handler.obtainMessage(MY_MESSAGE_TYPE, s);
handler.sendMessage(msg);
```

- Main (GUI) Thread:

```
private Handler handler = new Handler() {
    public void handleMessage(Message msg) {
        if (msg.what == MY_MESSAGE_TYPE) {
            String line = (String) msg.obj;
            listAdapter.add(line);
}  }  };
```

Inter-Thread Communication: Message Queues, Handlers

- Each thread can have a message queue
  - Main thread has message queue
- Each message queue can have zero or more handlers
  - New handler gets attached to message queue of current thread
- Handlers
  - Sending messages to a message queue
    - handler.sendMessage(msg);
  - Handling messages from a message queue
    - **public void** handleMessage(Message msg) { ... }
- Uses
  - Schedule messages for execution at some point of time
  - Enqueue actions for execution by another thread

Gracefully Shutting Down Connection

- Activity:

```
protected void onDestroy() {
   // inform connection thread
   connectionThread.shutdown();
   try {
      // wait for thread to terminate
      connectionThread.join();
   } catch (InterruptedException e) {}
   super.onDestroy();
}
```

- Connection thread:

```
public void shutdown() {
    running = false;
}
public void run() {
   ...
   try {
      running = true;
      while (running) { readAndShow(in); }
      in.close();
      out.close();
      socket.close();
   } catch (IOException e) {}
}
```

Competing Technologies

- IrDA („Infrared Data Association“)
  - 4 Mbit/s
  - Narrow and conical transmission shape
  - Requires line of sight
  - 1 m
  - Cheap: < $1 for transceiver module

- Wireless LAN (Wi-Fi: IEEE 802.11b)
  - 11 or 54 Mbit/s (and more)
  - Different modes: central base station / ad hoc
  - 100 mW
  - A priori more expensive and higher power requirements

ZIGBEE

“Personal Area Networks”

- Some wireless applications require even smaller
  - Power consumption
  - Cost
  - Size
- Examples
  - Building automation
  - Interactive toys
  - Smart badges (e.g., for location tracking)
  - Remote controls
  - Wireless sensor networks
  - Smart environments

ZigBee

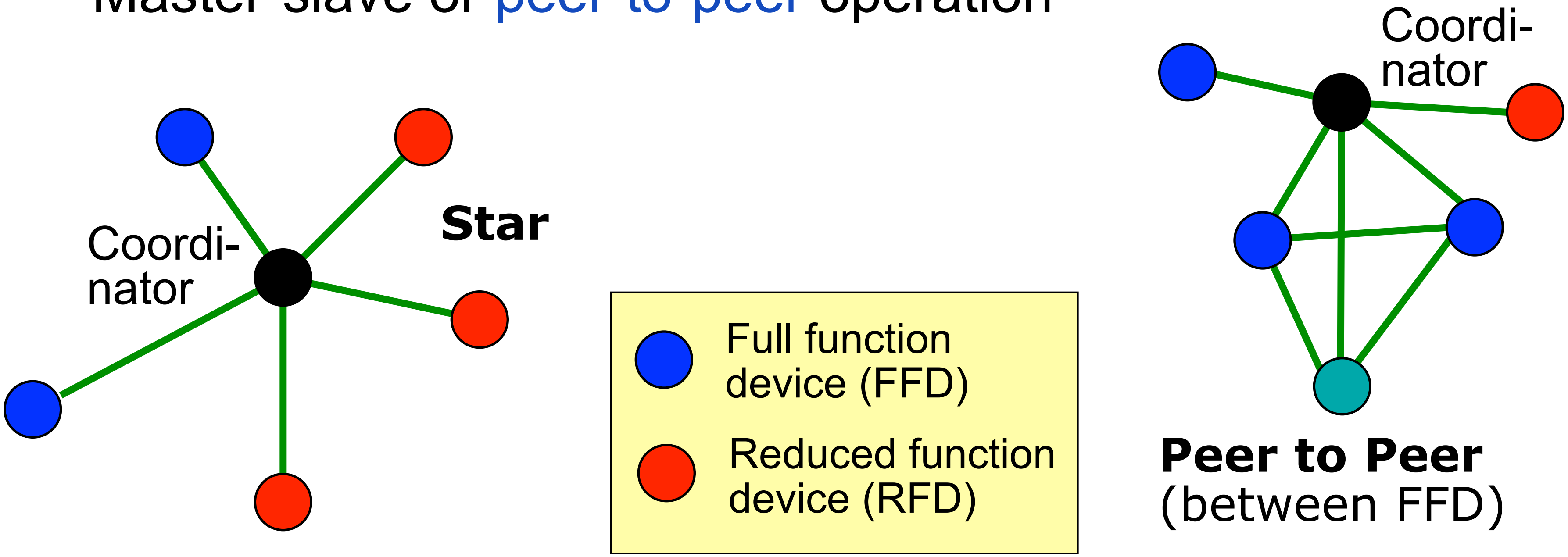
- Protocol specifications for low-power wireless communication
  - Published by ZigBee Alliance
  - ZigBee specification: http://www.zigbee.org
