Mobile Computing Models

Outline

- Mobile computing models and Application architectures
- Popular mobile computing models
  - Mobile client-server computing
  - Mobile Peer-to-Peer computing
  - Mobile Agents
- Mobile Internet Architectures
- Mobile devices

Mobile computing models

- **A *mobile computing model* specifies a *computing paradigm* for designing and organizing mobile applications.**
  - Divide the mobile computing system into a collection of *components*
  - Specify how to use these components and their *relationships* to design mobile computing applications
  - Examples:
    - *Client-Server model*: clients, servers, requests and results;
    - *Mobile P2P model*: peers, neighbors, sharing & cooperation;
    - *Mobile Agent model*: agents, hosts, communication & coordination;

Application architectures

- **Based on a mobile computing model, a mobile application architecture defines how the model can be realized / implemented**
  - Specify the division of functionalities among the components in the model
  - Describe the mechanisms for implementing each participating component

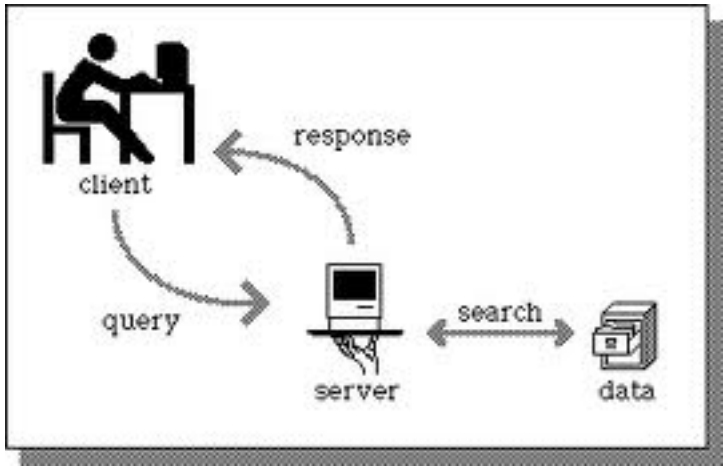
Model vs. Architecture

- **Model is defined from the software designer's viewpoint**
  - It is precise enough about the system organization and performance but not too explicit about mechanisms and implementation details
- **Application architecture is defined from the programmer's viewpoint**
  - It provides details about mechanisms and implementations

Example: C/S model

- **A network computing model in which each computer or process on the network is either a *client* or a *server***
  - Clients: asking for services, e.g., email
  - Servers: providing services
  - Communication networks: connecting client and server

Client-Server computing optimizes computing resources



Example: C/S application architectures

- ***Thin Client (Wireless Internet)*:**
  - Mostly for online access to Web content
  - Server side plays the main functional role
  - Limited user interface and capabilities on client side;
- ***Smart Client :***
  - Allows for working in 'occasionally connected' environment with offline access to important data and re-synchronization upon reconnection
  - Both server and client sides have good capabilities
  - Client side incorporates mobile database technology for persistent data storage

Mobile Internet Architectures

Mobile Internet

Thin Client

- Also called as Wireless internet applications
- Same architecture as wired Internet applications
- Business and enterprise data are stored on the servers
- Client side is a mobile device with internet browser called microbrowsers (due to their limited size and functionality). No other software is required on the client device

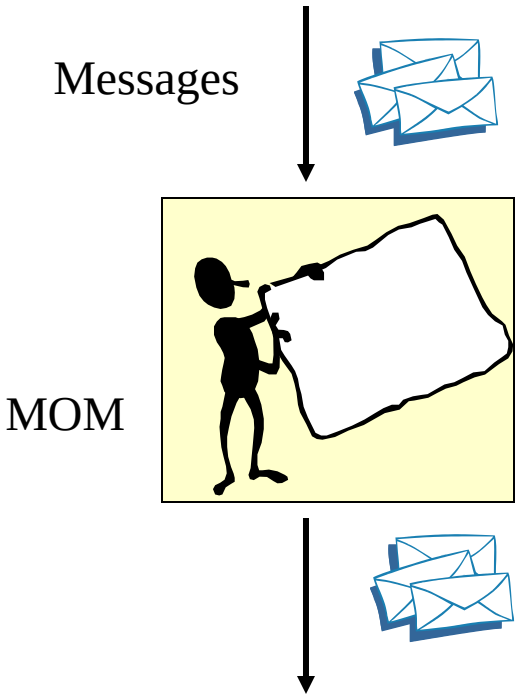
Smart Client

- Enables user to access data when disconnected from the network.
- Instead of a microbrowser, application software is developed on the client device, so it is mobile and can be used even when the connection is unavailable.
- By deploying applications to the device itself, we have capability to give client application some “smarts” or business logic.

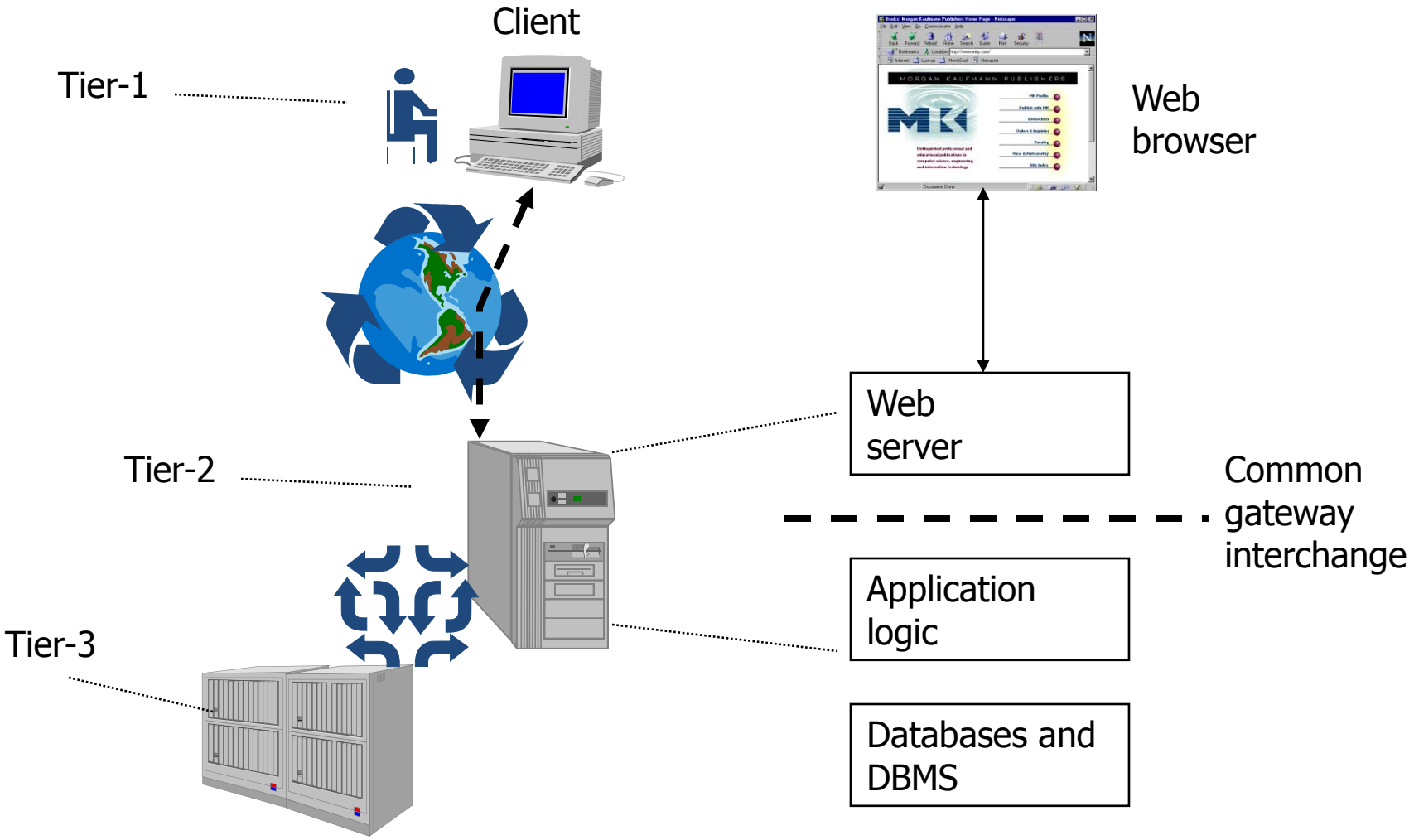
Mobile Internet

Mobile messaging

- Client and server applications are *loosely coupled* through the exchange of *self-describing* messages
  - Messaging can mean email, paging, SMS, voice, data, text etc
  - Store and forward delivery of data - sender and receiver are not necessary on-line at the same time
    - Messages are stored when the connection is not available and forwarded when it becomes available
- Can be used itself as a stand-alone application architecture, or in conjunction with wireless Internet or smart client as an enhancement.

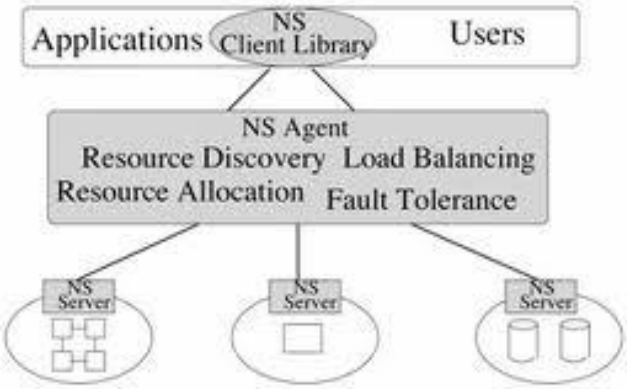


Three-tier client/server



Extended C/S models

- **Place some application logic in a middle component, called agent /proxy, between client and server.**
  - **Agents** can help perform some processing and/or mask out mobile computing limitations
  - More flexible with thin client and smart client: application logic can be changed readily



Extended C/S model

- **Agents can also be associated with client side or server side.**
  - Client-side agent functionalities
    - Perform some pre-fetching and compression operation.
    - Cache data for satisfying client's request during disconnection.
  - Server-side agent functionalities
    - Messaging and queuing.
    - Optimize transmission over wireless link
    - Offload some processing from client
    - Handle disconnection of client

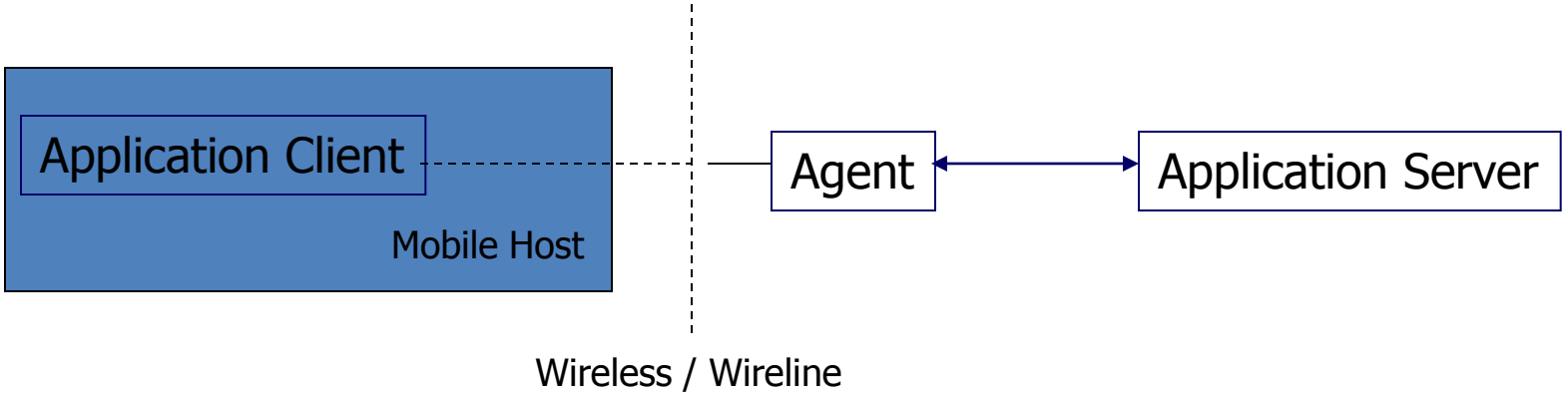
Extended C/S model

- **Agent types**
  - ***Shared client***: multiple clients can share the same agent
  - ***Client-specific***: each client has one agent
  - ***Service-specific***: each service of the client is associated with an agent (e.g., web browsing, database access)
  - Agent can be at a fixed location or can move with the client

Extended C/S architectures

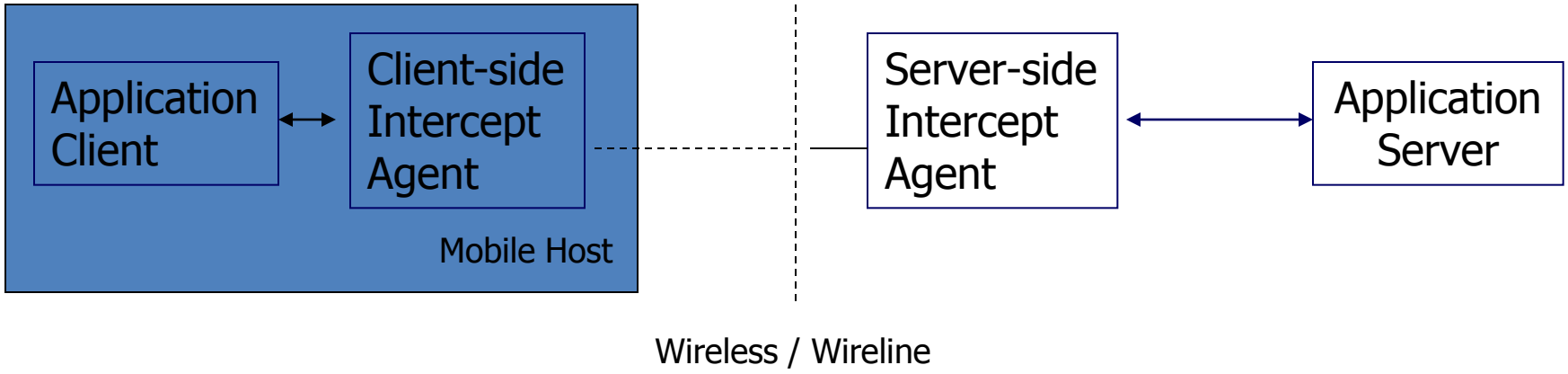
- Two common architectures based on the extended C/S model
  - **CAS** model (**C**lient/**A**gent/**S**erver)
  - **CIS** model (**C**lient/**I**ntercept/**S**erver)

