Jabber/J2ME based mobile Instant Messaging client for Presence Service

Design, Implementation and Testing on Nokia 6600

University of Applied Sciences Stuttgart
Mai Kozakai

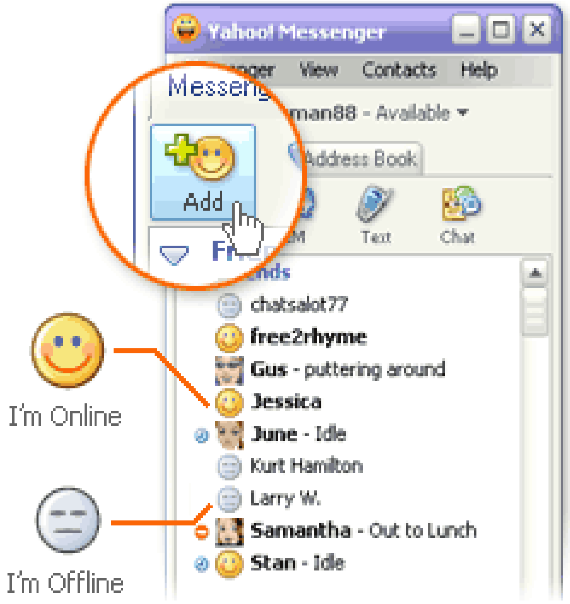
UNIVERSITY OF APPLIED SCIENCES
HOCHSCHULE DER MEDIEN

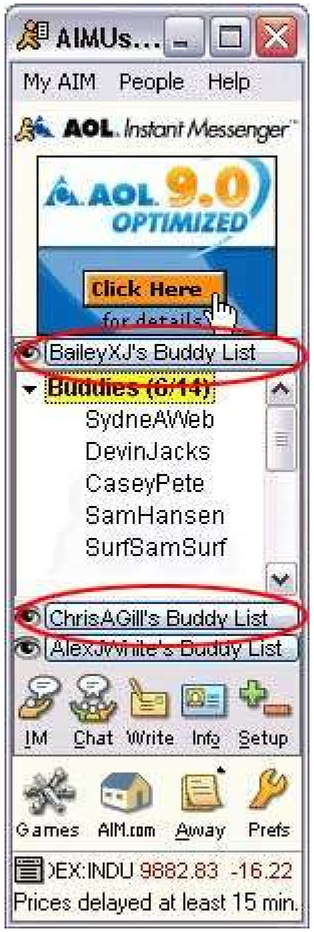
Contents Overview

- **Background / Motivation**
  - Instant Messaging (IM) and Presence
  - Concept of "*Presence Service*"
- **Applied Technologies**
  - Jabber Instant Messaging Protocol
  - J2ME: MIDlet programming
- **Application Development**
  - Requirement analysis & software design
  - Environment
  - Implementation of tough tasks / Demo
- **Conclusion**
  - Achievement & feature tendencies

Instant Messaging & Presence Feature

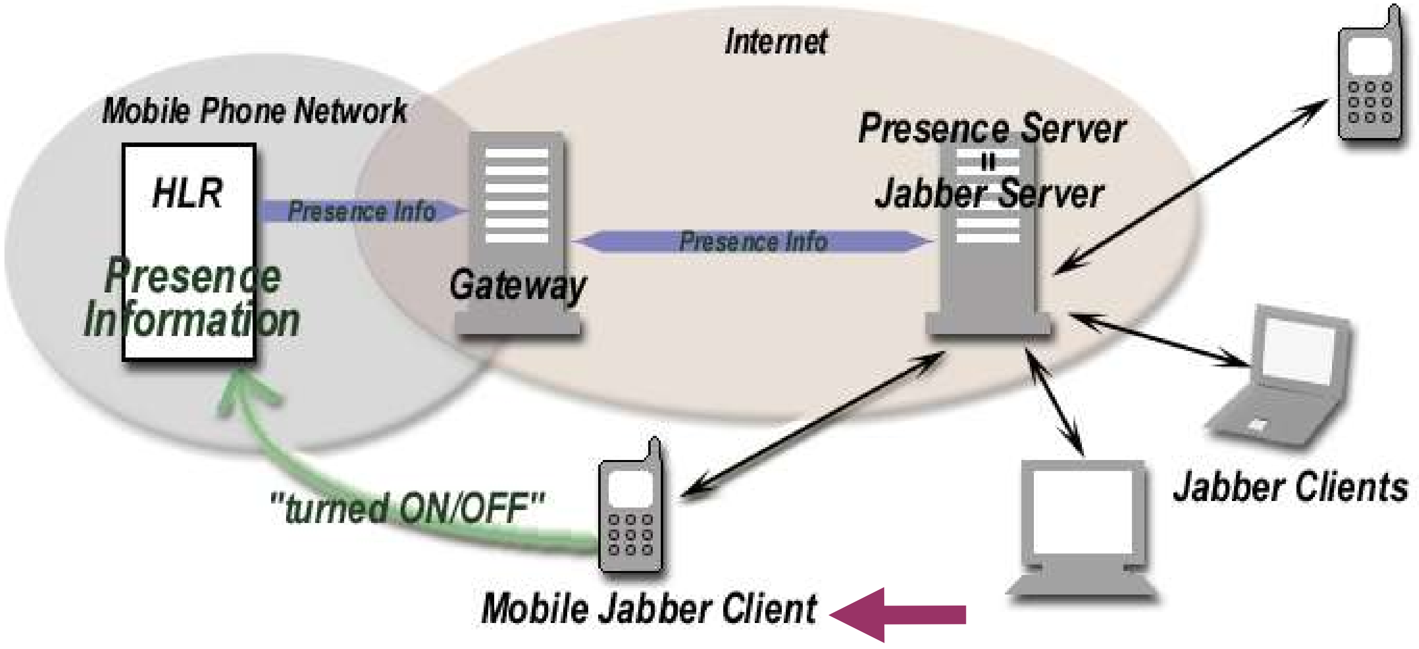
- **INSTANT**: communication in (almost) real time
  - 1-to-1 chat, conference, game
- **PRESENCE**: online availability
  - Boosting up...
    - Communication efficiency
    - Productivity
  - Bothering nobody
- Major IM products:
  - AIM, ICQ
  - MSN Messenger
  - Yahoo! Messenger
  - IBM Lotus Sametime