- Builds on IEEE 802.15.4 for wireless personal area networks (WPANs)
  - Wireless control and monitoring applications
- Simpler and cheaper than Bluetooth
- Requirements
  - Low data rate
  - Long battery life
  - Security

ZigBee Application Areas

- Home entertainment and control
  - Smart lighting, temperature control, safety and security, movies and music
- Home awareness
  - Water sensors, power sensors, smoke and fire detectors, smart appliances, access sensors
- Mobile services and telecommunication
  - M-payment, m-monitoring, m-security and access control, m-healthcare and tele-assist
- Commercial buildings
  - Energy monitoring, lighting, access control
- Industrial plants, hospital care
  - Process control, asset management, environmental management, energy management, industrial device control

ZigBee Characteristics

- 2 kbit/s up to 250 kbit/s max
- Low complexity, cost and power consumption
- Multi-month / multi-year battery life
- Support of latency-critical devices (e.g., joysticks)
- Master-slave or peer-to-peer operation



ZigBee Radio Layer

- ZigBee physical (PHY) and medium access control (MAC) layers conform to IEEE 802.15.4 “Wireless Personal Area Network” (WPAN)
  - Unlicensed ISM bands (2.4 GHz, 915 MHz, 868 MHz ISM bands)
  - Direct-sequence spread spectrum coding
- Over-the-air data rate
  - 250 kbit/s per channel in 2.4 GHz band
  - 40 kbit/s per channel in 915 MHz band
  - 20 kbit/s in 868 MHz band
- Transmission range: 10 to 75 m
- Maximum output power: 1 mW
- Channel access: “carrier sense, multiple access / collision avoidance” (CSMA/CA)

WIRELESS LAN

Wireless Local Area Network (WLAN)

- Ethernet cable replacement
  - Infrastructure-based WLANs require access point
- Small area installations
  - Offices, homes, coffee shops
- City-wide installations (metropolitan area networks)
  - E.g. free WLAN service in Mountain View, California, by Google

- Stations: Devices with a Wireless Network Interface Card
  - Access points: base stations for the wireless network
  - Wireless clients: mobile or fixed user devices

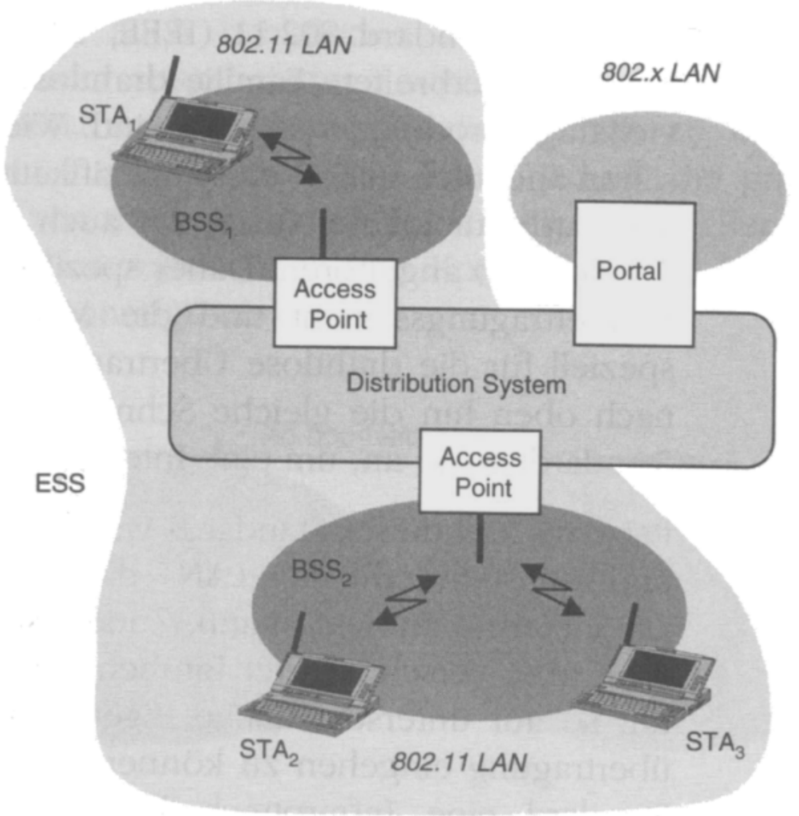
Wi-Fi

- Trade name for IEEE 802.11 WLAN technologies
  - Wi-Fi Alliance: http://wi-fi.org
- IEEE 802.11: set of standards for WLAN communication
  - 802.11a, 802.11b, 802.11g, 802.11n, etc.