Client/Agent/Server architecture



- **Agent proxy: a surrogate of the client on the wired network side**
  - Communications between client and server pass through agent
  - Continuously maintains the client's presence on the fixed network
  - To the server, the agent looks like the mobile client, so standard C/S interaction occurs between agent and server
  - Different protocol can be used for interactions between mobile client and agent

Client/Intercept/Server architecture



- Pair of agents - one stays on client side and the other on fixed network.
- Agents are transparent to both client and server – no limit on functionality or interoperability of the client.
- Two agents cooperate to facilitate highly effective data reduction and protocol optimization, and application-specific optimization

CAS vs. CIS

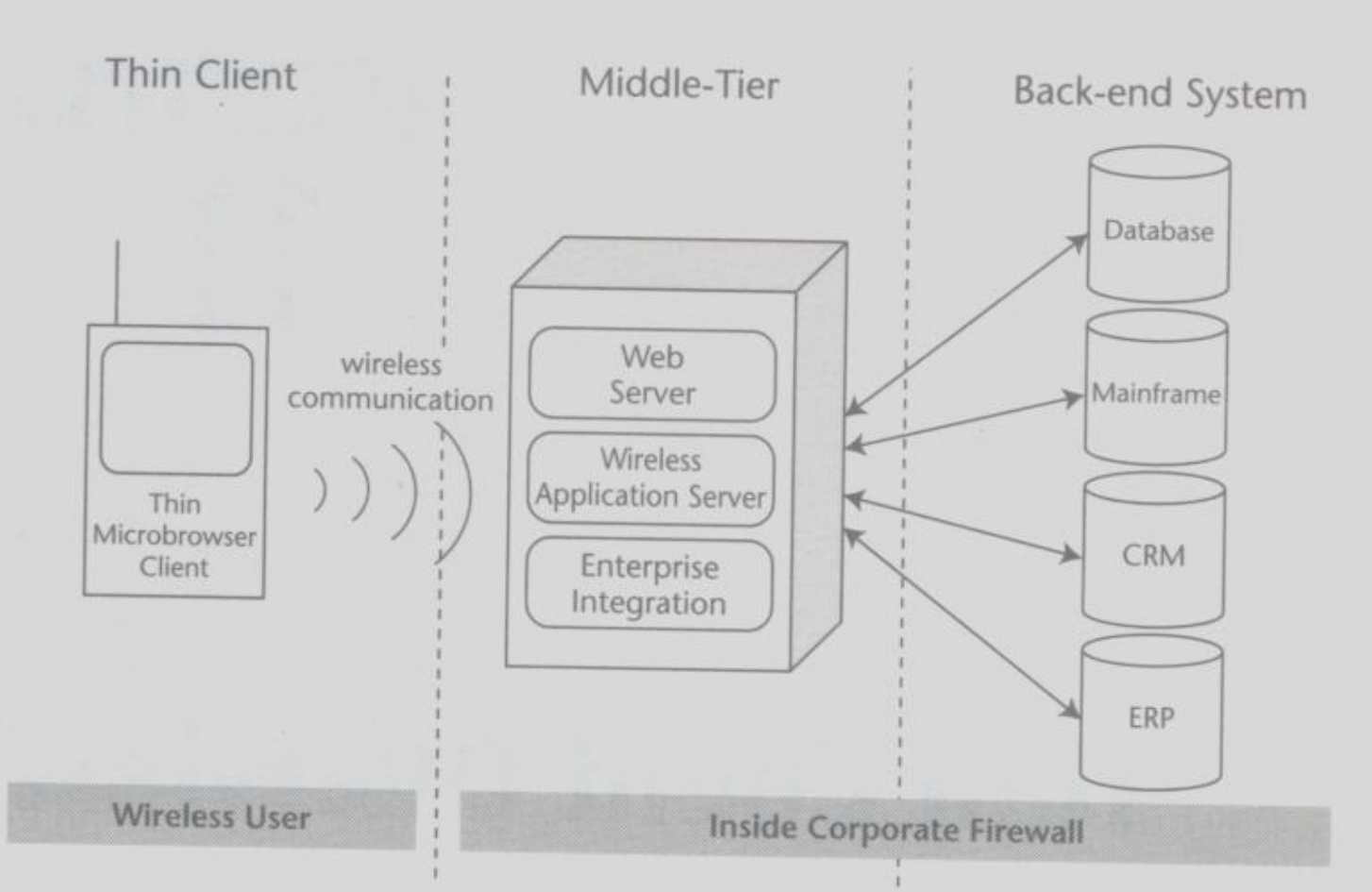
Advantages of CAS

- Complex client requests can be managed by agent with only the final result transmitted to the client
- Server can also offload some activities to agent (e.g., compression)
- Agent can cache some results to improve performance

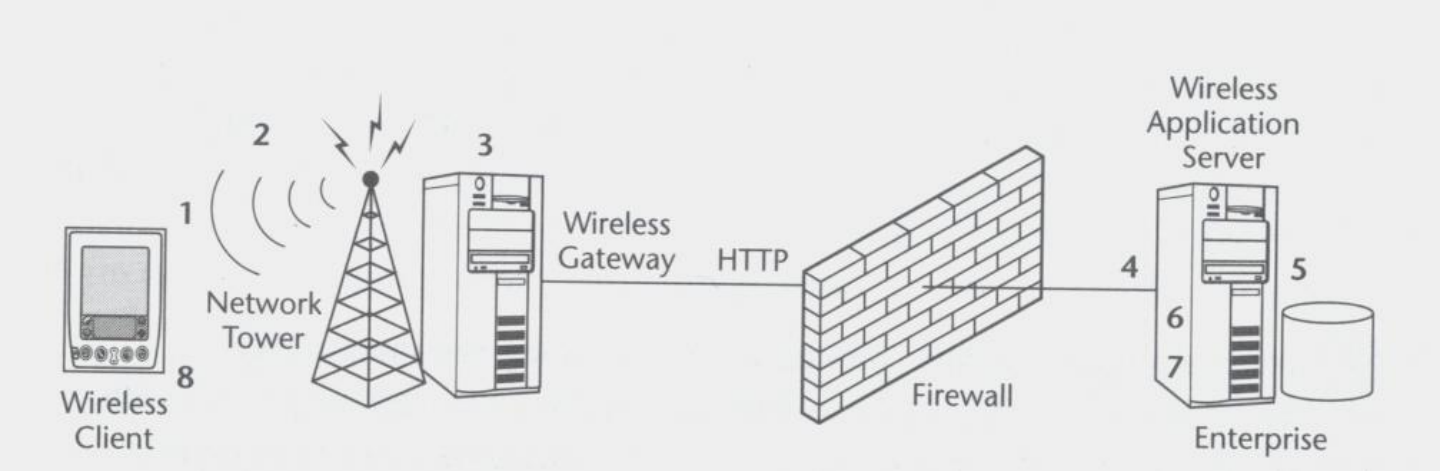
Disadvantages of CAS

- Changes are needed in client code to communicate with agent
- In normal operation, system overhead will increase.

Thin Client architecture



Processing a wireless request



1. **Establish a wireless session**
2. **Submit a request**
3. **Translate a request**
4. **Receive a request**
5. **Identify the wireless client**
6. **Process the request**
7. **Transform content for device**
8. **Return the content**

When the wireless network is not IP based, the base station contains a ***gateway*** which translates the request and send it as HTTP to the web server.

Wireless gateway

- **Gateway serves as the link between wireless and wired world**
  - Usually hosted by wireless network provider (e.g., in a BS)
- **Provided functions**
  - Protocol gateway
    - e.g., wireless protocol ↔ IP, WAP ↔ HTTP, HTML ↔ WML etc.
  - Optimized communication streams
  - WAP push messaging
  - Enhanced security
- ***WAP gateway* is a commonly used wireless gateway**

Servers

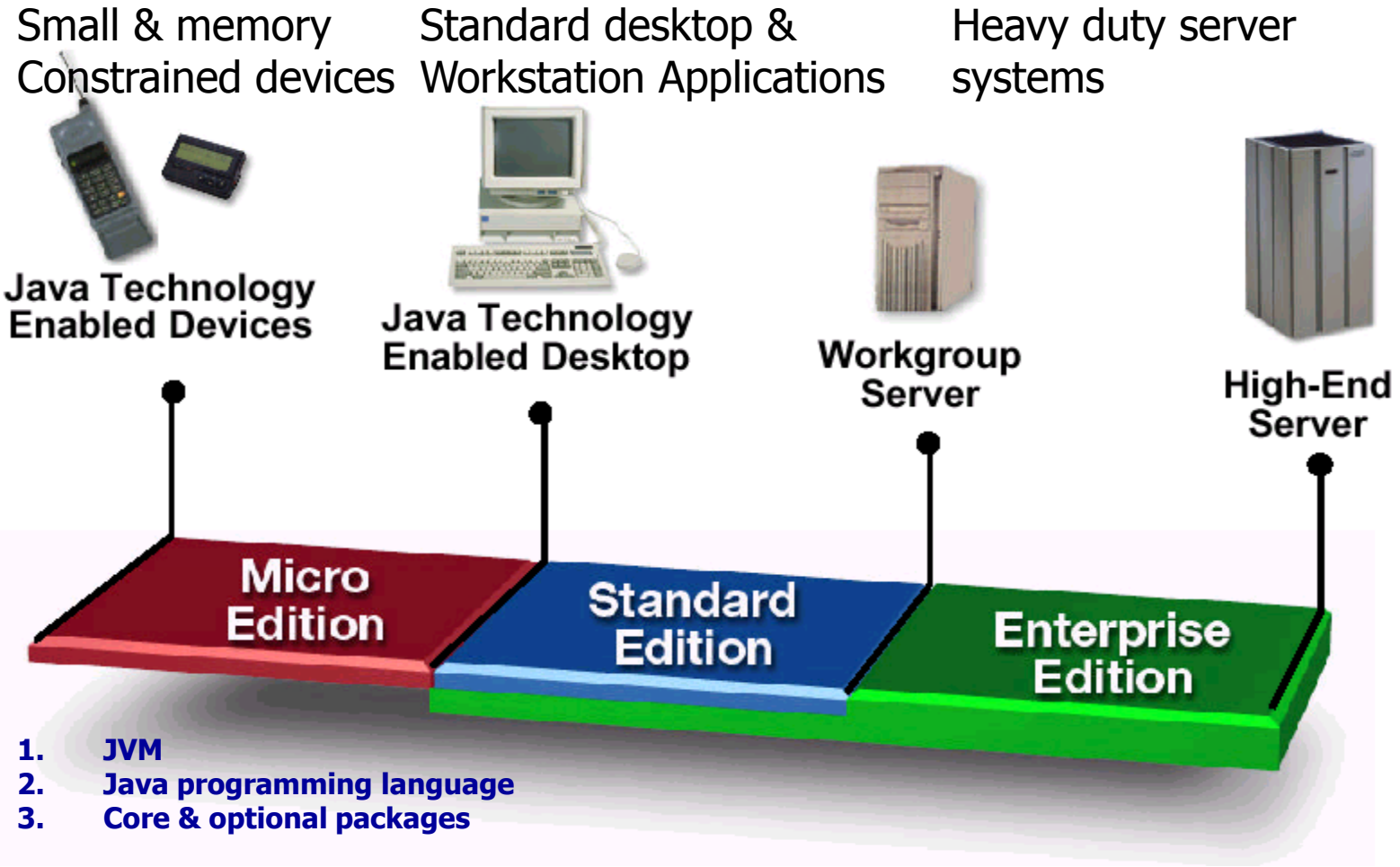
- ***Web server***
  - Listens to the incoming HTTP request and sends response back to the client.
  - Format data to be returned to client.
- ***Wireless application server***
  - Usually located within the enterprise
  - Provide core infrastructure to build wireless Internet applications, and is the engine driving the architecture
  - Including features like
    - *device & browser identification*,
    - *content transformation*,
    - *session management*,
    - *enterprise data management*.

