Using JMS & J2ME for Building Interactive Mobile Applications

Martin Erzberger

Softwired AG, Zürich

What you will learn

- Why Java on Mobile Devices?
- What is Java on Mobile Devices?
- Why JMS on Mobile Devices?
- What is JMS (Java Message Service)?
- Beyond JMS: Hayabooza

Why Java on Mobile Devices?

Market trends

- Appearance of communicator devices. (Communicator = Cellular phone + PDA)
- Appearance of packet-oriented wireless bearers: GPRS, EDGE, and UMTS.
- In 2004, mobile Internet devices will outnumber PCs with Internet access. (TIMElabs, Nomura Research).





Resulting Challenges

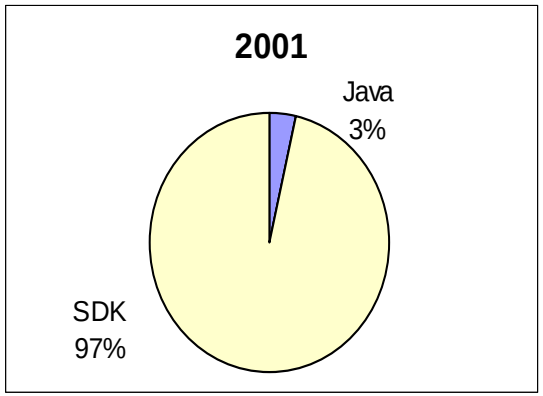
- First of all: Will a Microbrowser be enough?
- The battle about OS, Microchip, Bearer has only just begun. What to develop for?
  - Operating Systems: PalmOS, WinCE, Symbian
  - Bearers: SMS, (WAP), GPRS, UMTS, ...
  - Chips: StrongARM, Dragonball, ARM, …
- Network issues: Delays, interruptions, data loss
- Scalability: 100.000s of devices, and more
- Security, Accounting, Profiling, …

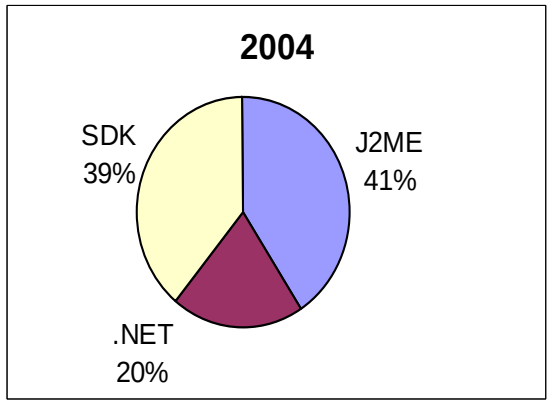
Solution: End-to-end Java

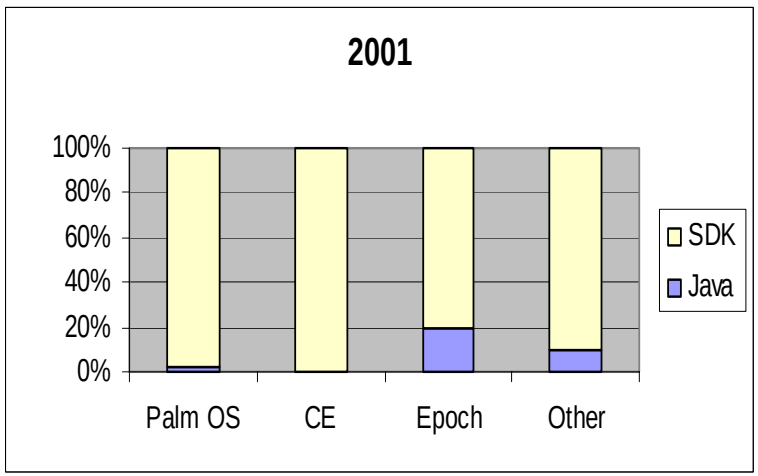
- On the mobile client:
  - Richer user experience (interactive maps, tickers, trackers, games, chat)
  - Less data transmission (local caching / computing)
  - Support disconnected operation (store-and-forward)
  - Device/bearer independence (write-once-go-anywhere)
- Standards
  - J2ME, PersonalJava

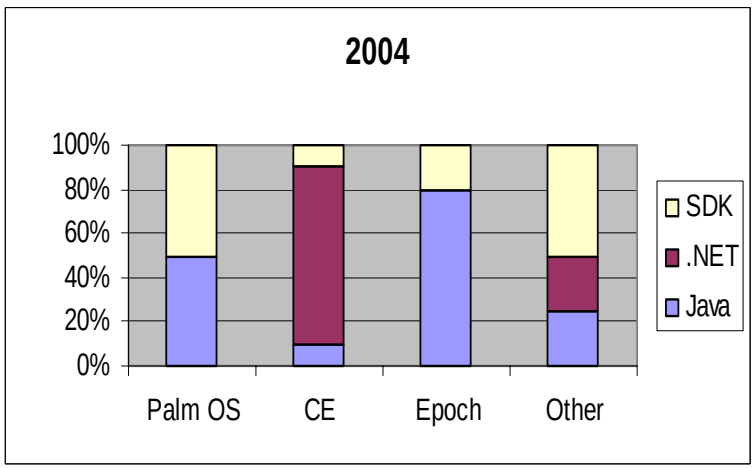
- On the server:
  - Decreased time-to-market (server applications can be developed more rapidly in Java. Reuse of J2EE components)
  - Better quality (Java is less prone to programming errors)
  - Motivated developers (Everybody wants to do Java !)
- Standards
  - EJB, JMS, JNDI ➡ J2EE

Market report [source: Gartner]









What is Java on Mobile Devices?