  - 2.4 GHz and 5 GHz bands

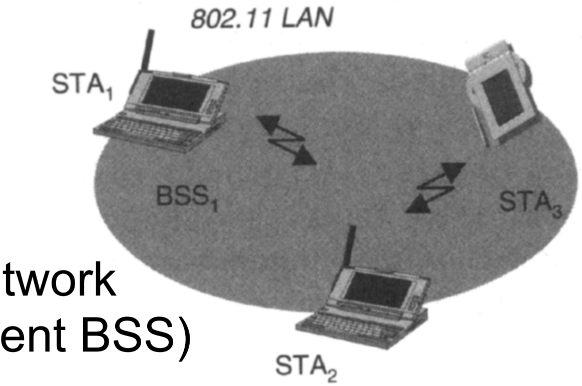
802.11x	Published	Frequency band	Data rate	Range (indoor)	Range (outdoor)
		(GHz)	(Mbit/s)	(m)	(m)
–	1997	2.4	2	~20	~100
a	1999	5	54	~35	~120
b	1999	2.4	11	~38	~140
g	2003	2.4	54	~38	~140

WLAN Architecture

- Basic service set (BSS)
  - Set of communicating stations
  - Infrastructure BSS
    - Identified by MAC address of access point
  - Independent BSS (IBSS)
    - Ad-hoc network (no access points)
- Extended service set (ESS)
  - Set of connected BSSes
  - Identified by SSID (Service Set Identifier, character string)
  - Distribution system (DS) connects access points in an ESS



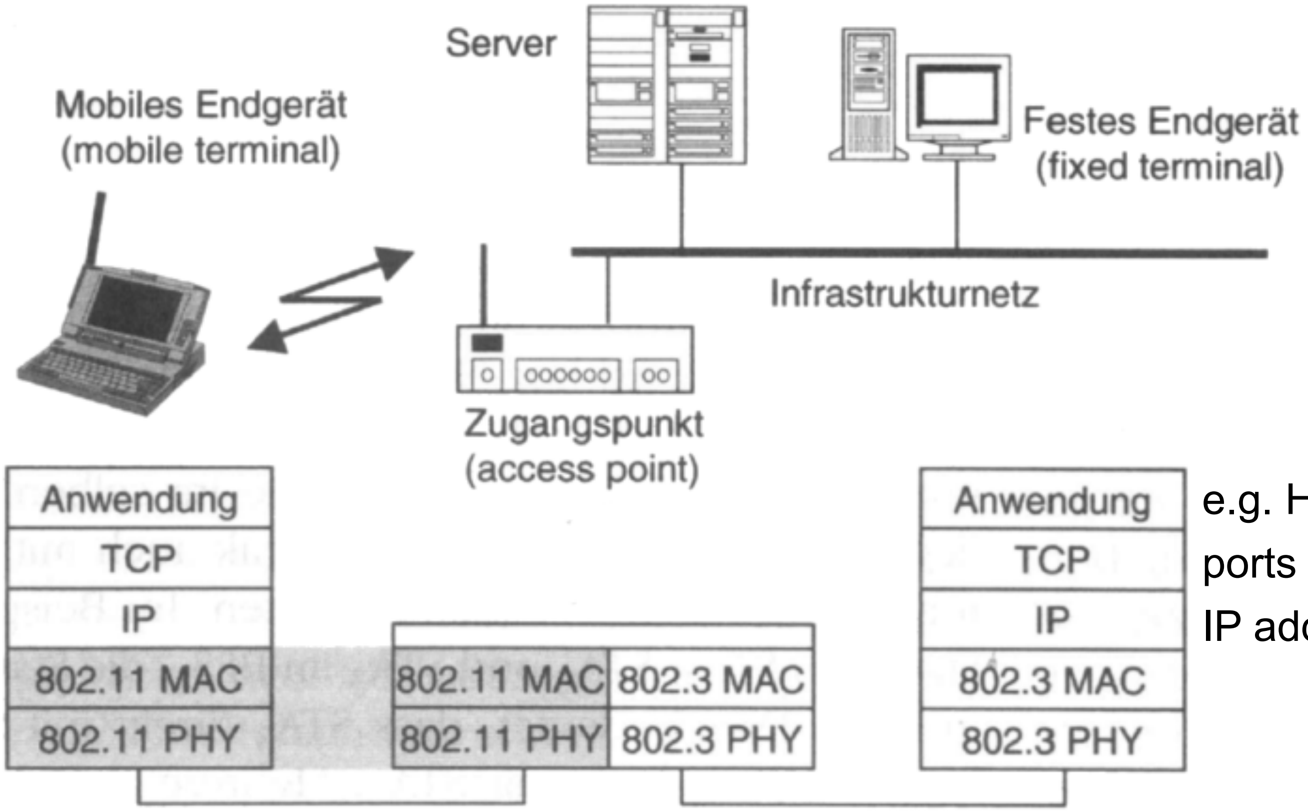
Infrastructure-based WLAN



Ad-hoc network (Independent BSS)

Protocol Architecture

- WLAN (IEEE 802.11) fits seamlessly into LAN (IEEE 802.3)
  - WLAN connected to LAN via a bridge
  - Wireless access transparent to applications



e.g. HTTP
ports
IP addresses

e.g. HTTP
ports
IP addresses

WLAN Security

- Unobservable interception of wireless packets
  - No wire tapping necessary
  - Long range with good antennas
- Shared-key encryption
- Wired Equivalent Privacy (WEP)
  - Original encryption standard for WLAN
  - Weak, can easily be broken
  - AP uses the same key as all the clients
- Wi-Fi Protected Access (WPA, WPA2)
  - Developed by the Wi-Fi Alliance to replace WEP
  - Higher security (esp. WPA2)

Wireless Technologies Comparison

	WLAN (802.11)	Bluetooth (802.15.1)	ZigBee (802.15.4)
Range	~100 m	~1-100 m	~10 m
Data throughput	~2-54 Mbit/s	~1 Mbit/s	~250 kbit/s
Power consumption	Medium	Low	Ultra low
Size	Larger	Smaller	Smallest
Cost/complexity	Medium	Low	Very low

Bandwidth and range vs. consumption and cost

Source: Andrew D. Parker, http://lecs.cs.ucla.edu/~adparker/EE202A/hw2

GSM / GPRS / UMTS

First Generation – Analog

- Characteristics
  - Analog systems
  - Primarily designed for voice communication
  - Many different standards (1980s: 7 incompatible standards in Europe! – national regulations!)