Two leading specifications

- **J2EE**
- **Microsoft .NET Framework**

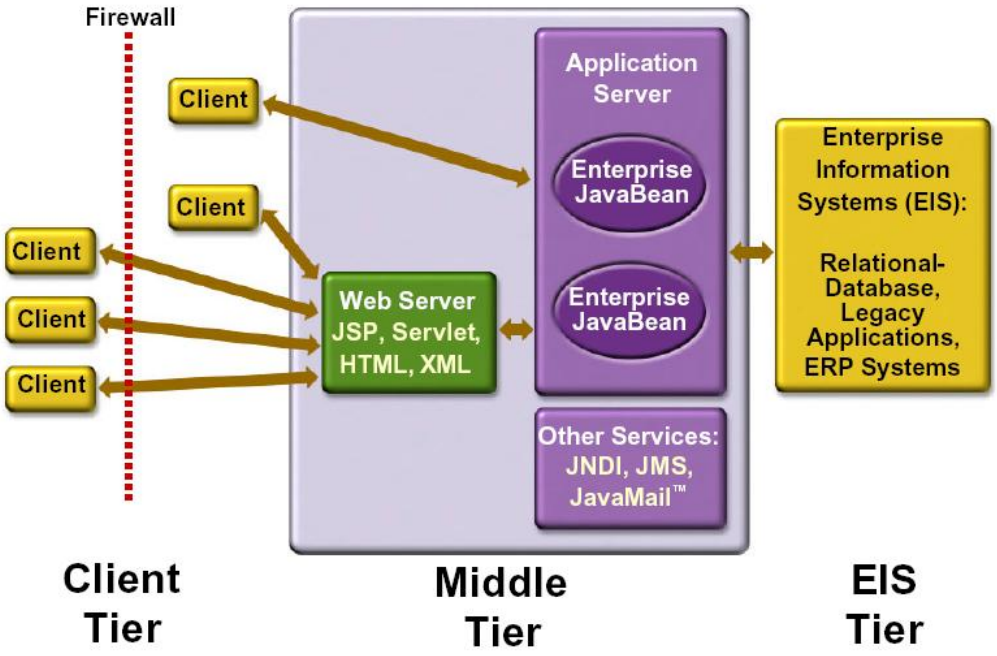
J2EE

- **The Java 2 platform includes** J2SE**,** J2EE**, and** J2ME**.**



J2EE

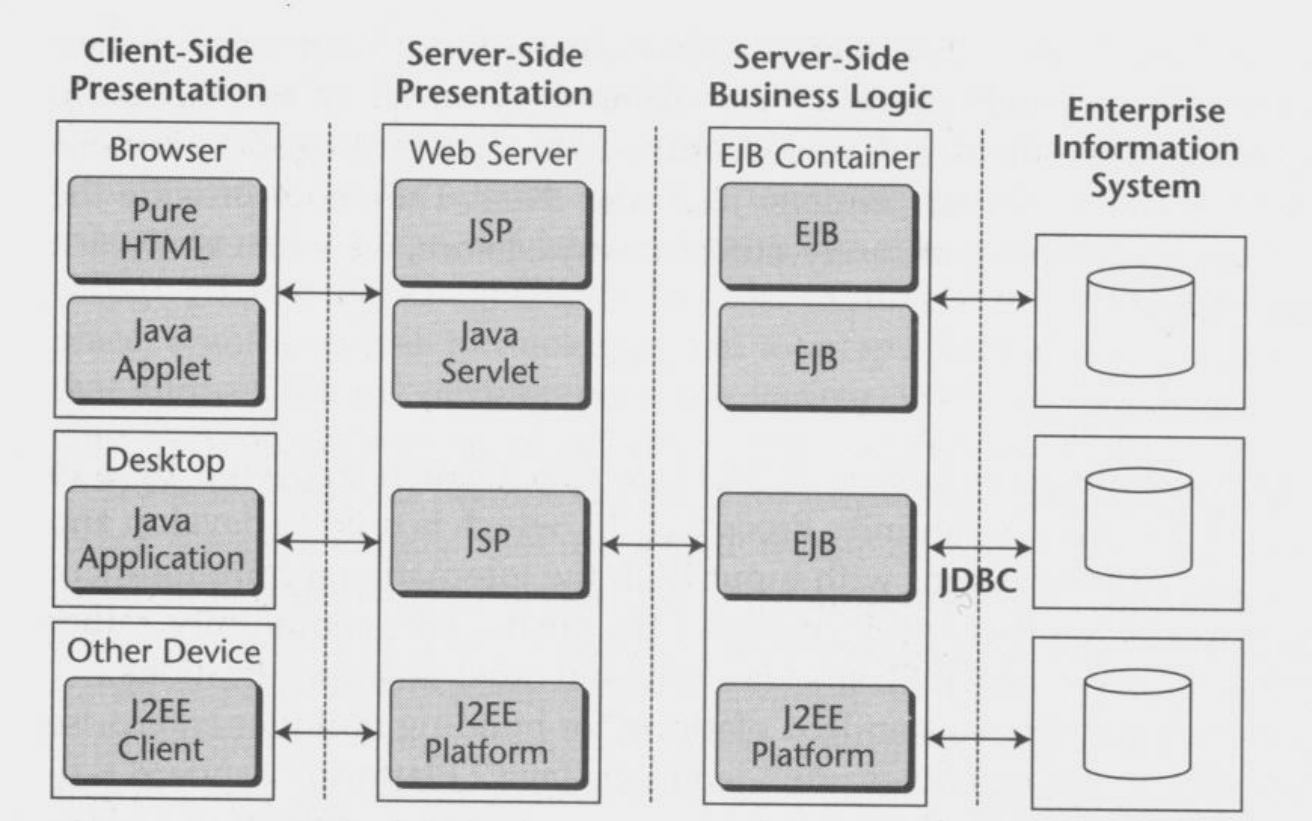
- **Defines a standard for building multi-tier enterprise applications.**
- **Three-tiered architecture:**
  - extend the standard two-tiered client and server model by placing a multithreaded application server between the client application and back-end storage



J2EE

- Four main technologies:

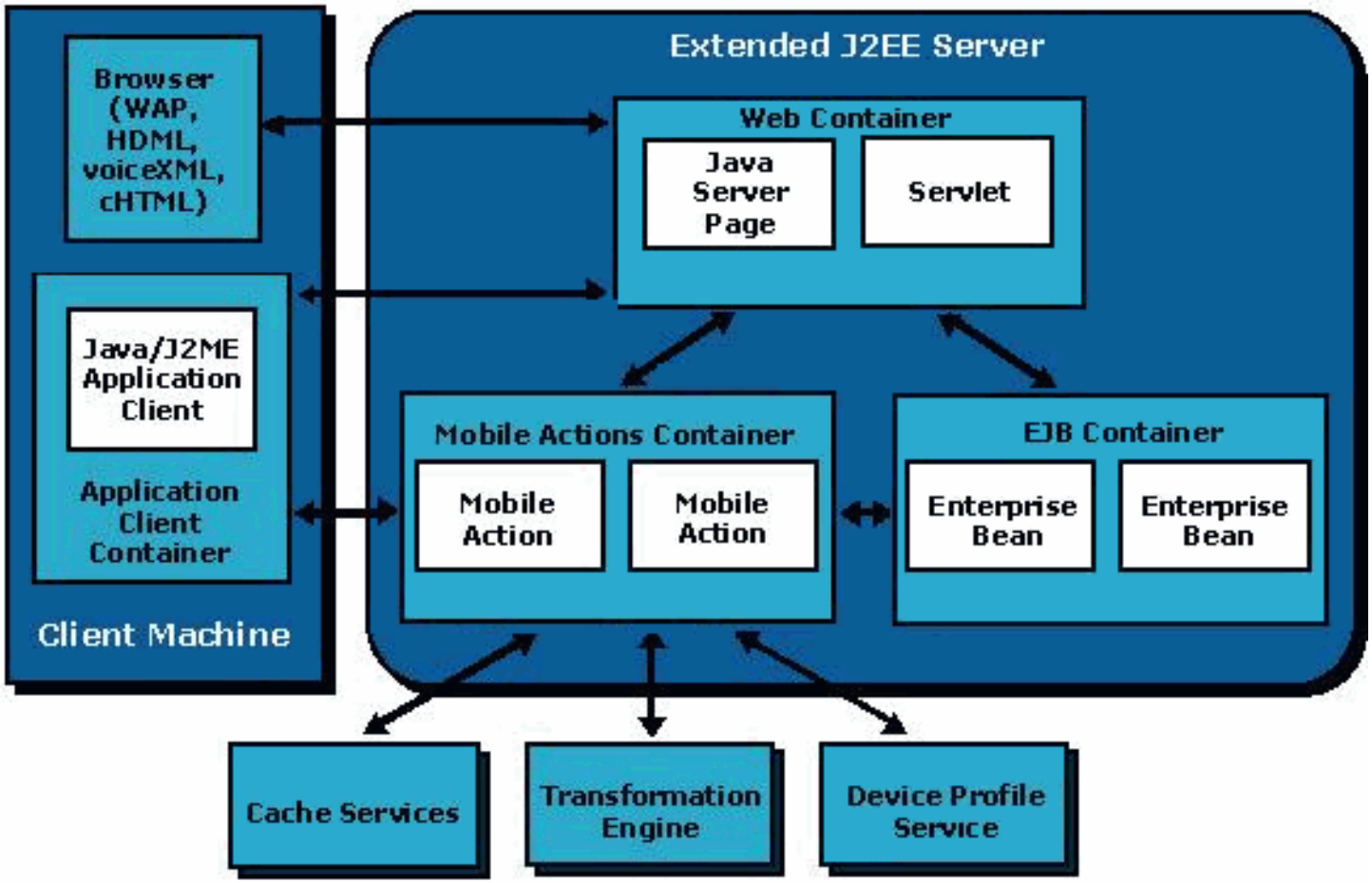
  Applets, Java servlets/JSP, EJB, JDBC.



J2EE Components

- **Application clients and applets**
  - **components that run on the client**
- **Web server (Servlets, Java Server Pages (JSP))**
  - **Web components that run on the server**
  - Java classes extending network server functionality
  - Coordinating the interaction between the user and EJB and produce forms and information for returning results to the user
  - Dynamic generation of contents
- **Enterprise Java Bean (EJB)**
  - **Business components that run on the server**
  - Packaging business logics and apply them to backend EIS
- **J2EE Connectors**
  - **Connect and access backend EIS**
  - Relational databases, JDBC, SQL

J2EE extensions for mobile application development



Wireless Internet Standards

- **WAP**:
  - Standard wireless Internet application protocol
- **i-mode from NTT DoCoMo (Japan)**:
  - Proprietary product based on wireless internet.
  - Only available in Japan
  - Various services available :
    News, Weather, Banking, Ticket booking, On-line ordering, Game, Wallpaper, Music....

WAP overview

- **A standard for the delivery and presentation of Internet content to mobile devices.**
- **WAP forum was founded in 1997 by Ericsson, Motorola, Nokia and Openwave System.**
  - *http://www.wapforum.org* (now called Open Mobile Alliance)
  - Released the first version of WAP (**WAP 1.0**) in 1998.
  - Released the second version of WAP (**WAP 2.0**) in 2000.
- **Now more than 300 corporations have joined the forum, making WAP the de facto standard for wireless Internet applications.**
  - 90% of mobile phones will be WAP-enabled

WAP - Motivations

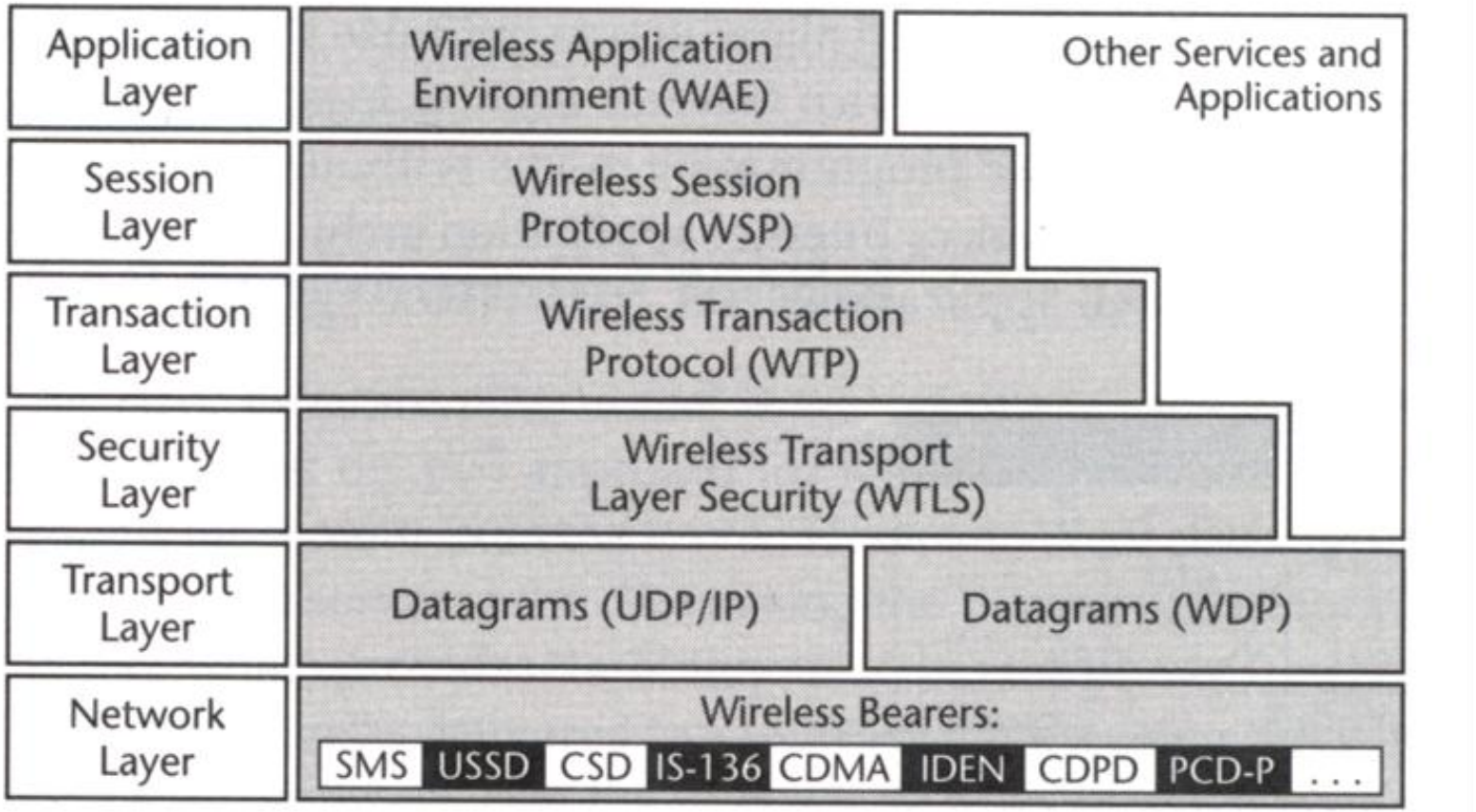
- **HTTP/HTML: not designed for mobile applications**
- **HTTP 1.0 characteristics**
  - designed for large bandwidth, low delay
  - stateless, client/server, request/response communication
  - connection oriented, one connection per request
  - TCP 3-way handshake, DNS lookup overheads
  - big protocol headers, uncompressed content transfer
  - primitive caching (often disabled, dynamic objects)
  - security problems (using SSL/TLS with proxies)
- **HTML characteristics**
  - designed for computers with "high" performance, color high-resolution display, mouse, hard disk
  - typically, web pages optimized for design, not for communication; ignore end-system characteristics