Java's Role In Wireless and Mobile Devices

network-based functionality

local functionality






Basic phone | Extended phone | Smart phone | PDA | HPC | Wireless notebook

Voice, SMS

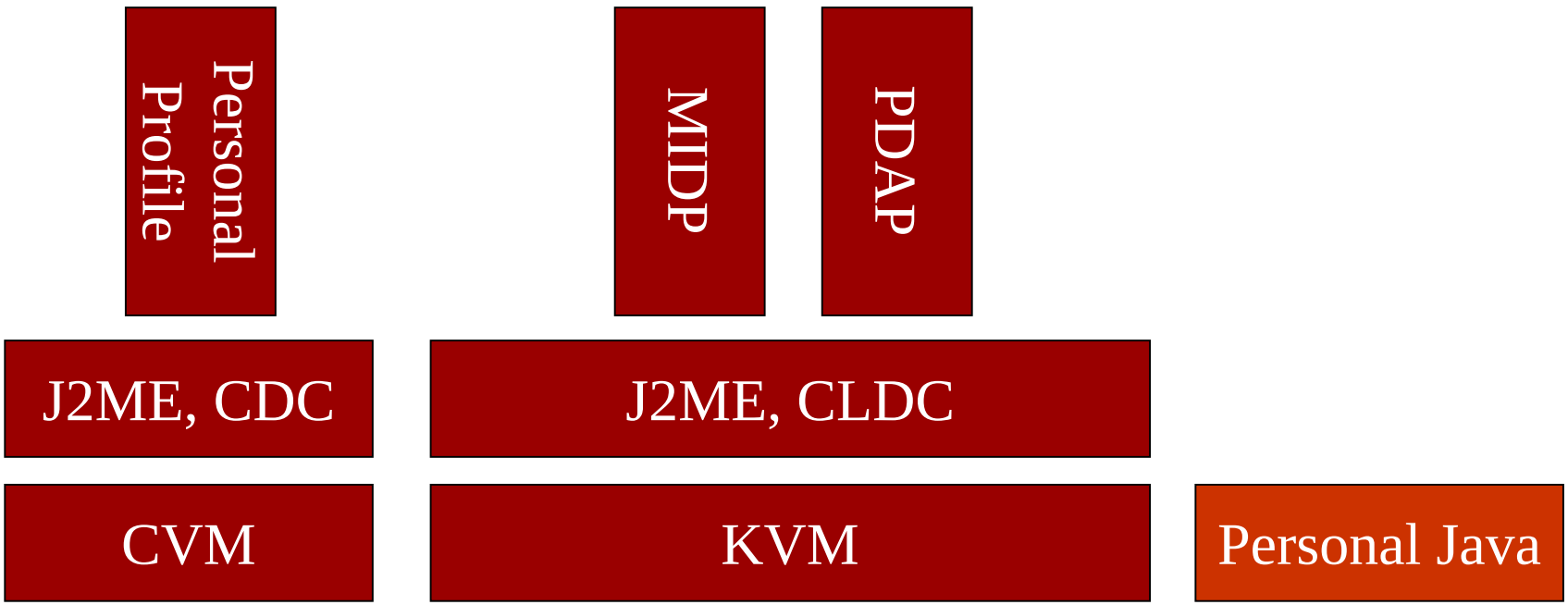
"Small device" OS | "Full" OS

HTML/XHTML Microbrowser (WAP) | Full Browser

J2ME (CLDC) | J2ME (CDC) | J2SE

The Java 2 Micro Edition (J2ME)

- Java tailored for resource-constrained devices
- Small core, extensible with "device profiles"



J2ME and Mobile Devices

PalmOS	•Now: CLDC, MIDP •Expected: PDAP
Windows CE	•Now: Personal Java •Expected: CDC + Profile
Symbian	•Now: Personal Java + JavaPhone API (Part of Epoc 6.0 and 6.1) •Expected: CDC + Profile

Why JMS on Mobile Devices?

MIDP

- CLDC offers the Connection Framework
- MIDP implements the HTTPConnection
- Basically a stream to read and write, with higher level HTTP parsing capabilities
- Other CLDC/MIDP implementations may offer additional Connectors (such as UDP or TCP)
- No higher level abstractions (RMI, JMS, RPC)

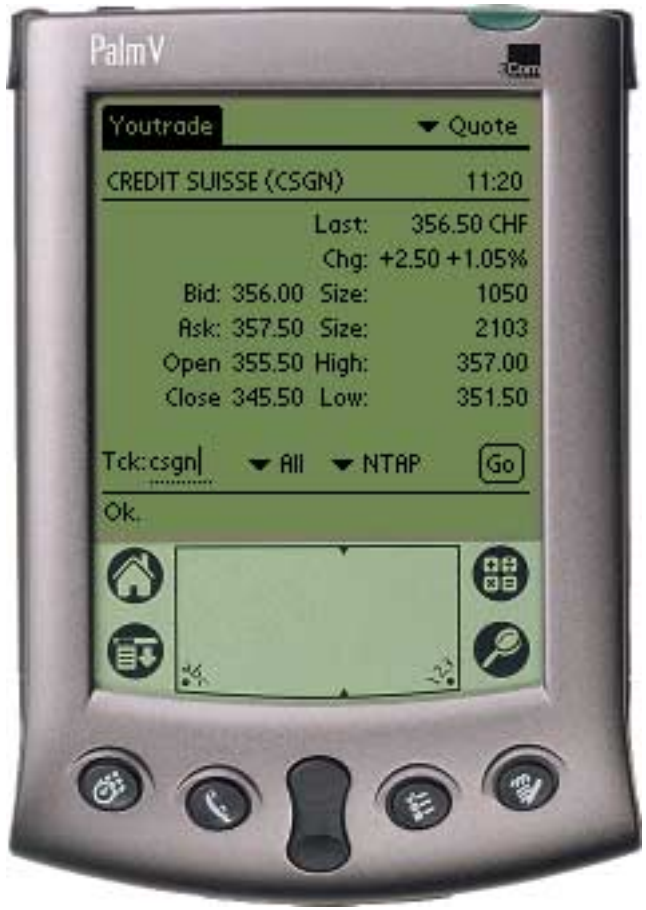
Something to Think About

- In wireline systems, it's an *exceptional case* when network connections break
  - This assumption is present in RPC-style middleware such as CORBA, RMI, and DCOM, and in most Socket-based applications
- In wireless systems, it's the *normal case* when network connections break
  - Middleware must cope with this fact.
  - Unlike RPC, Message-oriented middleware (JMS) was designed for these conditions