  - Little protection against eavesdropping
- Systems
  - 1958: A-Netz (D)
  - 1972: B-Netz (D)
  - 1981: NMT (Nordic Mobile Telephone, Scandinavia)
  - 1983: AMPS (Advanced Mobile Phones Service, USA)
  - 1985: C-Netz (D)

First Generation – Analog

- 1983: AMPS (Advanced Mobile Phones Service)
  - Separate frequencies (“channels”) for each conversation
    - Frequency division multiple access (FDMA)
  - Considerable bandwidth
  - No protection from eavesdropping
  - 1998: USA still 80% AMPS

Analog Mobile Telephony in Germany

- 1958-1977: A-Netz (D)
  - "Öffentlicher beweglicher Landfunk (ÖbL)"
  - First mobile phone service in Germany
  - 10'500 users maximum
  - Hand-connected calls
- 1972-1994 B-Netz (D)
  - Self-dialled connections in both directions
  - 27'000 users maximum (reached in 1986)
  - Calling a mobile user required knowledge of location (users had to dial location prefix)
- 1985-2000: C-Netz (D)
  - 800'000 users maximum
  - Handover between cells
  - Dedicated number independent of location

Second Generation – Digital

- Characteristics
  - Digital systems (enhanced voice quality, SMS)
  - Connection-oriented
  - Compatible to ISDN telephony
  - Uniform standard in Europe (GSM)
- Systems
  - 1982: Global System for Mobile Communications, GSM (Europe)
  - 1991: first GSM network operational in Finland
  - 1993: PDC (Personal Digital Cellular, Japan)
  - 1995: IS-95 CDMA (cdmaOne, USA)
  - 1990: IS-54 and IS-136 TDMA (Digital AMPS, USA)

2.5 Generation – Digital

- Characteristics
  - Improvement and extension of GSM
  - HSCSD: multiple GSM connections in parallel
  - GPRS: packed service based over GSM
  - EDGE: increased bandwidth with better encoding

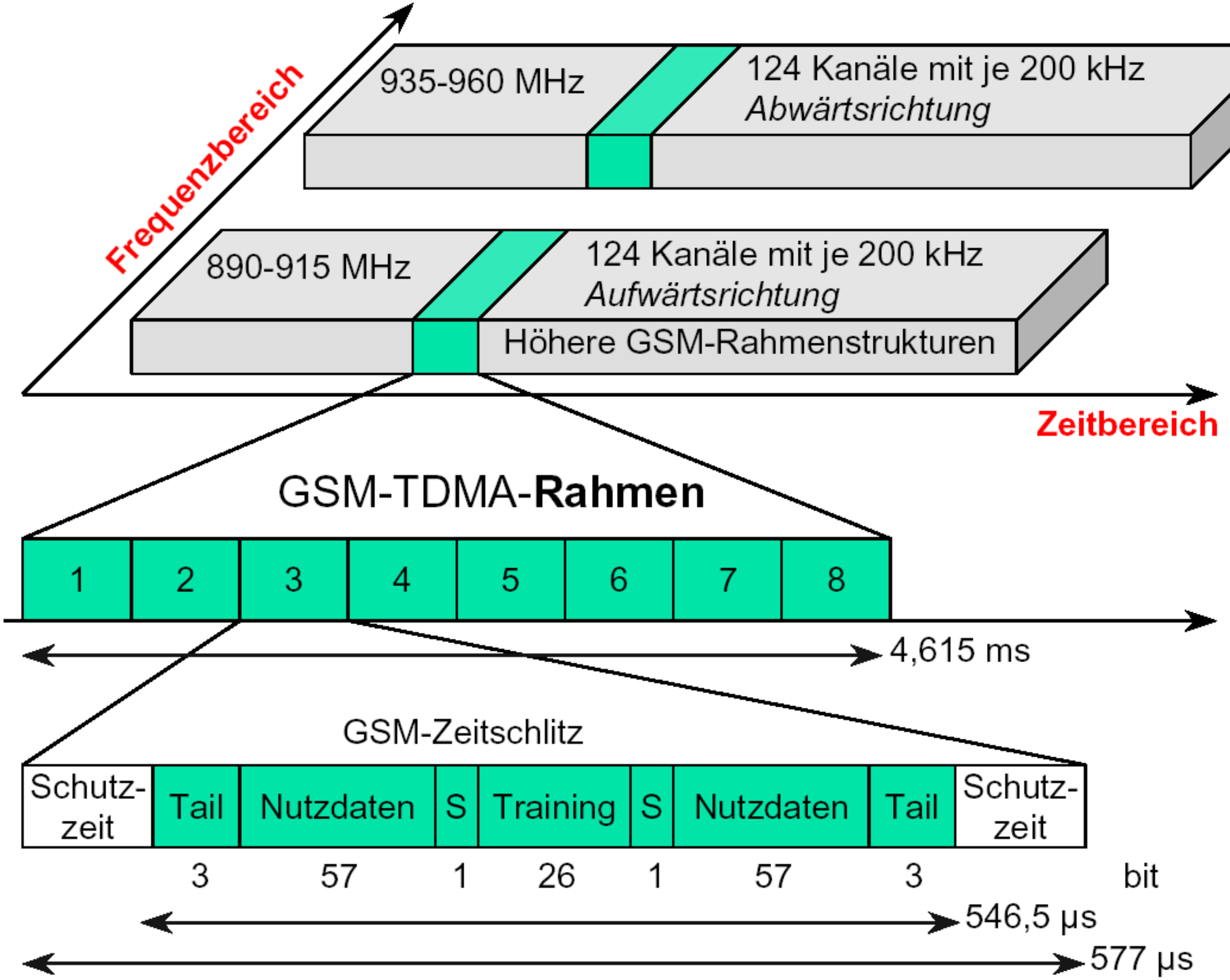
Third Generation – UMTS

- Characteristics
  - Digital system
  - Both connection and packet oriented
  - Global (worldwide) standard
  - Multimedia data
- Systems
  - 1992: frequency allocation fixed
  - 2002: UMTS networks in operation (universal mobile telecommunications system, also called 3GSM)

GSM Characteristics

- Mobile communication via a connection-oriented wireless channel
- Supports voice and data services (9.6 kbits/s)
- Separate data and control channels
  - SMS on control channel
- Phone number independent of location
  - GSM supports handover and location management
- Security via subscriber identity module (SIM)
  - Voice channel encrypted
- High system complexity (draft standard 5000 pages!)