Need support for mobile, wireless Internet

WAP - Motivations

- **Enhanced browsers**
  - client-aware support for mobility
- **Proxies**
  - Client proxy: pre-fetching, caching, off-line use
  - Network proxy: adaptive content transformation for connections
  - Client and network proxy
- **Enhanced servers**
  - server-aware support for mobility
  - serve the content in multiple ways, depending on client capabilities
- **New protocols/languages**
  - WAP/WML

WAP components



WAP defines its own protocol stack, optimized for wireless communications

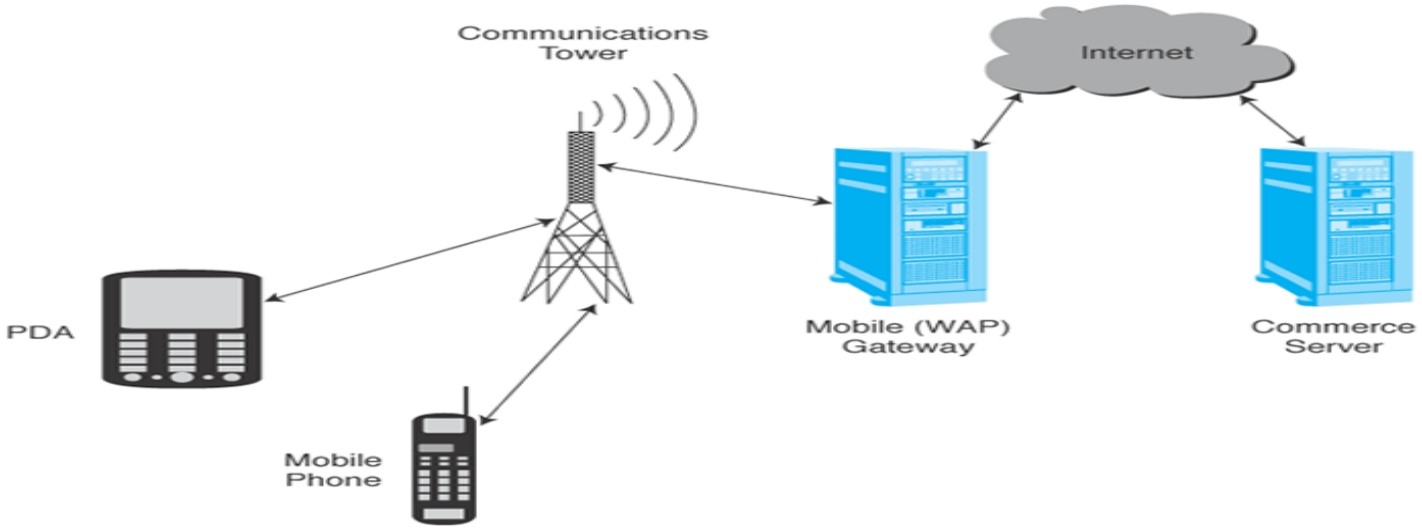
WAP protocol stack

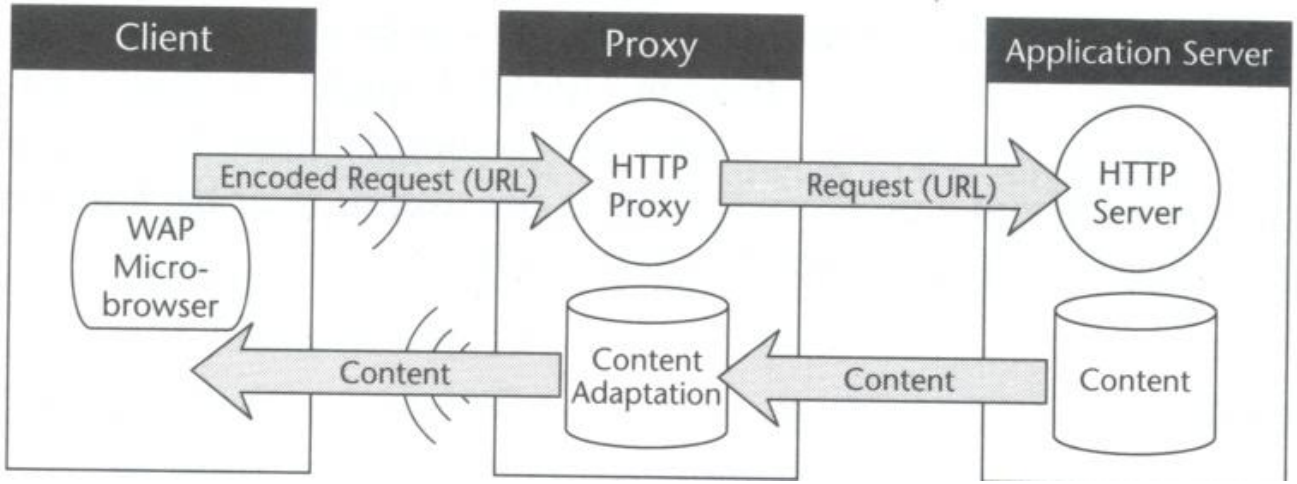
- **WAP1.x: proprietary protocols in pre-3G wireless networks.**
  - WAE (Wireless Application Environment) <-> Browser & HTML
  - WSP (Wireless Session Protocol) <-> HTTP
  - WTP (Wireless Transaction Protocol) <-> TCP/IP
  - WTLS (Wireless Transport Layer Security) <-> TLS/SSL
  - WDP (Wireless Datagram Protocol) <-> UDP
- **WAP2.x : 2.5G and 3G, using standard Internet protocols.**
  - WML2 based on XHTML
  - WP-HTTP (Wireless Profiled HTTP)
  - TLS (Wireless Profiled Transport Layer Security)
  - WP-TCP (Wireless Profiled TCP)
- **Other WAP2.x Services**
  - WAP Push
  - User Agent Profile
  - External Functionality Interface (EFI)
  - Wireless Telephony Application (WTA)
  - Persistent storage interface
  - Data synchronization
  - Multimedia Messaging Service (MMS)

WAP programming model

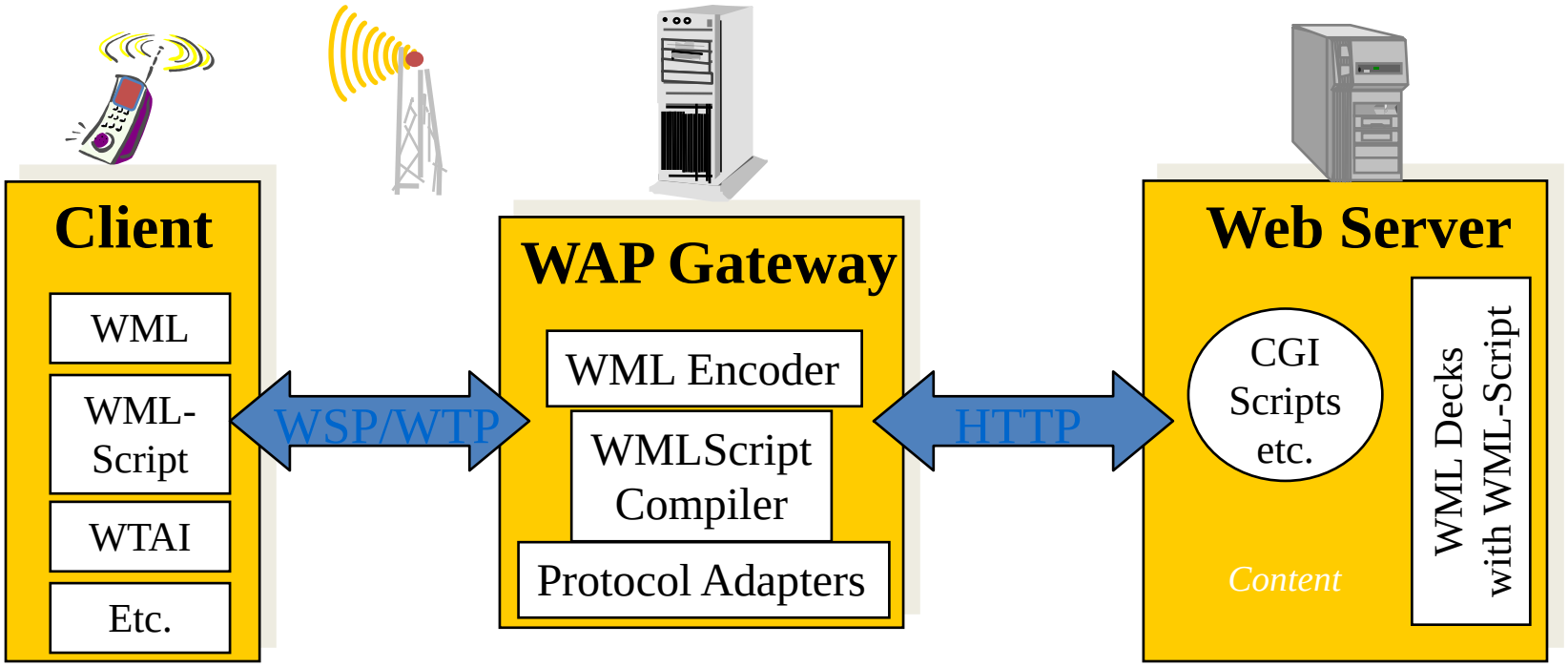
- **Similar to Internet programming model**
  - Uses both Pull and Push approaches to obtain the content
- **Two models**
  - Using a WAP gateway (also called WAP proxy)
  - Without a WAP gateway

Using a wireless gateway



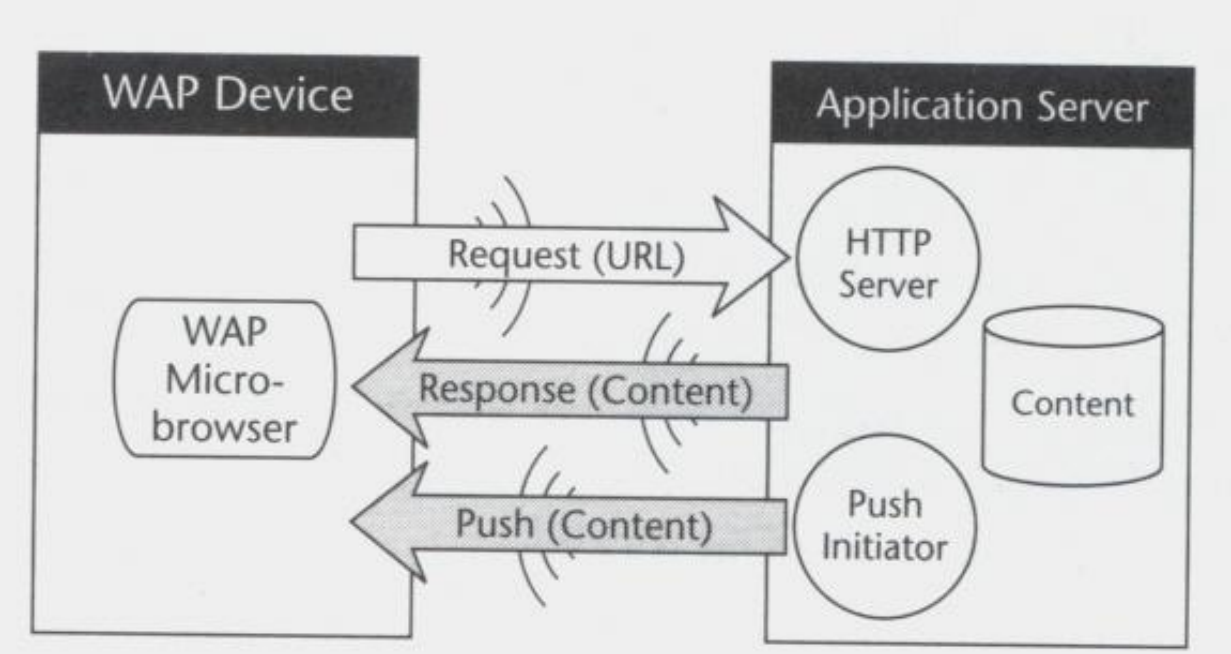


Using a wireless gateway



- **Gateway's functions:**
  - Translating requests/results between protocols: WAP <-> TCP/IP/HTTP
  - Encoding/Decoding web content
  - Implementing "push" functionality using WTA (Wireless Telephony Application Specification) - Information "pushed" to WAP devices