From Yesterday's University

```
catch ( IOException ex ) {
    System.out.println
     ("Error reading from http");
    ex.printStackTrace();
}
```

J2ME, CLDC Application



- Developed by Ergon (www.ergon.com) for youtrade.com (Credit Suisse)
- Uses JBed VM from Esmertec (www.esmertec.com)
- Requires persistent data connection (TCP) throughout session

JMS to the Rescue

- High Level Abstraction
- Can be implemented in a reasonable footprint
- Does not require additional services (such as Serialization, Reflection, TCP/IP stack, Naming)
- Can deal easily with unreliable networks
- Is the "missing link" between J2EE and J2ME

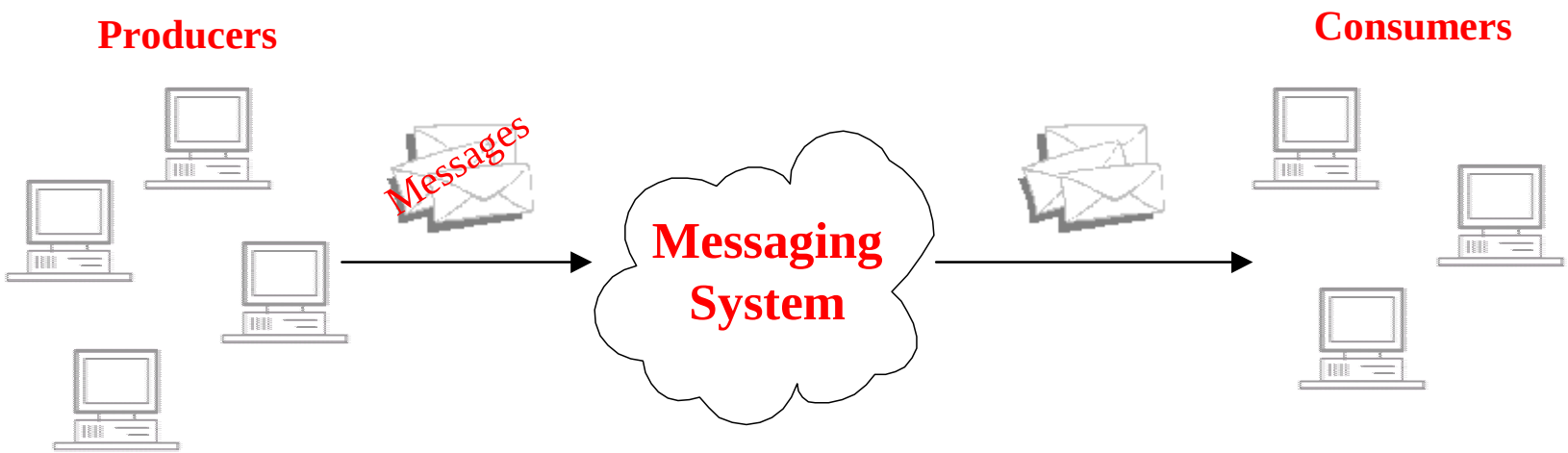
What is JMS (Java Message Service)?

What is JMS?

- JMS stands for "Java Message Service"
- Sun's Definition: *JMS is an API for accessing enterprise messaging systems from Java programs.*
- JMS is for *messaging systems* what JDBC is for *database systems*: A standardized API to access them.

What are Messaging Systems?