GSM – FDMA / TDMA



Frequenzbereich

935-960 MHz

124 Kanäle mit je 200 kHz
Abwärtsrichtung

890-915 MHz

124 Kanäle mit je 200 kHz
Aufwärtsrichtung

Höhere GSM-Rahmenstrukturen

Zeitbereich

GSM-TDMA-**Rahmen**

1 2 3 4 5 6 7 8

4,615 ms

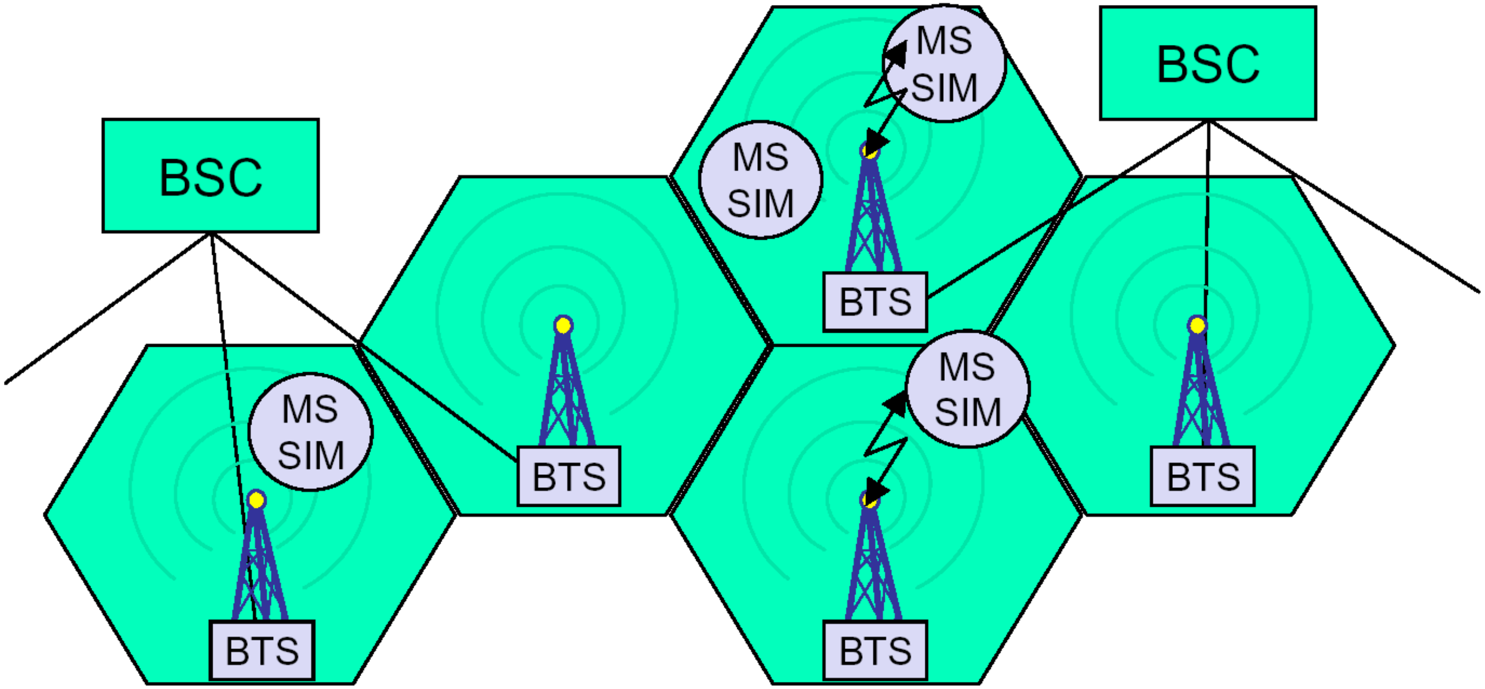
GSM-Zeitschlitz

Schutz-zeit	Tail	Nutzdaten	S	Training	S	Nutzdaten	Tail	Schutz-zeit	
	3	57	1	26	1	57	3		bit

546,5 µs

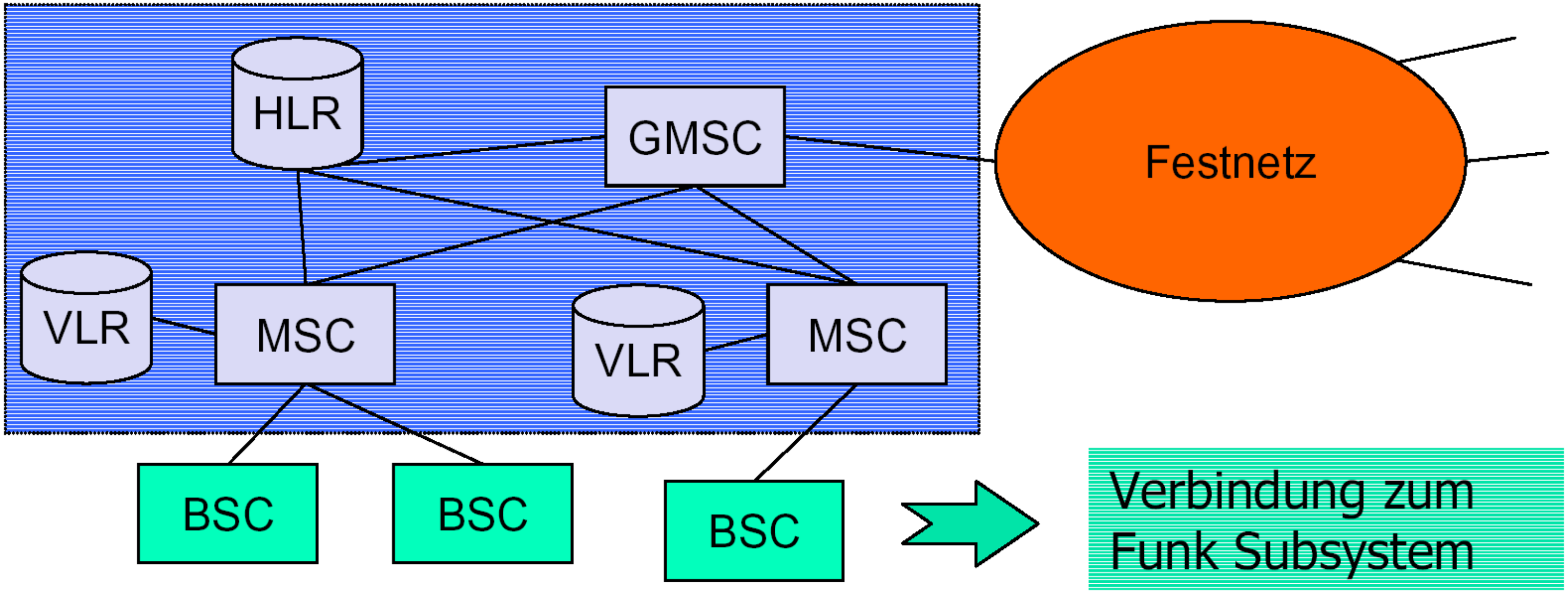
577 µs

GSM – Radio Subsystem



• BSC:	Base Station Controller
• BTS:	Base Transceiver Station
• MS:	Mobile Station
• SIM:	Subscriber Identity Module

GSM – Network and Operation

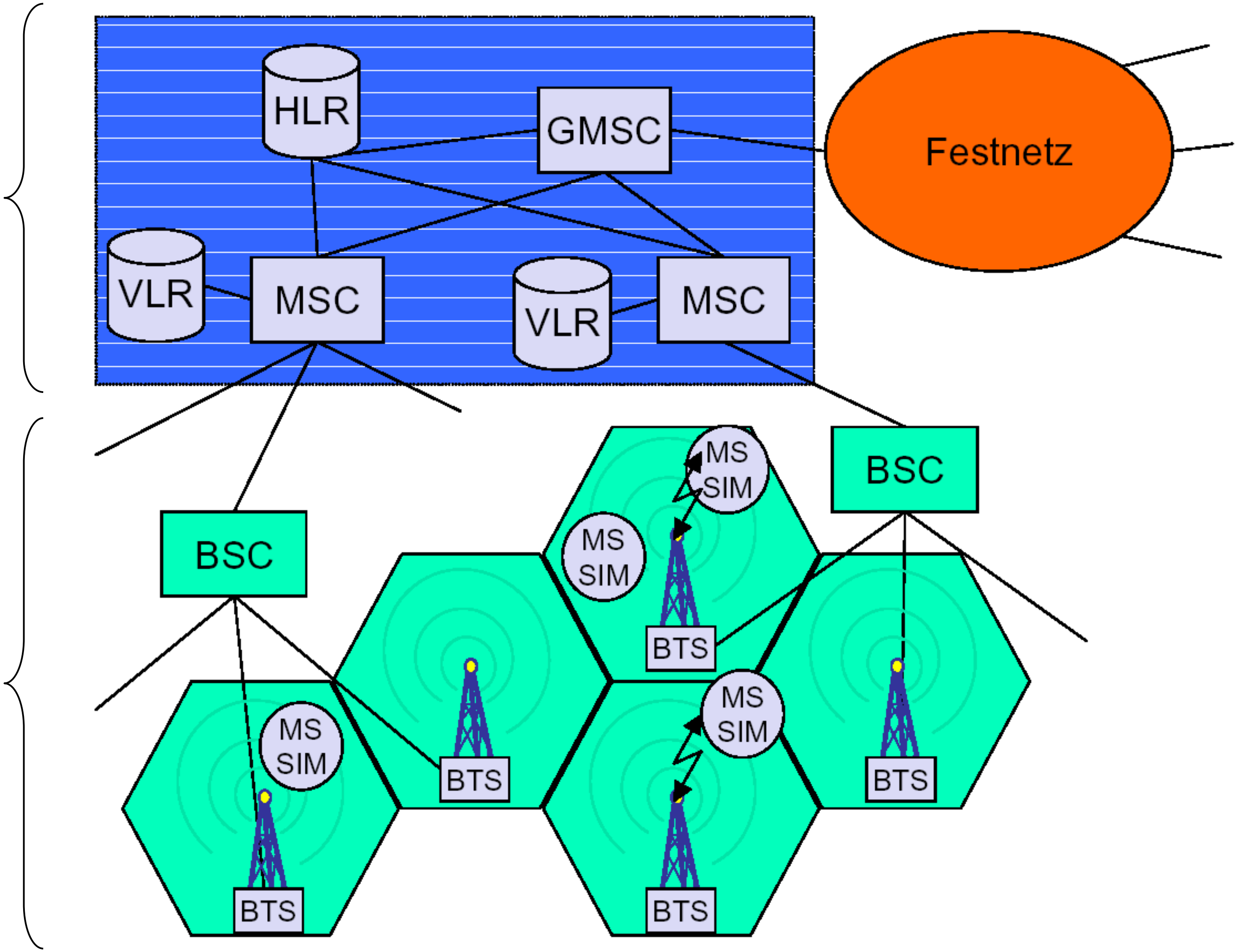


MSC	Mobile Services Switching Center
GMSC	Gateway MSC
VLR	Visitor Location Register
HMR	Home Location Register

GSM Architecture

Network subsystem / operation and maintenance subsystem

Radio subsystem / mobile station



Cell-Based Systems

- Locality of cells has many advantages