Without wireless gateway

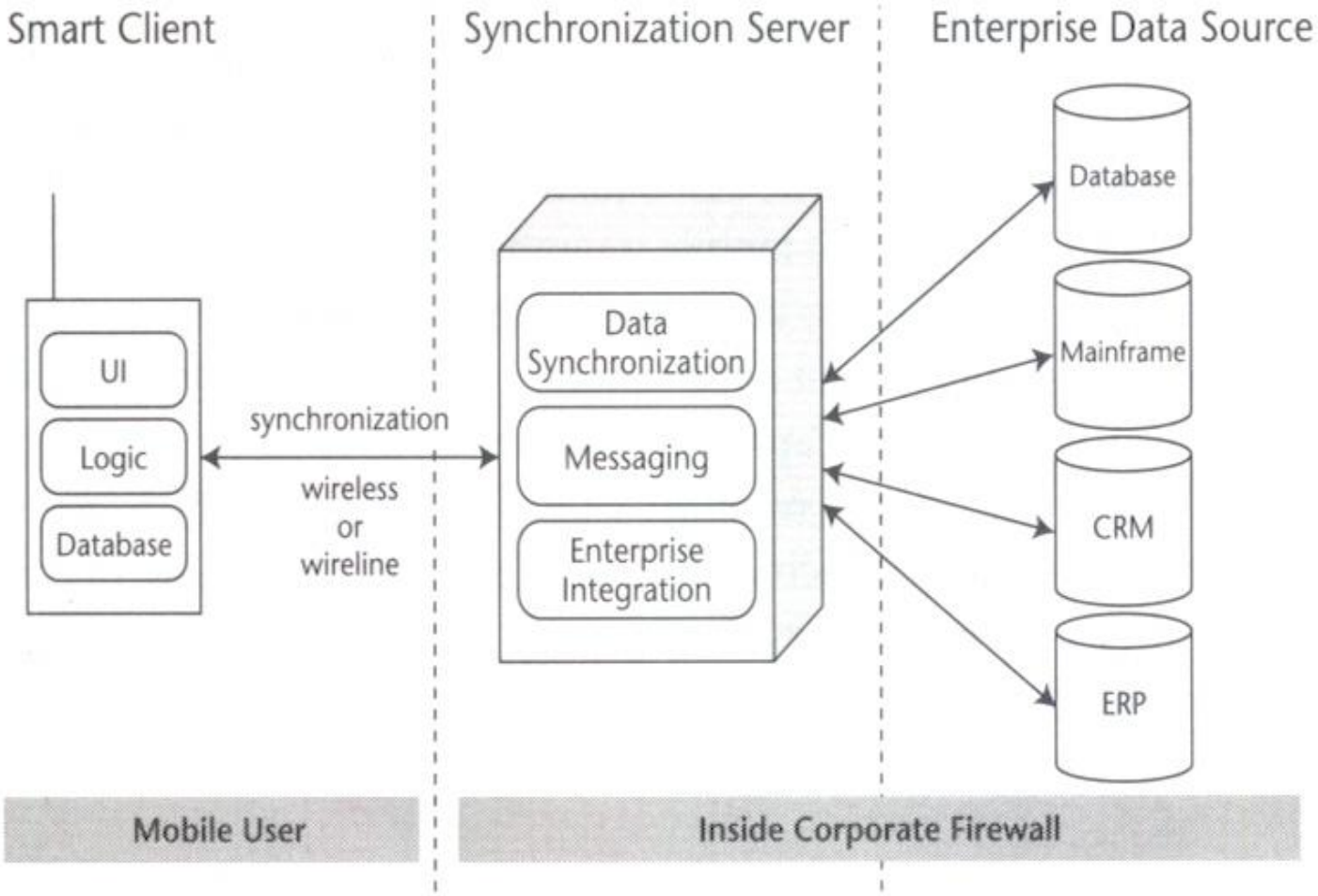


- WAP2.x no longer requires a WAP gateway
- Allows HTTP communication between the client and the original server - No need for conversion
- But you can still benefit from using a gateway, if you want to optimize communication and facilitate other wireless services, such as location, privacy, WAP push.

WAP vs. HTML5?

- What is HTML5?

Smart Client architecture



Smart Client

- Execute **application logic** and messaging, either a native executable or a Java application
- Provide **UI**: the interface to the end user
  - can have rich and sophisticated user interface.
  - programmed with development tools
  - usability!
- Have **persistent data storage**: client-side data
- Communicate to the enterprise system's **synchronization** or **messaging** layer either wirelessly or over wireline connection.

Data synchronization

- Backup of data from mobile devices
- Update data on mobile devices
- Propagation of updates between desktop and mobile devices
- Transaction Processing *(e.g., email, data collection)*

Data synchronization

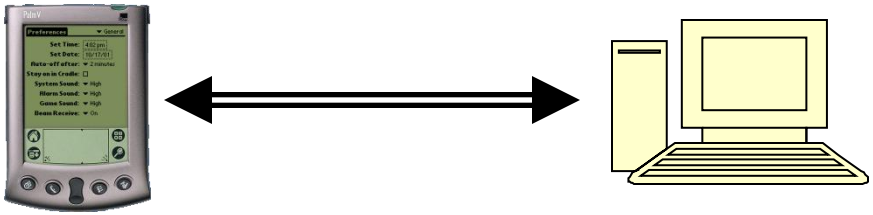
- **Synchronization software:**
  - ensure consistency among different copies of the same data on clients and the server.
- **In most cases, synchronization is executed on the server but client side still needs to know certain synchronization knowledge**
  - location of the synchronization server,
  - data to be synchronized etc
- **Most vendors provide a synchronization client module:**
  - Microsoft's *ActiveSync*
  - Apple's *iTunes*
  - BlackBerry's *Desktop Software*
  - 

Synchronization server

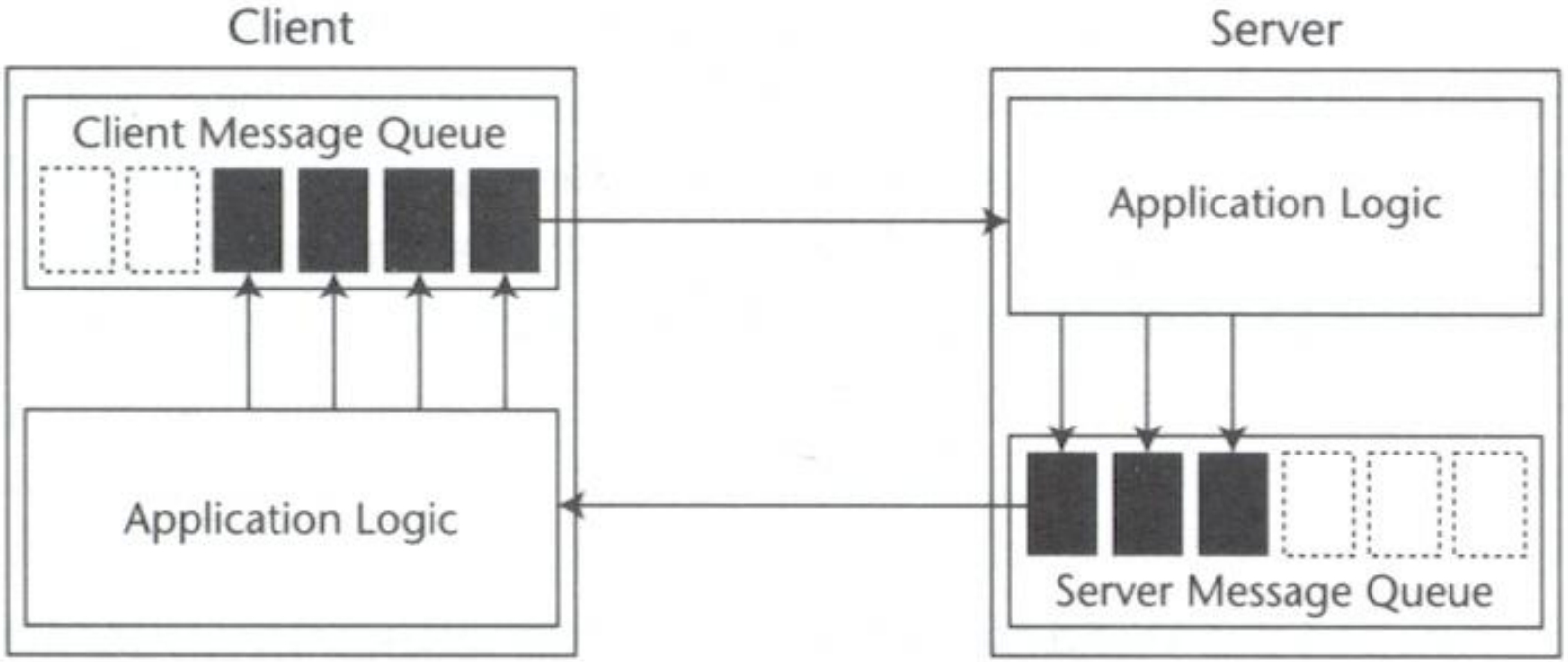
- **Receives data from the client application and communicates to the enterprise data source.**
  - Ensuring the minimal amount of data transferred.
  - Data can be exchanged in a variety of formats (vCard, vCal, various XML-based formats (XML, SyncML))
  - Integration with back-end data source is often done using JDBC or ODBC.
- **Synchronization logic:**
  - Executes protocols which can simply take the updates and apply them to data store, or perform more complex actions, e.g., detecting & resolving conflicts.
  - SyncML Initiative
- **Support for *disconnected operations* –we will talk about this later**

Why is synchronization difficult?

- Synchronization logic can be very complex
- Can be difficult to map fields from various sources
- Tree structures are tricky to synchronize (categories, directory, keywords)
- Allowing many-to-many synchronization is hard to do efficiently



Mobile Messaging architecture



Mobile messaging

- **Messaging client**
  - Store-and-Forward:
  - When connection is unavailable, message can be stored in an outgoing queue.
  - Sends automatically when the connection was established to the server.
- **Messaging server**
  - Communicates with messaging client and to the enterprise system.
  - MOM (message oriented middleware).
  - Many of the systems were built on JMS (java message service).
- **Enterprise data source**
  - Messaging server can interact with variety of back end systems like databases, business applications and other messaging systems.

Syn vs. Asyn. messaging

- **Synchronous messaging: blocking**
  - Sender and receiver of the message must be ready at the same time.
  - May lead to poor user experience.
  - "Only 20% of inter-application communication is synchronous, 80% is asynchronous"

Syn. vs. Asyn. messaging

- **Asynchronous Messaging: *nonblocking***
  - Sender only initiates the operation and doesn't need to wait for response before continue working.
  - When response appears, sender receives the message and respond appropriately.
  - User can continue to perform other tasks while waiting for a response.
    - Even if the message takes time to complete, user may not notice the delay at all.

Push vs. Pull messaging

Pull messaging

- Client periodically polls the server to check if there is a new message for it. If yes, it will retrieve the message and react accordingly.
- Can be used to enhance smart client applications
  - e.g., to check whether new set of data is available on the server

Push vs. Pull messaging

Push messaging

- Used to notify the mobile user when specific event occurs in the enterprise.
- Push information to the mobile user without user interaction – user should not be required to request the information manually.
  - In practice, however, often the client has to do some form of check to see if new data is present.
- HDML notification and WAP Push

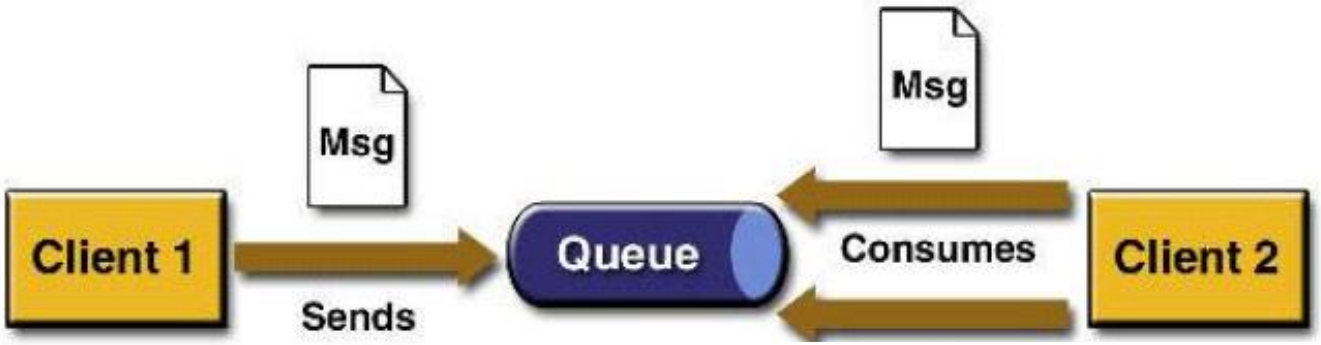
User-to-User vs. Application-to-Application messaging