- Crude analogy: E-mail systems for applications

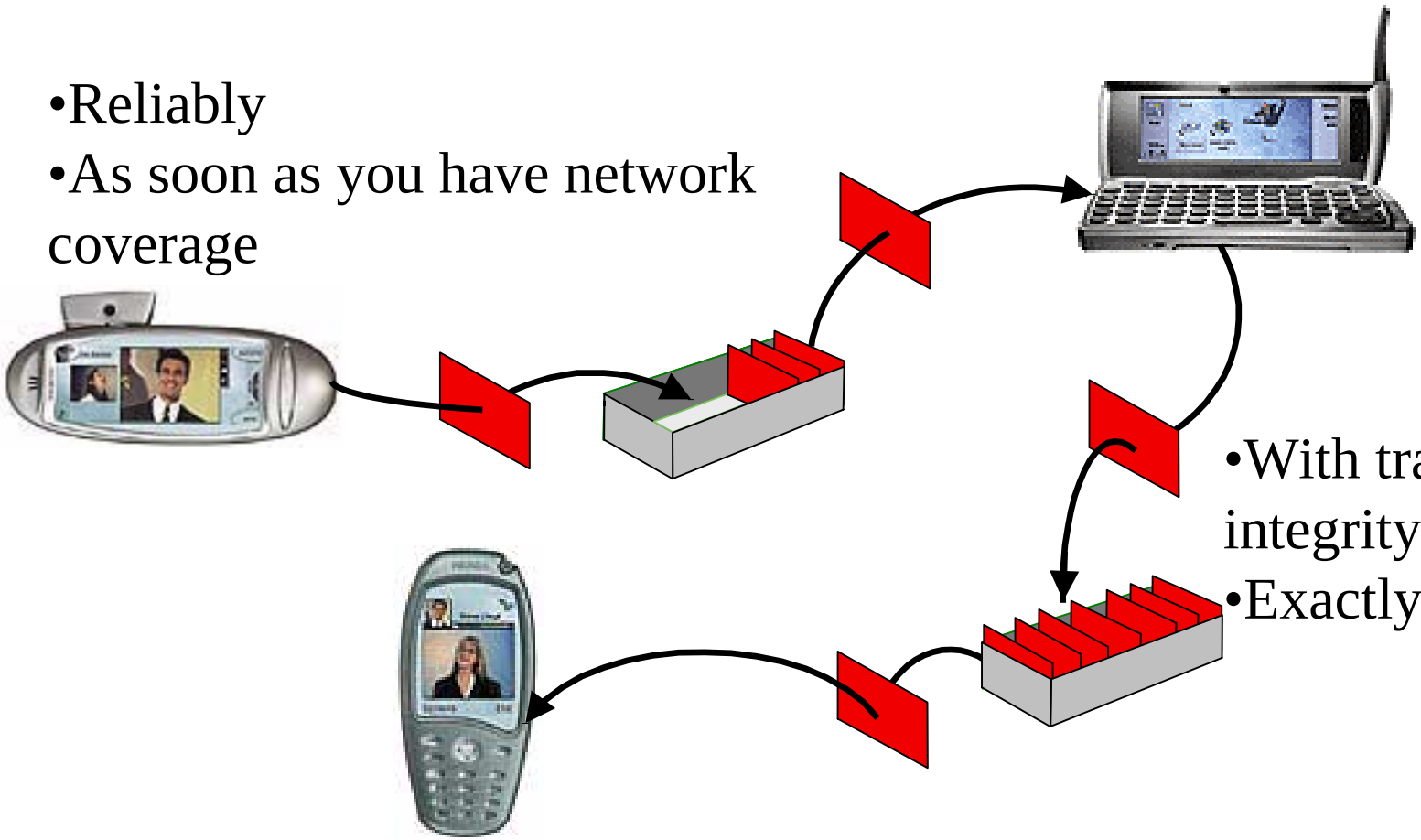


E-Mail for Applications

- Imagine
  - Mobile applications constantly sending little e-mails to each other
    - Address-Book entries
    - Calendar-Entries
    - News-Updates
    - Business-Transactions (Order, Confirm, Pay)
    - Delay-information, Traffic-Conditions, …

E-Mail for Applications (2)