  - More users (reuse of frequencies)
  - SDMA (space division multiple access)
  - Less interference
  - Less send/receive power

GSM – Identifiers

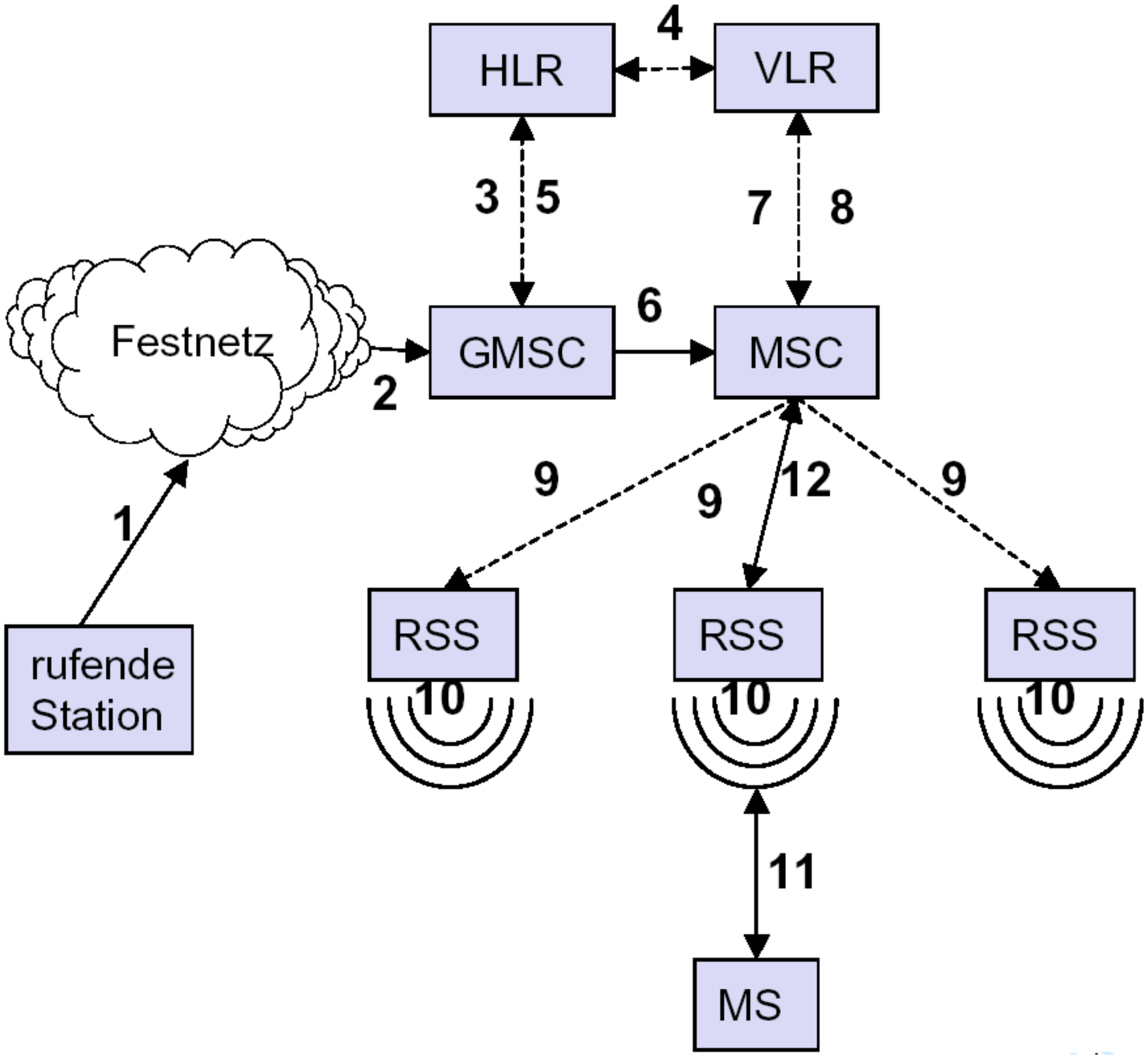
- MSISDN (Mobile Subscriber ISDN)
  - Hierarchical phone number (CC, NDC, SN)
  - Bound to SIM, not to MS
  - Identifies HLR
- IMSI (International Mobile Subscriber Identity)
  - Internal unique identity of the user (MCC, MNC, MSIN)
- TMSI (Temporary Mobile Subscriber Identity)
  - Real identity not revealed (system uses TMSI instead of IMSI)
  - Periodic change of TMSI
- MSRN (Mobile Station Roaming Number)
  - Same structure as MSISDN
  - Stored in HLR
  - Identifies current MSC / VLR

GSM – VLR / HLR

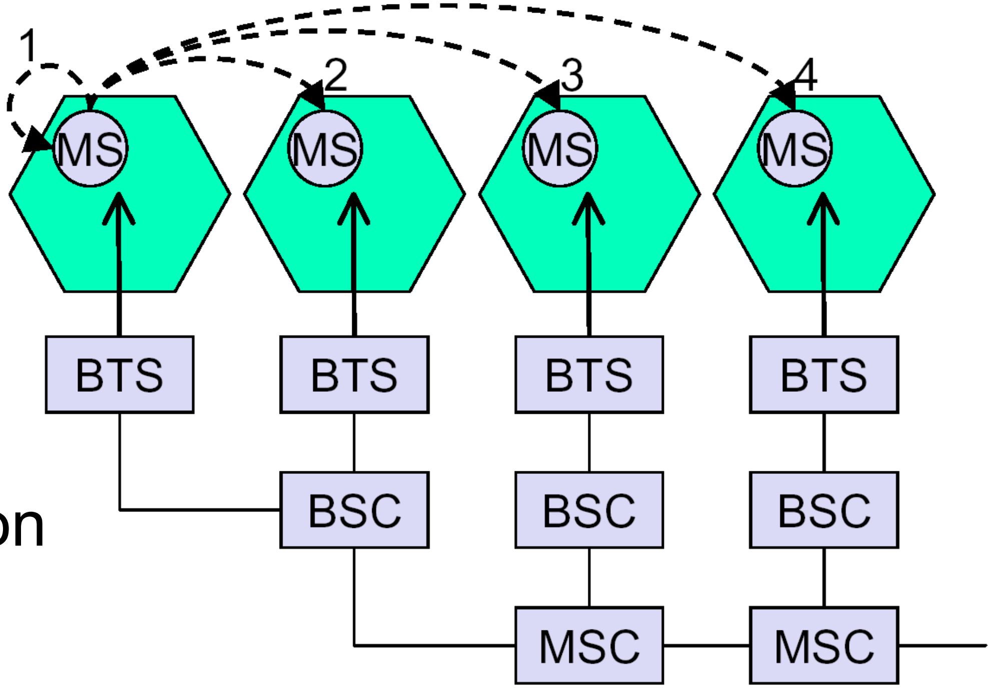
- Home Location Register
  - One per MS (mobile station)
  - Most important database in GSM
  - Identification via MSISDN
  - Stores user data
    - MSISDN
    - Enabled services
    - Authentication data