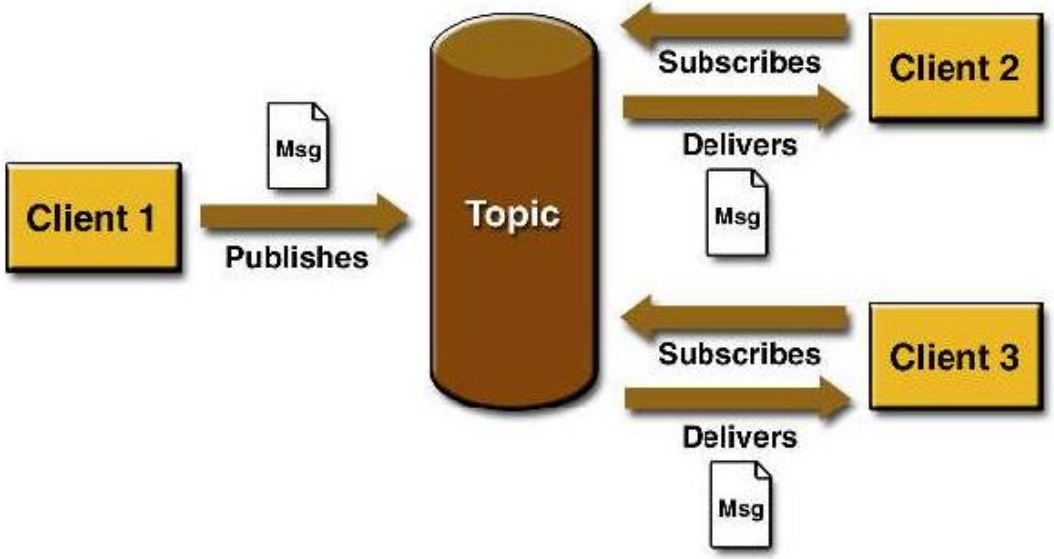
- **U2U: messages sent from one user to another**
  - Email, paging / SMS, instant messaging (IM)
  - Graphics and formatted text sent by EMS (enhanced message service)
  - Multimedia content sent by MMS (media message service)

U2U vs. A2A messaging

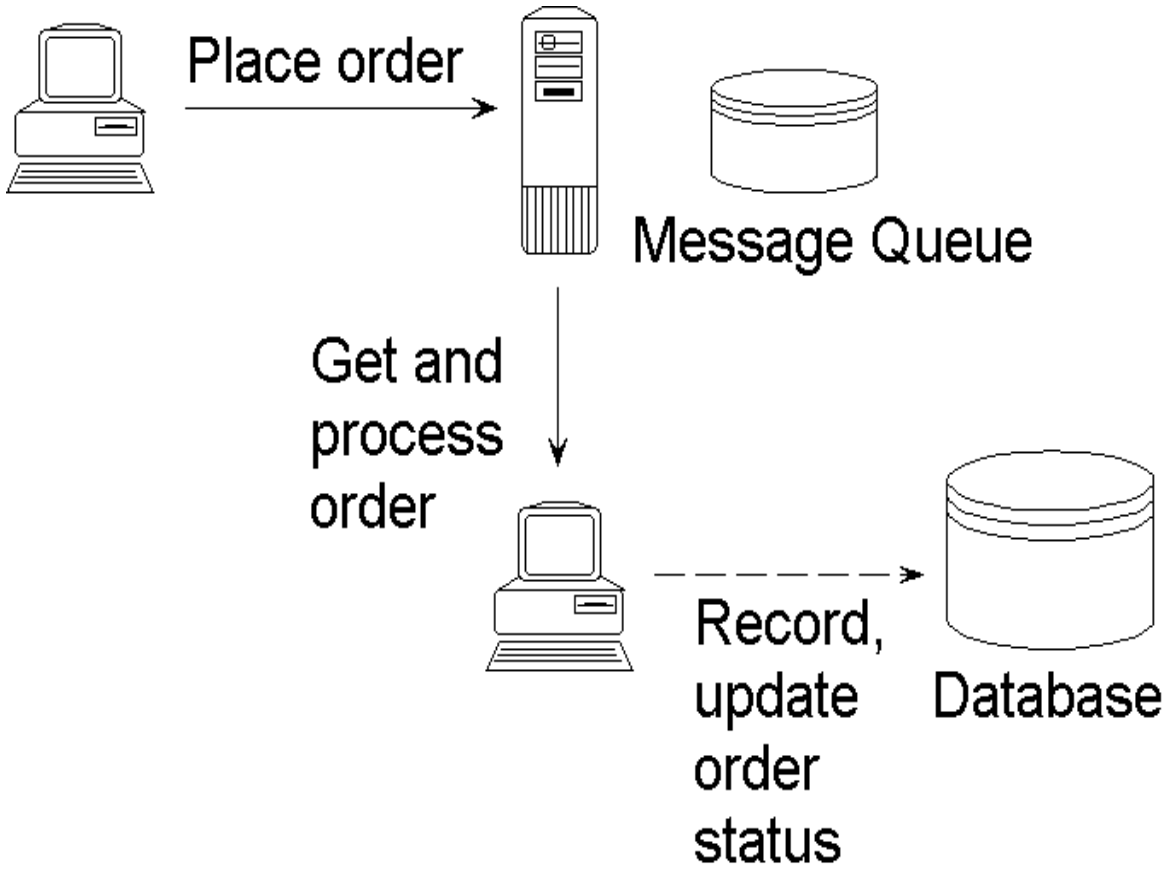
- **A2A: In many cases user interaction is not needed for messaging. Applications communicate data directly with each other.**
- **Rather than using standard client software, A2A messaging is incorporated in custom applications.**
  - Client must be able to both send and receive messages
  - When receiving message, client needs to relay the information appropriately to user or application
    - Notifying user directly
    - Updating a set of data in client data store

Point-to-Point messaging vs. Pub/Sub messaging

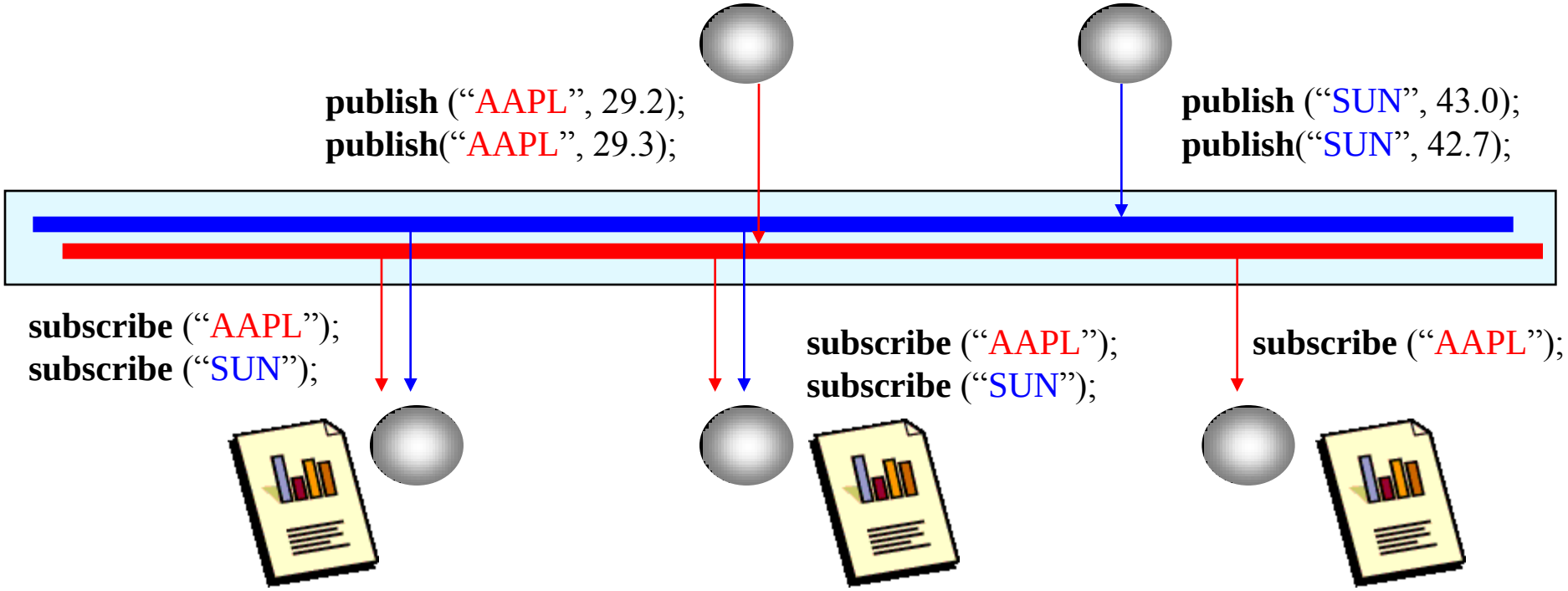




Point-to-Point messaging



Pub/Sub messaging



Email

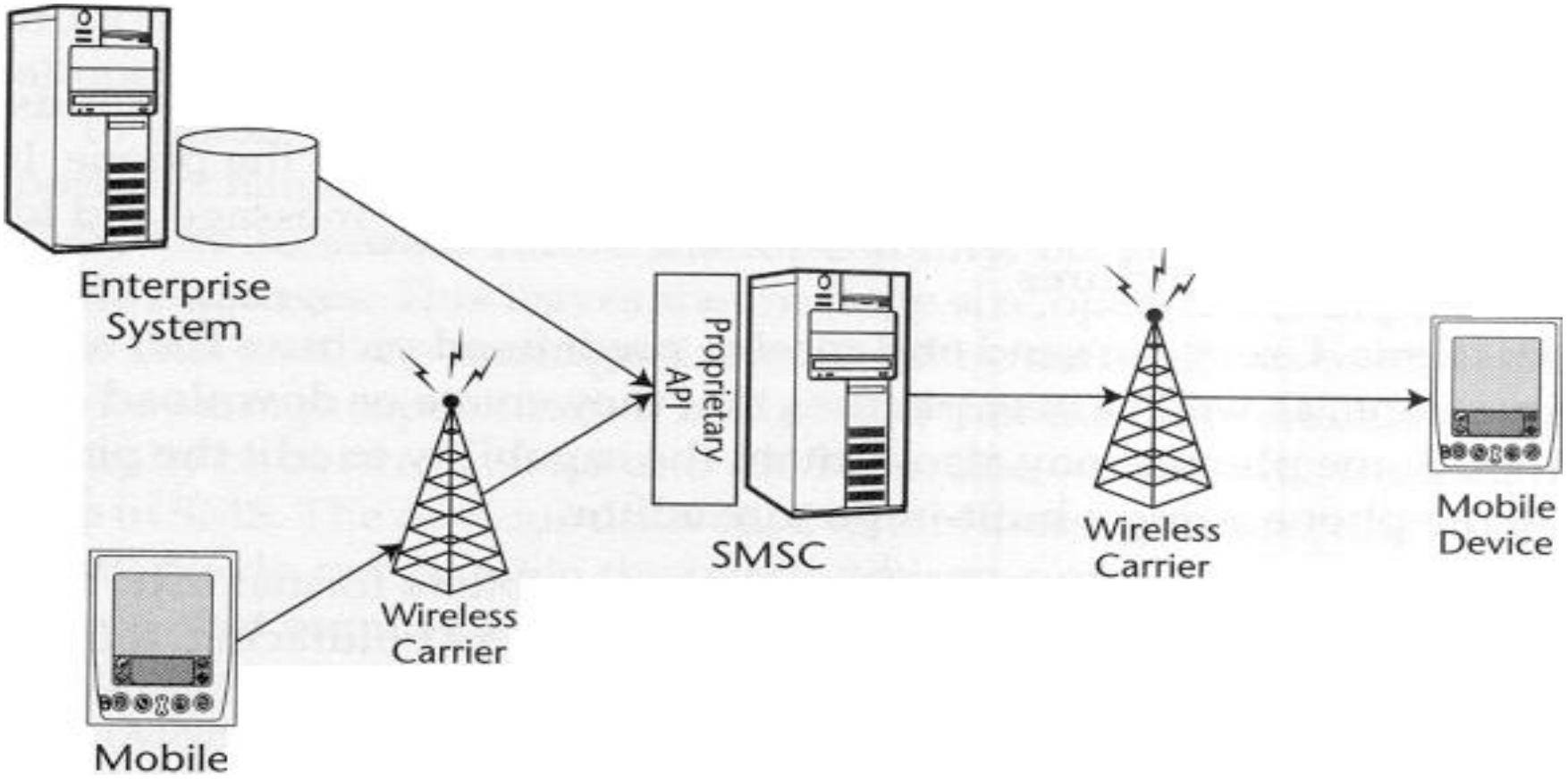
- **The killer application of messaging - billions of messages are sent everyday.**
- **Email client can be either stand alone or browser based, and has to communicate with an email server to send and receive messages.**
- **Most widely used servers**
  - SMTP (Simple Mail Transfer Protocol) server used to send messages
  - POP (Post Office Protocol) server used to store incoming mail.
  - IMAP (Internet Message Access Protocol) is an alternative to POP which allows users to access email from multiple machines.
  - JavaMail API: its implementation provides access to mail services using POP3, IMAP, and SMTP.