- Reliably
- As soon as you have network coverage

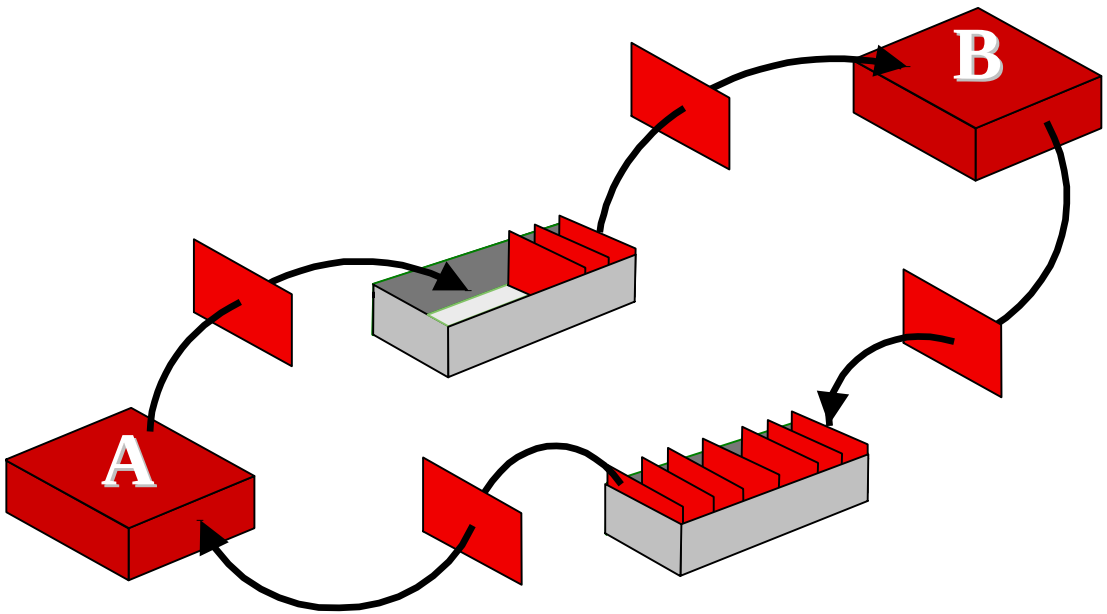


- With transactional integrity
- Exactly-once

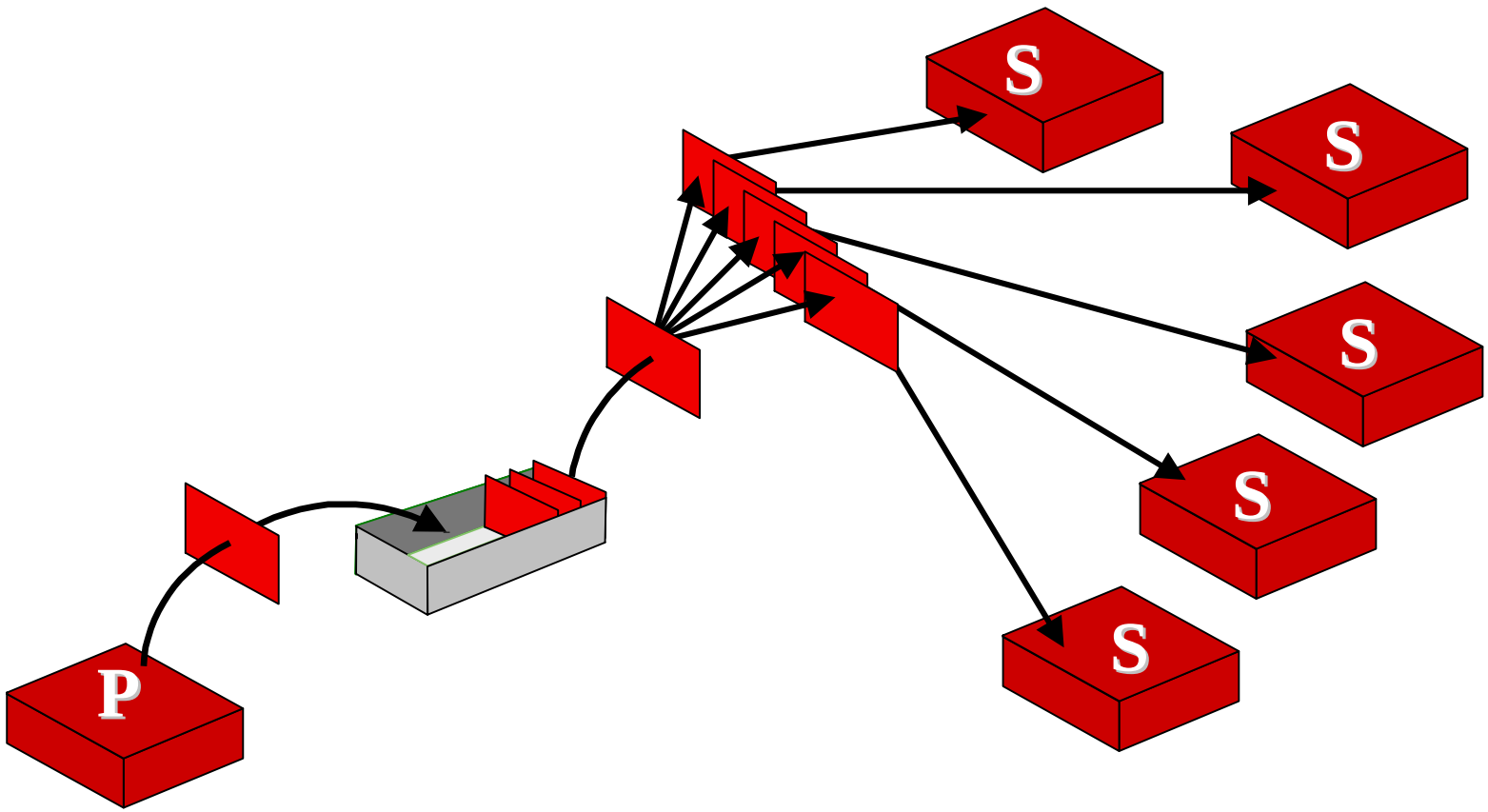
Prerequisites

- Device Support
  - "Always-on" capability, "connected" stand-by mode (like a phone being able to receive SMS any time)
  - Support for packet-oriented bearers
- Network
  - Packet-oriented bearer (GPRS, UMTS), Roaming-support for worldwide operation
- Middleware
  - Message-oriented, store-and-forward, scalable

JMS Point to Point



JMS Publish / Subscribe



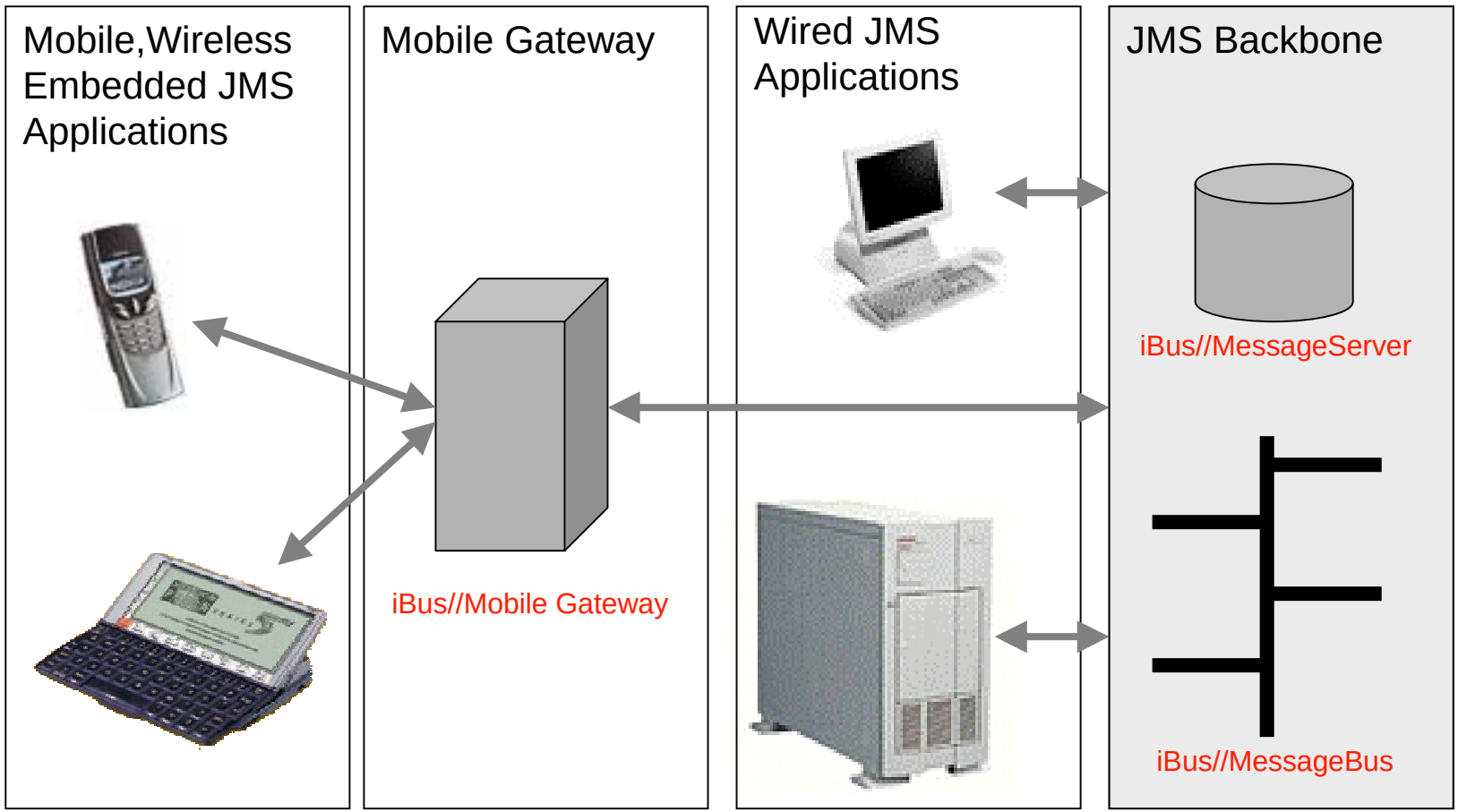
More about JMS…

- Tomorrow's Session N6:
  "Introduction to the Java Message Service (JMS) Standard"
- 4:45pm – 6:15pm