Mobile Presence Service

- Extending IM/Presence system to mobile phone
- Device's On/Off state → Presence: Online/Offline
- Leveraging Home Location Register (HLR)
- 24×7 presence awareness



Jabber Instant Messaging Protocol

- What is Jabber?
  - streaming **XML** protocols
    - exchange messages, presence and other structured information
    - similar to legacy IM (AIM, ICQ etc.) systems
- Why Jabber?
  - **free**, **open**, **public**, and **easily understandable**
  - **Standard** - Internet Engineering Task Force (IETF) approved the core protocols: **XMPP**
    - EXtensible Messaging and Presence Protocol
    - Proposed Standard, moving toward Draft Standard

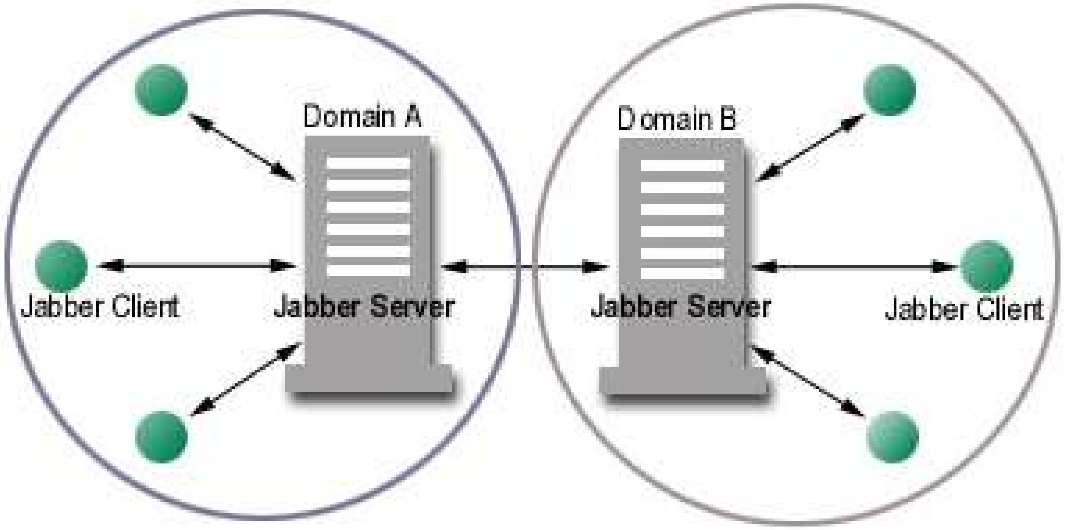
EXtensible Messaging and Presence Protocol

- **XMPP: Core**
  - approved 2004-01-29
- **XMPP: Instant Messaging and Presence**
  - approved 2004-02-05
- Jabber Enhancement Proposals (JEPs)
- Jabber supporter: Hitachi, HP, Jabber Inc. etc.