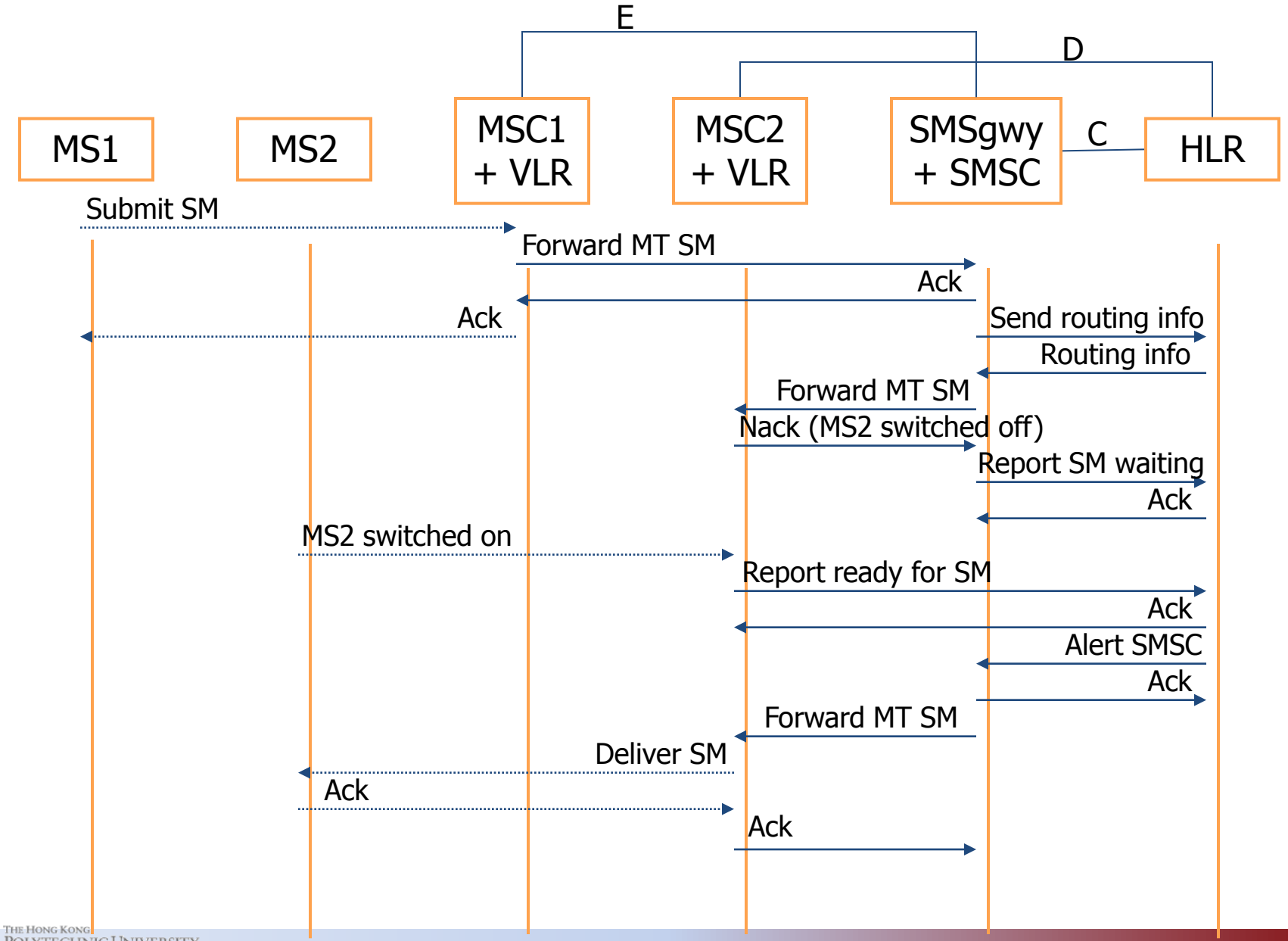
Short Message Service(SMS)

- **Introduced in Europe in 1991, supported on digital wireless networks such as GSM, CDMA, etc.**
  - Used to send and receive short text messages (160 characters) - eliminates the need of pager
  - SMS messages can be received during voice call.

- **Many applications: both consumer-oriented corporate applications**
  - Personal communications
  - Information services: stock, weather, sports, etc.
  - Advertising
  - Location tracking; Remote monitoring
- **Benefits:**
  - Guaranteed message delivery using store and forward.
  - Low cost
  - Revenue source for service providers (~200 billion messages/yr; ~$16 billion annual revenue).

SMS architecture




E
D
MS1
MS2
MSC1 + VLR
MSC2 + VLR
SMSgwy + SMSC
C
HLR
Submit SM
Forward MT SM
Ack
Ack
Send routing info
Routing info
Forward MT SM
Nack (MS2 switched off)
Report SM waiting
Ack
MS2 switched on
Report ready for SM
Ack
Alert SMSC
Ack
Forward MT SM
Deliver SM
Ack
Ack

SMS architecture

- **Messages sent from a mobile phone are taken care by carrier through the SMSC to the destination device**
- **Messages from other sources will not be taken care by the carriers but need the help of the SMSC for delivering the message**
  - May need to know SMSC's APIs
  - Standard protocol used to communicate with SMSCs include TAP (Telocator Alphsnumeric Protocol) and SMPP (Short Message Point to Point).

SMS architecture

- **Another possible way to send SMS is to use email interface (SMTP).**
  - Many carriers make SMTP access available to their systems, allowing SMS messages to be sent using an email interface (e.g., Java Mail).
  - Send email message to destination user's telephone number and carrier domain (8005551234@mobile.att.met).
  - Message is routed through SMTP server to SMSC, then delivered to mobile over wireless network.
  - JavaMail API provides a generic model of an email system, and Sun Microsystems' reference implementation of JavaMail supports the three most popular email protocols – SMTP, POP3, and IMAP

Multimedia Message Service (MMS)

- More data format, including voice, audio and video clips and presentation information.
- Standardized by **OMA (Open Mobile Alliance)** for message encapsulation and application protocols, and **3GPP (3rd Generation Partnership Project)** for network architecture and general functions
- 1st generation MMS size is 30 KB to 100 KB, requiring 2.5 G wireless network with minimum bandwidth of 14.4 Kbps.

MMS features

- **Messages are laid out as slide shows, containing at least one slide divided into two sections (one for text and another for multimedia).**
- **Actual contents are separate files sent along with the slides, incorporated to the slide show using SMIL (Synchronized Multimedia Integration Language).**
  - SMIL (from W3C) is based on XML, and used to control the presentation of multimedia elements

MMS architecture

- **Similar to SMS in delivery - use MMSCs (similar to SMSCs), which use a store-and-automatic forward mechanism.**
  - Supports for email addressing so messages can be sent to an email address from the MMS client.
- **Each vendor's MMSC has its own APIs. But unlike SMSCs, MMSCs are not expected to provide an SMTP interface.**
- **Transport of MMS is accomplished using WAP transport, making MMS bearer independent and possible to use WAP push to deliver messages.**

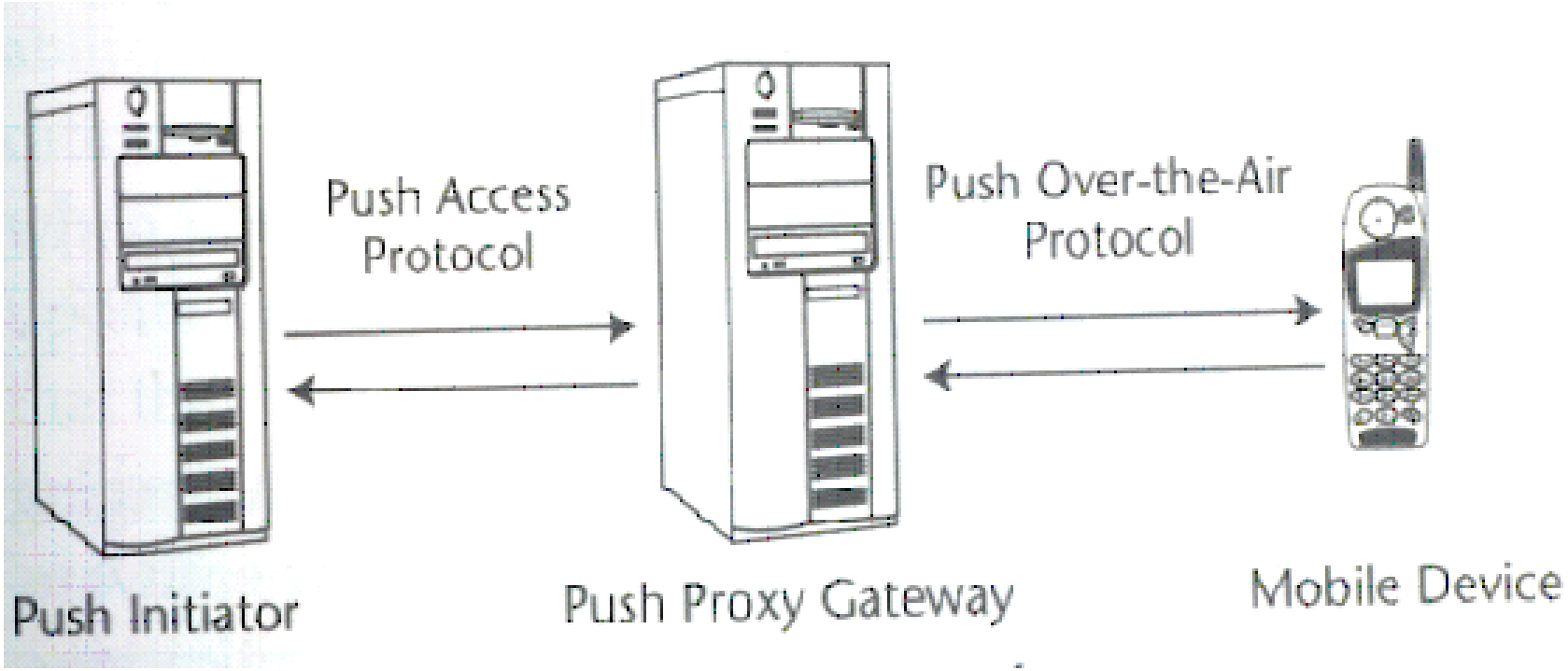
Instant Messaging (IM)

- **IM is the next killer application for wireless industry - fastest growing communications function on the Internet**
  - Over 130,000,000 worldwide users
  - 3,000,000 new users every month
  - 1,000,000,000 instant messages are sent every day !
- **Provides capabilities similar to paging, SMS, and email with additional feature “Presence"**
- **Mobile IM enables users everywhere to communicate with one another, regardless of their type of connectivity.**
  - Several mobile IM products are available from Microsoft, Yahoo, ...
  - True mobile IM requires interoperability between IM services.
  - Wireless Viliage (OMA): A joint effort (Nokia, Motorola, and Ericsson) to create standards of IM for handset makers and carriers to follow.

Push messaging

- **Urgent messages (notifications & alerts) are sent to mobile users through wireless devices which contains URL link called *actionable alerts*.**
- **Receiver performs the action based on the information.**
- ***WAP push* is the industry standard (by WAP Forum) for pushing content to WAP-enabled devices.**
  - Allows multiple gateway vendors, microbrowser providers, and wireless carriers to communicate using the same protocol.

WAP Push architecture



WAP Push: how it works

Push Initiator (PI)

- An application that pushes the content (in XML format) and delivery instructions to the PPG.
- PI must have two sets of information about the destination, domain name of the PPG and client address.
- Runs on standard web server and communicates with PPG using PAP (Push Access Protocol, currently runs on top of HTTP)

WAP client

- Typically a wireless device that contains a WAP microbrowser capable of receiving WAP push content.

WAP Push: how it works

Push Proxy Gateway (PPG)

- **Mainly responsible of delivering push content to the WAP client.**
- **Store the message when they can't be delivered immediately to the client.**
- **Maintains the status of each message, allowing PI to cancel, replace and request status of the message.**
- **Uses OTA (Push Over The Air) protocol to deliver push content to WAP client over wireless network.**
  - OTA runs on top of HTTP or WSP (wireless session protocol)
- **Can respond to PI using "result notification" (an XML document), indicating whether message delivery was successful or not.**
  - Successful delivery occurs only when the WAP device has taken the responsibility for the pushed content.

WAP Push operations

- **WAP push specification adds three MIME types for delivery from PI to WAP client.**
  - **Service indication (SI)**
    - Provides the ability to send notification directly to the end users.
    - Contains information directly in the message or in URI form.
  - **Service loading (SL)**
    - Often behind the scene - without requiring user interaction.
    - Contains URI that points to content to be loaded into the browser's cache to make interaction efficient
    - Indicate whether the content should be executed immediately or placed in the cache memory.
  - **Cache operation (CO)**
    - Removes objects from the client's cache.
  - **SI and CO are optional.**

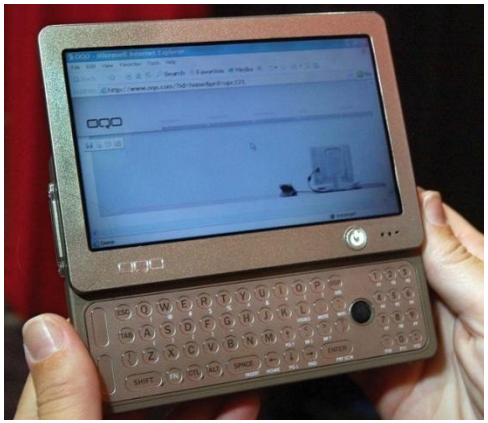
Mobile Device Platforms

Mobile device platforms

- **In addition to wireless networks, mobile devices is another important underlying technology for mobile computing.**
- **Mobile device platform includes *device hardware* itself and the *software platform*: OS and program development kit.**
- **We are seeing increasing device capabilities and functionalities**
  - More memory, more processing power, more battery life, larger and richer screens, better bandwidths.
  - Except input – which has remained largely unchanged

Mobile devices

- Pagers
- Cellular phones
- Portable media players
- Personal digital assistants (PDAs)
- Tablet PCs
- Laptop computers
- Mobile Internet devices (MIDs)



Converged technologies

- **Smartphones: cell phones with smart add-ons, e.g.:**
  - PDA capability
  - Fancy color graphics
  - MP3 player
  - GPS
- **PDAs with cell phone capability**
- **Sony Ericsson P800**
- **iPhone**