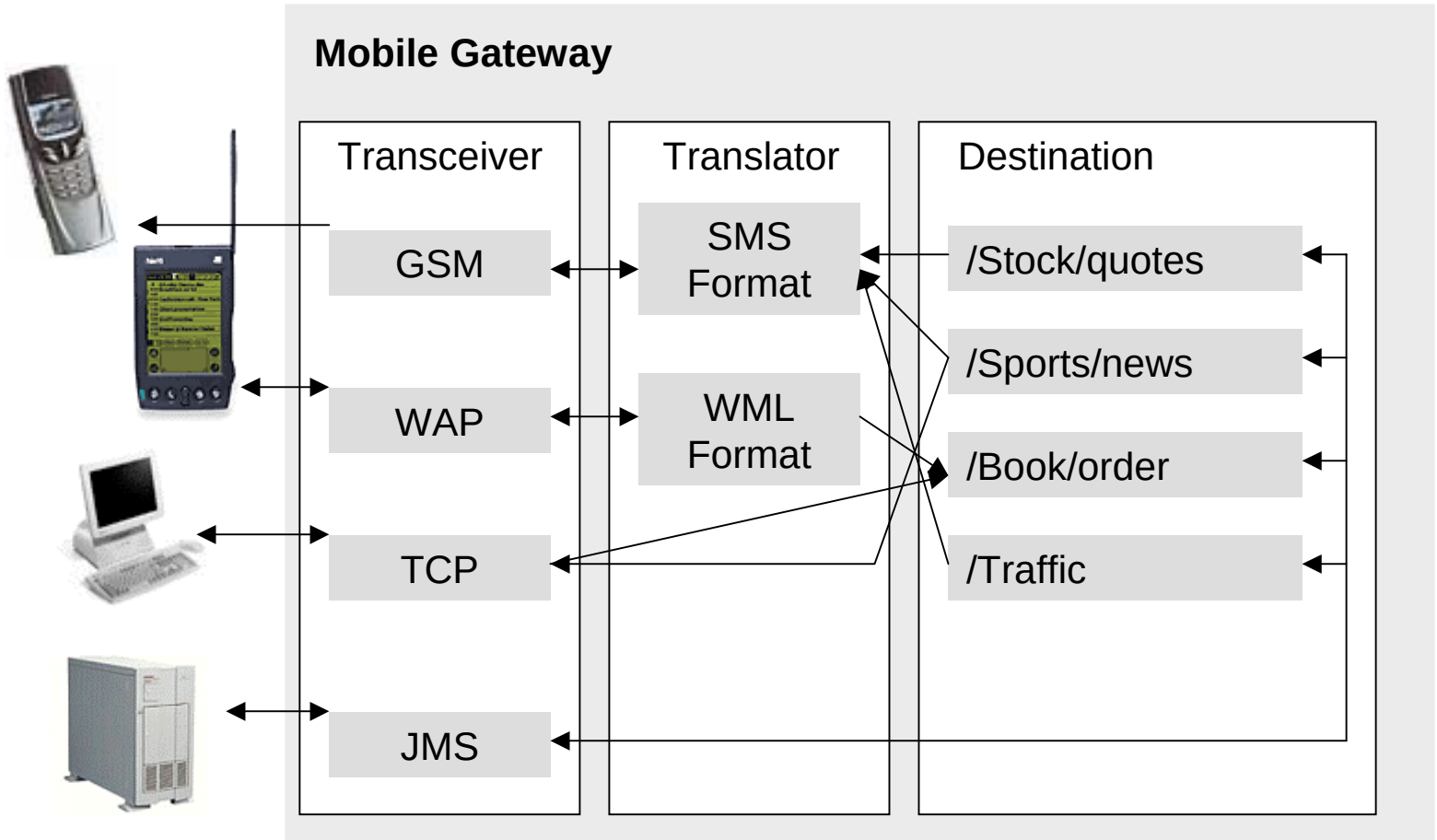
Mobile JMS

- Deployment
- Gateway Architecture
- Mobile Client Architecture

Architecture - Deployment

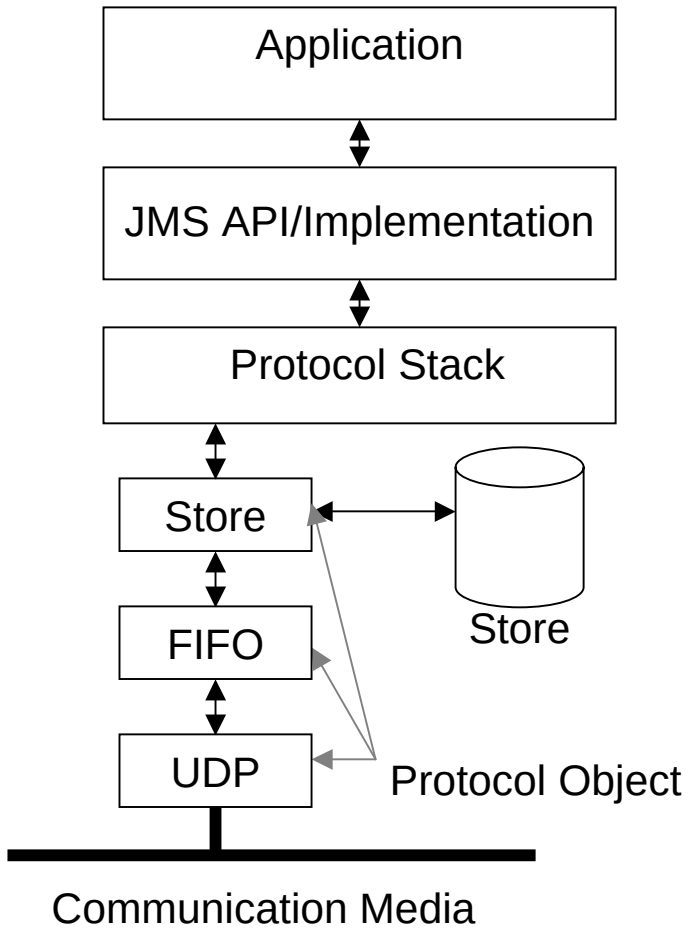


Architecture: Gateway



Architecture: Clients

- Application uses *regular JMS API*
- Protocol stacks are customizable
- Store enables *disrupted operation*
- Protocol Objects define transport and quality of service

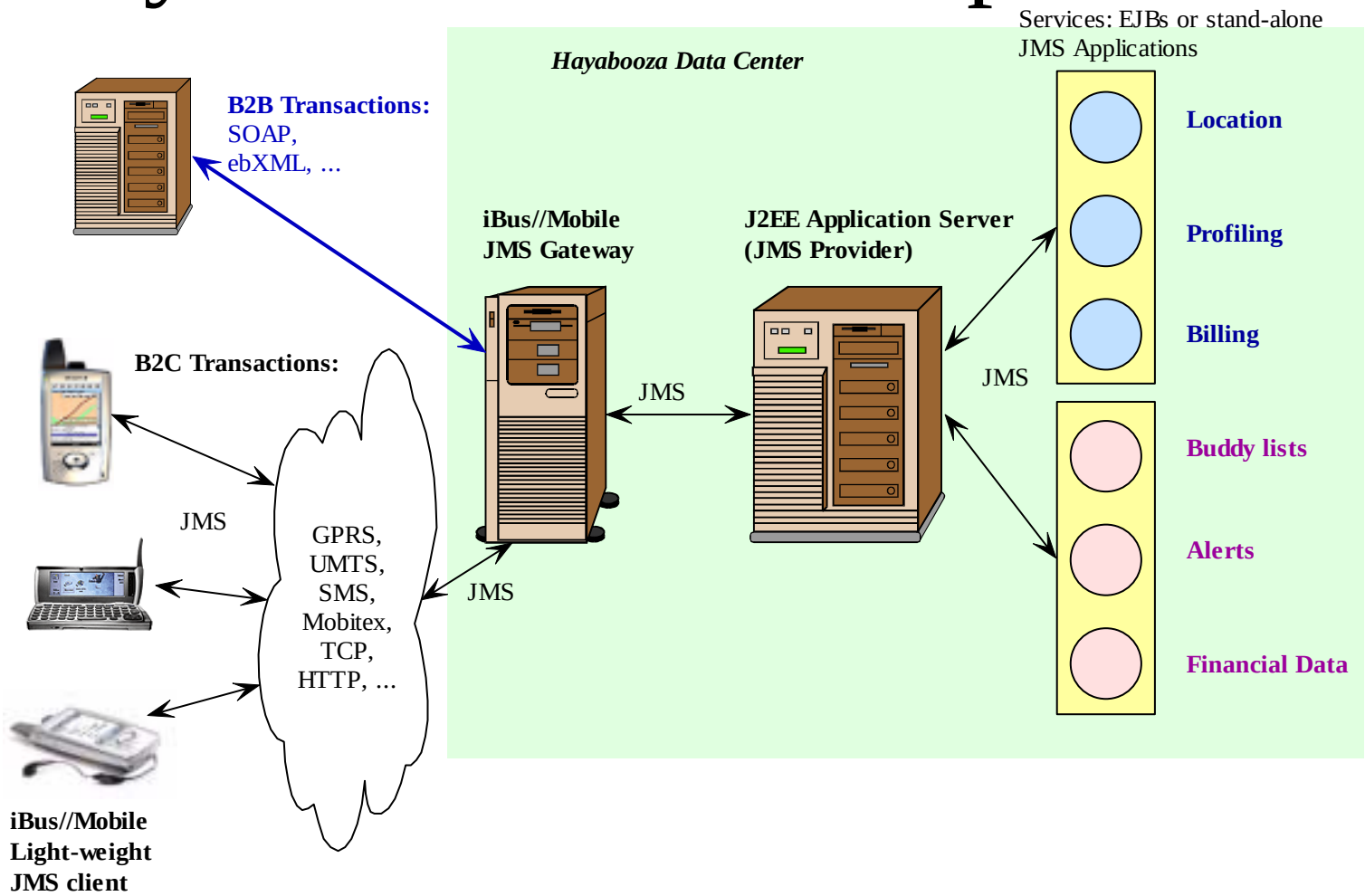


Code / Demo

- SimpleProducer.java