- Jabber = XMPP + JEPs

Jabber's Architecture

- Simple **client–server** architecture
  - TCP socket connection (port 5222)
- Decentralized, distributed servers
- Server manages Roster, Subscription, Blacklist
  - Enabling thin-client → ideal for mobile client ☺
  - Server as the single point of failure ☹

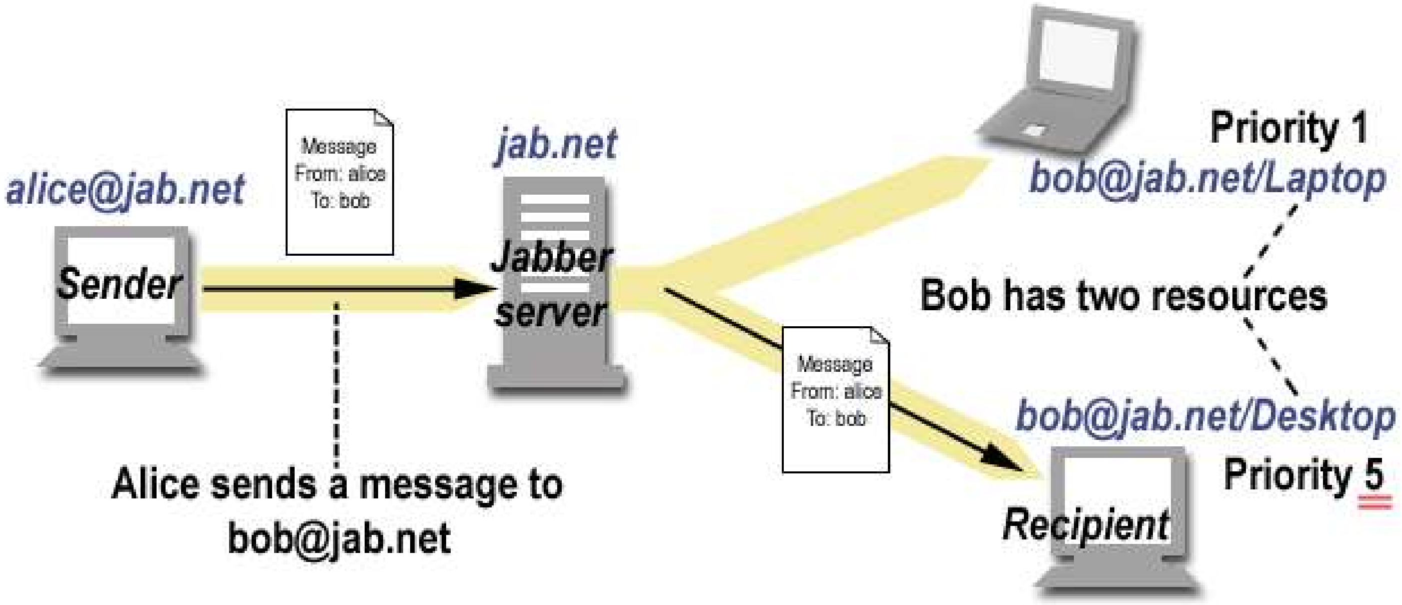


Jabber's Concept 1: Jabber ID (JID)

- All Entities can be addressed using Jabber ID
  - User, server, service on a server, conference room
- **username@hostname/resource**
- User's resource = location or device
  - romeo@jabber.org/work
  - juliet@jabber.net/home
  - juliet@jabber.net/mobile

Jabber's Concept 2: Priority

- **Resource Priority**
  - XMPP: Integer between -128 and +127, default:0
  - In the practice: positive integer
  - Higher number means higher priority



Jabber's Concept 3: Privacy

- **Presence Subscription System**
  - Disclose presence only to whom you have approved
  - Lasts until one of the entities cancel the subscription
- Blacklist function
  - Block communication to/from certain users
- Security
  - TLS for channel encryption
  - SASL (Stream Authentication and Security Layer Protocol) for stream authentication
  - Alternative to SASL: password digest authentication defined in JEP-0078

XMPP XML Stanzas

- **<message/>**

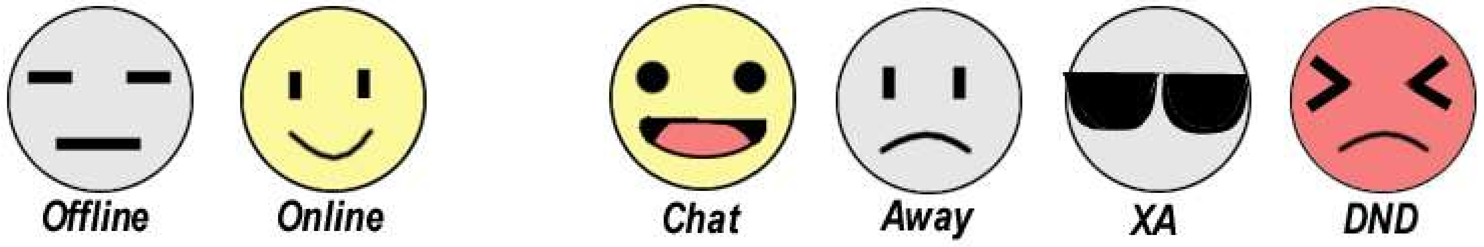
```
<message to='romeo@example.net'
   from='juliet@example.com/balcony'
   type='chat'>
      <body>Wherefore are thou, Romeo?</body>
     </message>
```