    - Current location (LA = location area)
- Visitor Location Register
  - One per MSC (mobile services switching center)
  - Data of all MS in area of MSC
  - Frequent update caused by appearing MS
  - Identification by MSRN

GSM – Call to MS

1 Call to MSISDN
2 Forwarding to GMSC
3 Call setup message to HLR
4 Query MSRN from VLR
5 Current MSRN
6 Call forward to current MSC
7,8 Request IMSI, TMSI
9,10 Call MS (paging)
11,12 MS responds



GSM – Handover



- Handover: transparently dispatch active connection to another access point
- Four kinds of handover
- Reasons for handover
  - Movement of MS (change of cell)
  - Load management
  - Noise on current channel

GSM Summary

- GSM is an very large standard
  - 5000 pages in original specification
  - Defines very many functional units and services
- Optimized for voice services
- Less suited for data services (http, ftp, …)
  - Low bandwidth
  - Connection-oriented (pay while connected)
  - Same capacity for both uplink and downlink
- Example of a successful standard

GPRS

- Extension of GSM
  - Integrated in “GSM Release 97”
- Packet-oriented data service
  - Use of time slots only if data available
- Better suited for data services, more flexible
  - Different bandwidth requirements up and down: different number of slots used
  - Pay for data volume, not for connection time
  - Different levels of quality of service (QoS)

GPRS – Data Rates

Technology	Download (kbit/s)	Upload (kbit/s)
CSD	9.6	9.6
HSCSD	28.8	14.4
HSCSD	43.2	14.4
GPRS	80.0	20.0
GPRS	60.0	40.0
EGPRS (EDGE)	236.8	59.2
EGPRS (EDGE)	177.6	118.4

UMTS

- Worldwide standard
- Various voice and data services (up to 2 Mbit/s)
- Compatible to Internet protocols
- Packet as well as connection-oriented
- Data rates
  - Up to 14 Mbits/s when stationary
  - At least 144 kbit/s even at high speeds
- Currently deployed systems
  - 384 kbit/s or 3.6 Mbit/s downlink, depending on handset

The End