Beyond JMS: Hayabooza

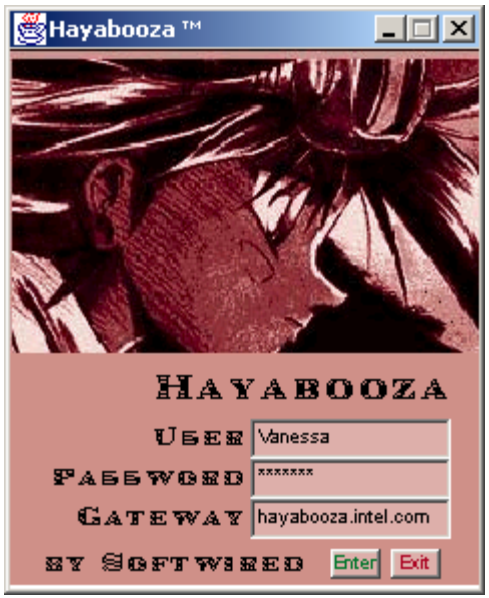
Hayabooza: iBus a step further



Hayabooza aspects

- Hayabooza = Wireless Services + Interactive Mobile GUI + iBus
- Platform-solution for „Third Generation Portals“ and WASP
- Common services: Location, Profiling, Accounting. Those are „integration wrappers“ for third-party offerings.
- Application Services: Financial data, gaming, mCommerce, groupware, location-aware ads, ...
- Extensible & scalable