OS for mobile devices

- **To deploy smart client applications, client devices should have a mobile OS.**
- **Provided features**
  - Multitasking
  - Persistent storage and memory management
  - Power management
  - Wireless networking protocols such as Bluetooth, TCP/IP.
  - Programming languages and development tools
  - Data synchronization with server-side data sources.
  - Applications
  - Device emulators

OS for mobile devices

- **Main OS & platforms**
  - Palm OS
  - Symbian OS (Nokia)
  - Window CE / Window CE .NET (Windows Mobile)
  - Windows Phone 7 (WP7)
  - Blackberry OS
  - Linux
  - Google's Android
  - iPhone OS and SDK
  - J2ME

Windows CE, Palm OS

Symbian OS, J2ME

iPhone's iOS platform

- iOS 4.3 (iPhone OS)
  - Derived from Mac OS X
  - iOS SDK
- Novel Feature
  - Multi-touch interface
  - Multi-tasking
  - Sensors
- iOS devices
  - iPad
  - iPod Touch

Google Android

- **So far, most mobile device platforms are closed systems owned by companies and consortium**
  - High royalty fee, OEM/ODM not happy
  - Low application development efficiency
  - Difficult to collaborate and re-use among developers
- **Android is a software platform and operating system for mobile devices, based on the Linux kernel, developed by Google and later the Open Handset Alliance**
  - The main goal is to make the best possible user experience available on a mobile device - an open platform allows just about anyone who wants the chance to get in on the game.
  - Andriod's API is freely available for any programmer to create Android applications

Google Android

- **Open Handset Alliance:**
  - A consortium established in 2007, led by Google, 48 hardware, software, and telecom companies: Intel, Motorola, T-Mobile, HTC, etc
  - Develop open standards for mobile devices
  - Main product: Android Platform (Nov. 2007), available as open source
- **October 22, 2008: HTC delivered G-Phone - Android capable smartphone**
- **Jan. 5, 2010: Google releases Nexus One**

Blackberry OS

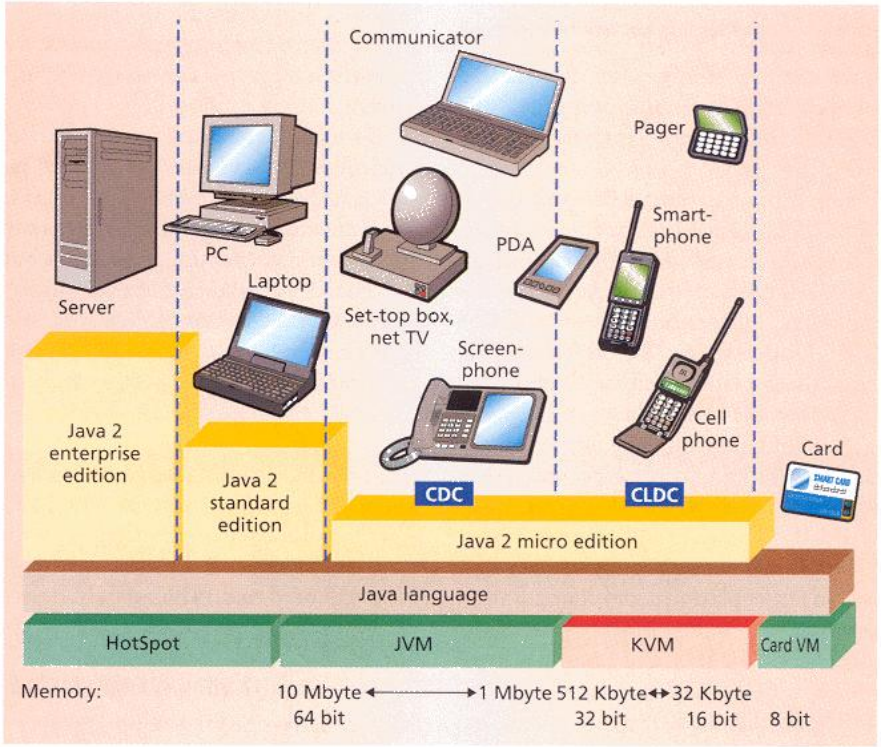
- Proprietary software platform made by Research In Motion for their BlackBerry line of handhelds.
  - Provides multi-tasking,
  - Makes heavy use of specialized input devices, particularly the thumbwheel,
  - Support a subset of MIDP2.0 and WAP1.2
  - Allows complete wireless activation and synchronization with Exchange's e-mail, calendar, tasks, notes and contacts,
  - Third-party developers can write software using these APIs, and proprietary BlackBerry APIs.
    - Fast Web browsing, Video camera, Youtube,

J2ME

- J2ME, now called Java ME, is Sun's answer to the development of Java services on lightweight, limited computing platforms.
  - Consumer devices
  - Embedded systems
- Support both connected / disconnected operations
- Rich user-network interaction model
- Fully programmable
- Portable

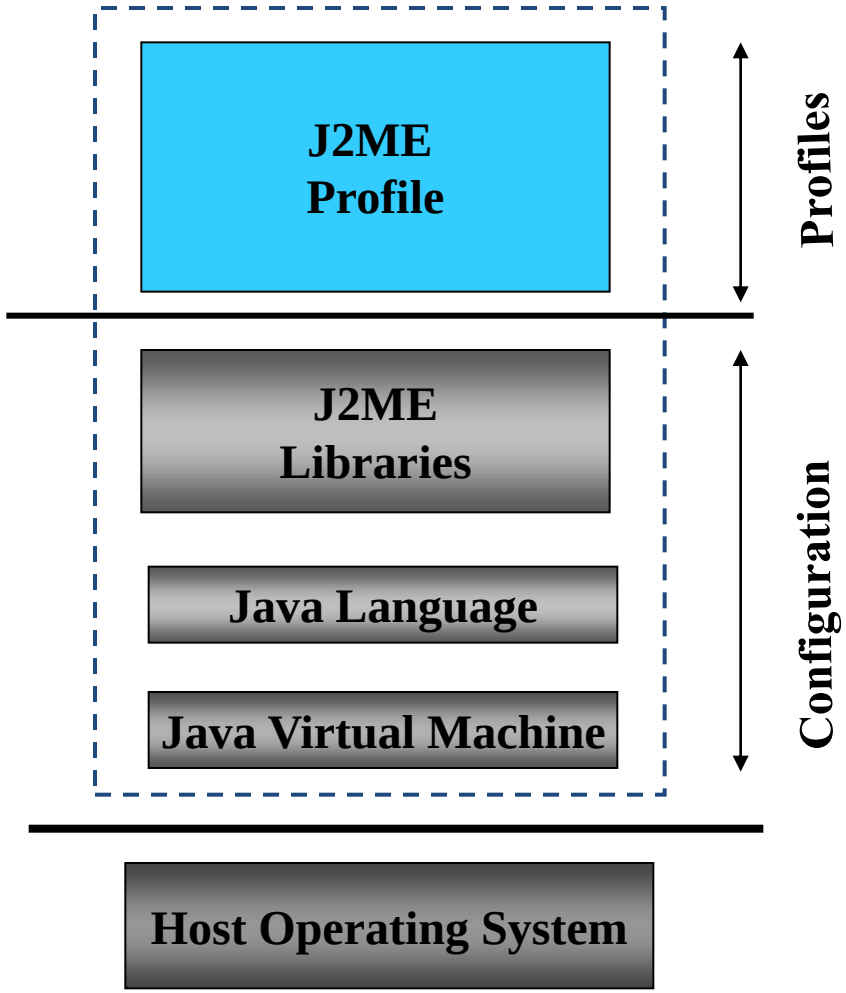
J2ME configurations

- J2ME currently is designed to target two types of devices (called configurations)
  - **CDC (Connected Device Configuration)** - targeted at shared, fixed (permanently connected) devices
  - **CLDC (Connected Limited Device Configuration)** - at mobile, personal that have low bandwidth & transient network connections.



J2ME architecture

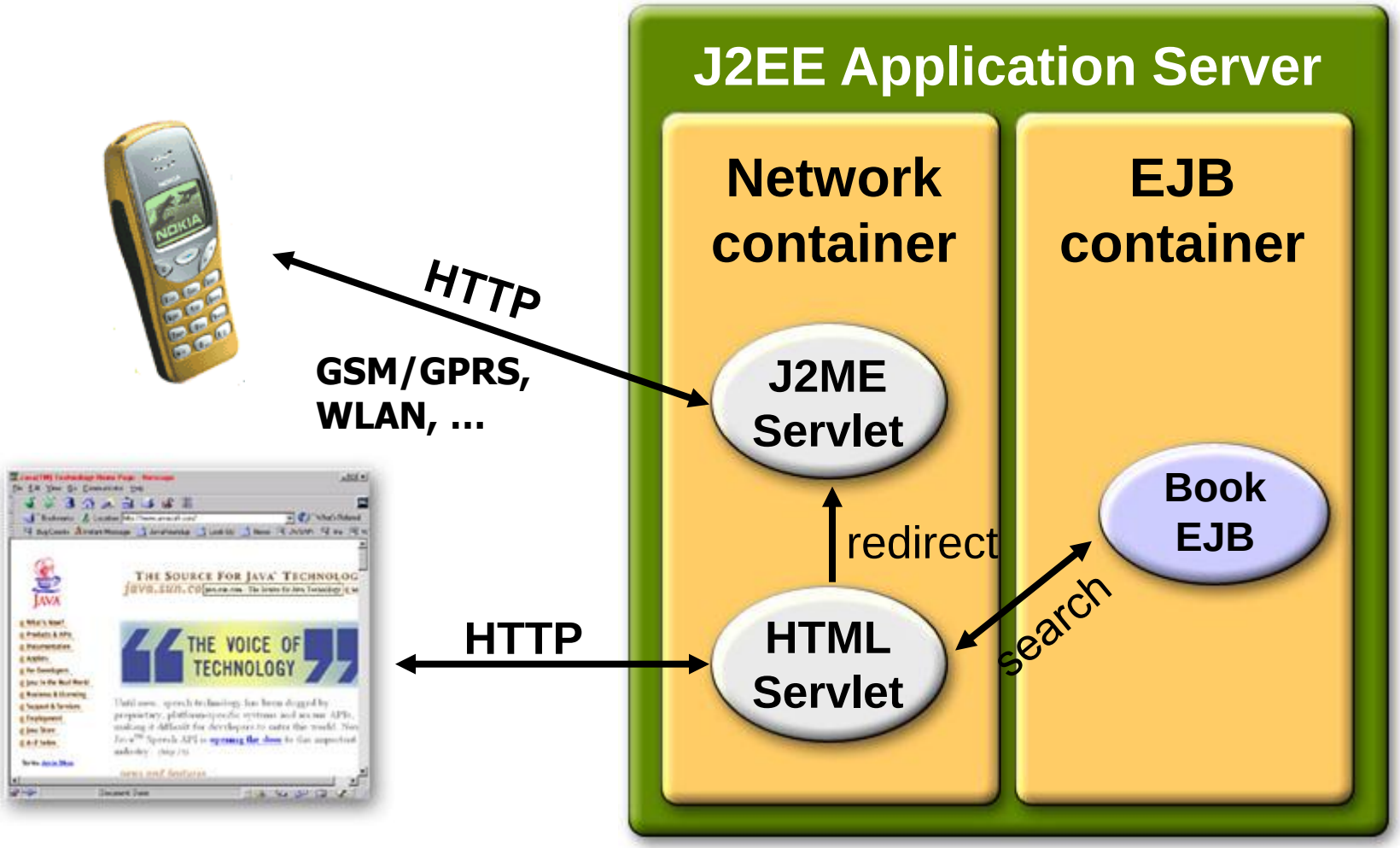
- J2ME consists of
  - **Configuration**
    - **JVMs** that fit inside the range of consumer devices
    - A **library of Java APIs** that are common for each type of devices
  - **Profile**
    - Additional set of Java APIs specialized for a particular kind of devices of that type, e.g., PDA and mobile phone
    - MIDP (*Mobile Information Device Profile*) is a particular profile specified for wireless handheld devices.



CLDC/MIDP Wireless Toolkit

- **CLDC 2.0 specification is available for download free of charge.**
  - Java language and virtual machine features, core libraries, input/output, networking and security are the primary topics addressed by the specification.
  - SUN provides reference implementation of CLDC on KVM.
- **MIDP 2.0 specification and implementation are also available for download.**

J2ME talks to J2EE



J2ME extensions

- http://www.jcp.org/en/jsr/all
  - 120, 205: Wireless Messaging API
  - 135: Mobile Media API
  - 169: JDBC Optional Package for CDC/Foundation Profile
  - 82: Java APIs for Bluetooth
  - 179: Location API for J2ME
  - 184: Mobile 3D Graphics API for J2ME
  - 172: J2ME Web Services Specification
  - 177: Security and Trust Services API for J2ME
  - 180: SIP API for J2ME
  - 190: Event Tracking API for J2ME

The trend

- **Focus shifted from device to OS platforms and application software.**
- Most mobile devices now have similar design and form factors, so OS platforms and application software become the key for competition in attracting the users.
  - WMC-2010 in Spain – Nokia, LG, Apple not present, but Google's Eric Schmidt and Microsoft 's Steve Ballmer showed

The Trend

- **The biggest problem now is the incompatibility among the OS platforms and application software.**
- **Big equipment manufacturers, mobile operators are forming alliance and pushing standards for mobile application software**
  - Alcatel-Lucent (www.**alcatel-lucent**.com)
  - Joint Innovation Lab (JIL): formed by China Mobile, Verizon Wireless, Vodafone, Softbank Mobile (http://www.jil.org/)
  - Wholesale Applications Community: formed by JIL and AT&T, Sprint Nextel, Orange and other mobile operators (www.wholesaleappcommunity.com)