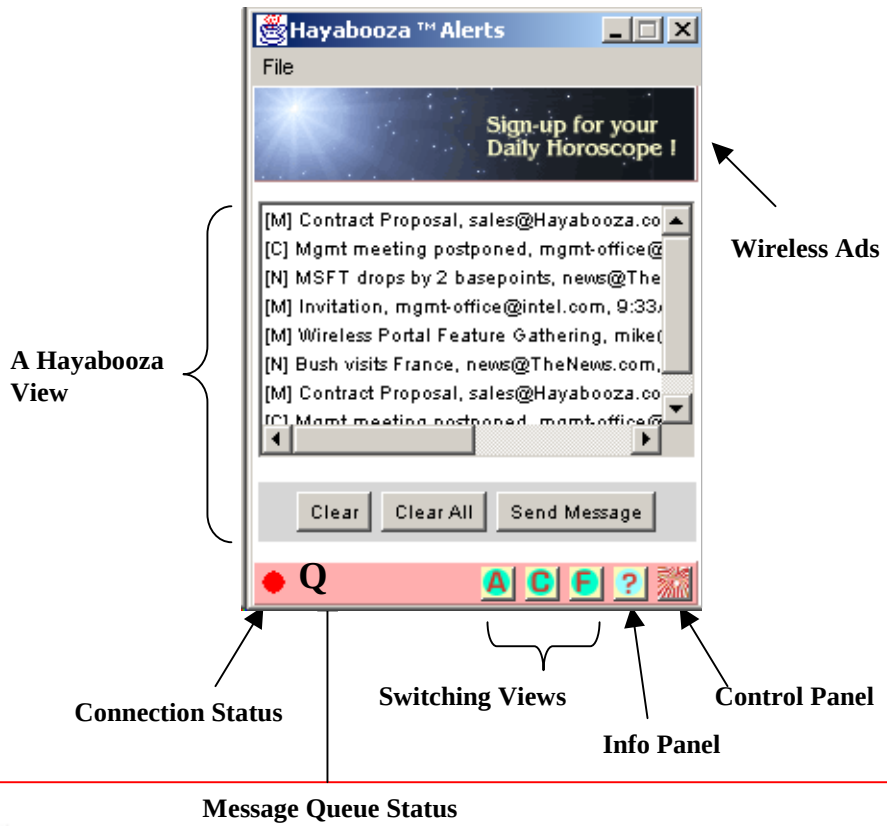
Hayabooza interactive client



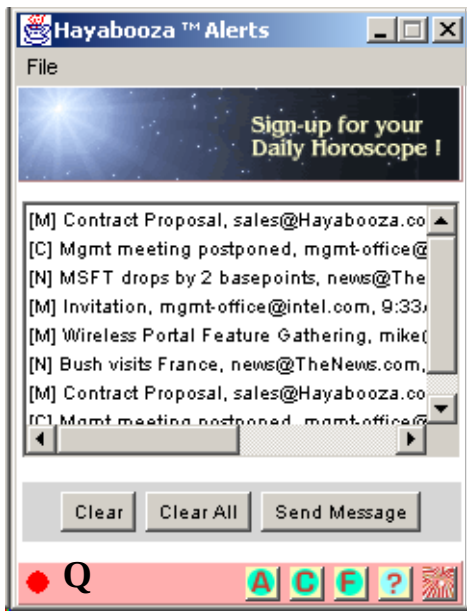
Hayabooza login screen

- Interactive GUI Java application
- Today; PersonalJava 1.1 (WinCE, Symbian)
- Tomorrow: J2ME (PalmOS, JavaPhone)
- Task-oriented user interface
- Emphasis on real-time interactions, and transactions

Hayabooza interactive client -- Concepts

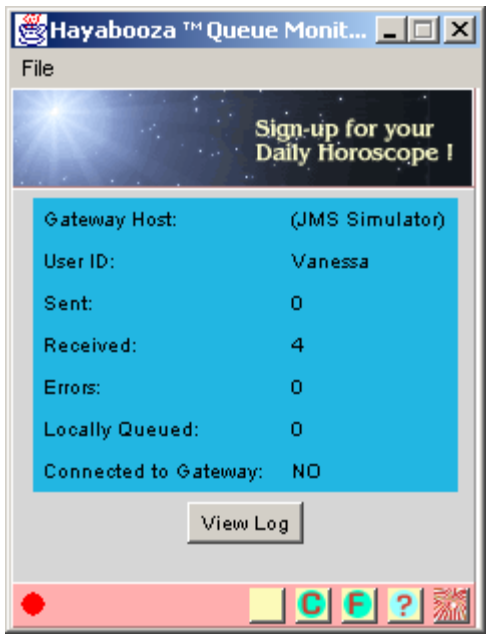


Hayabooza interactive client – Some „default“ tasks



Alert Notifications (e-mail, calendar, news)

Wireless Chat



Control Panel

DEMO

Conclusions

- For mobile application development, Java is a compelling choice
- For mobile application data exchange, JMS is a compelling choice. It might even be the only approach working in practice.
- While JMS transports data, more services (beyond J2EE) are required in a mobile environment: Billing, Profiling, Location, Security, …

Thank you!

More information:

Softwired AG
Martin Erzberger
Technoparkstrasse 1
8005 Zurich
Switzerland

martin.erzberger@softwired-inc.com
011 41 445 2370