- **<presence/>**
  - Exchanging presence information
  - Subscription Management
- **<iq/>**
  - Info/Query – Request-Response mechanism
  - Roster, registration, authentication, blacklist

Example <presence/> Stanza

- Predefined: *chat*(ready to chat), *away*(absent for moment), *xa*(extended away), *dnd*(do not disturb)

```
<presence from='juliet@example.com/balcony'>
    <show>away</show>
    <status>went out for lunch</status>
    <priority>5</priority>
</presence>
```

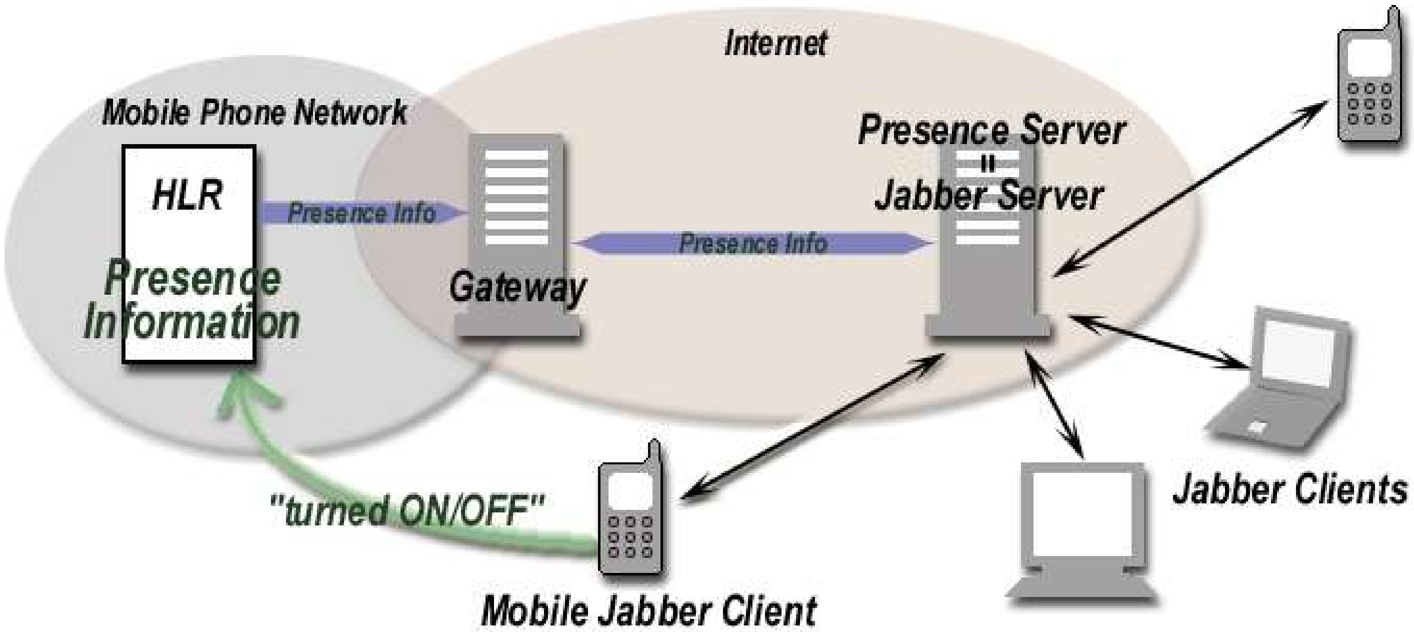


Restrictions on MIDlet Programming

- Very much limited number of classes/methods compared to J2SE
- Limited memory size on the device
  - Optimize code, use obfuscator
  - On exit: free connection & other variables explicitly
- Frequent interruption on runtime
  - Phone calls, SMS etc.
- Small display
  - divide long context into several screens
  - Make labels short & descriptive
- Sandbox model – no access beyond MIDlet Suite

Feasibility Check

- Server-initiated presence push after turning on
  - Use of **Push Registry**?
    - Nokia Series 60: only SMS & Bluetooth as incoming Push connection accepted
    - Use of SMS: define port to avoid confusion



Feasibility Check

- Jabber/XMPP requirement
  - Session establishment & authentication required before exchanging any XML stanza
    - Auto-login after receiving SMS push
- Presence service subscription management
  - Subscribe/unsubscribe or temporary stop of service
    - Out of Jabber's specification scope
- **Need to wait** for presence server specification...
  - Resetting goal: realization of XMPP conform Jabber client for mobile phones

Requirements

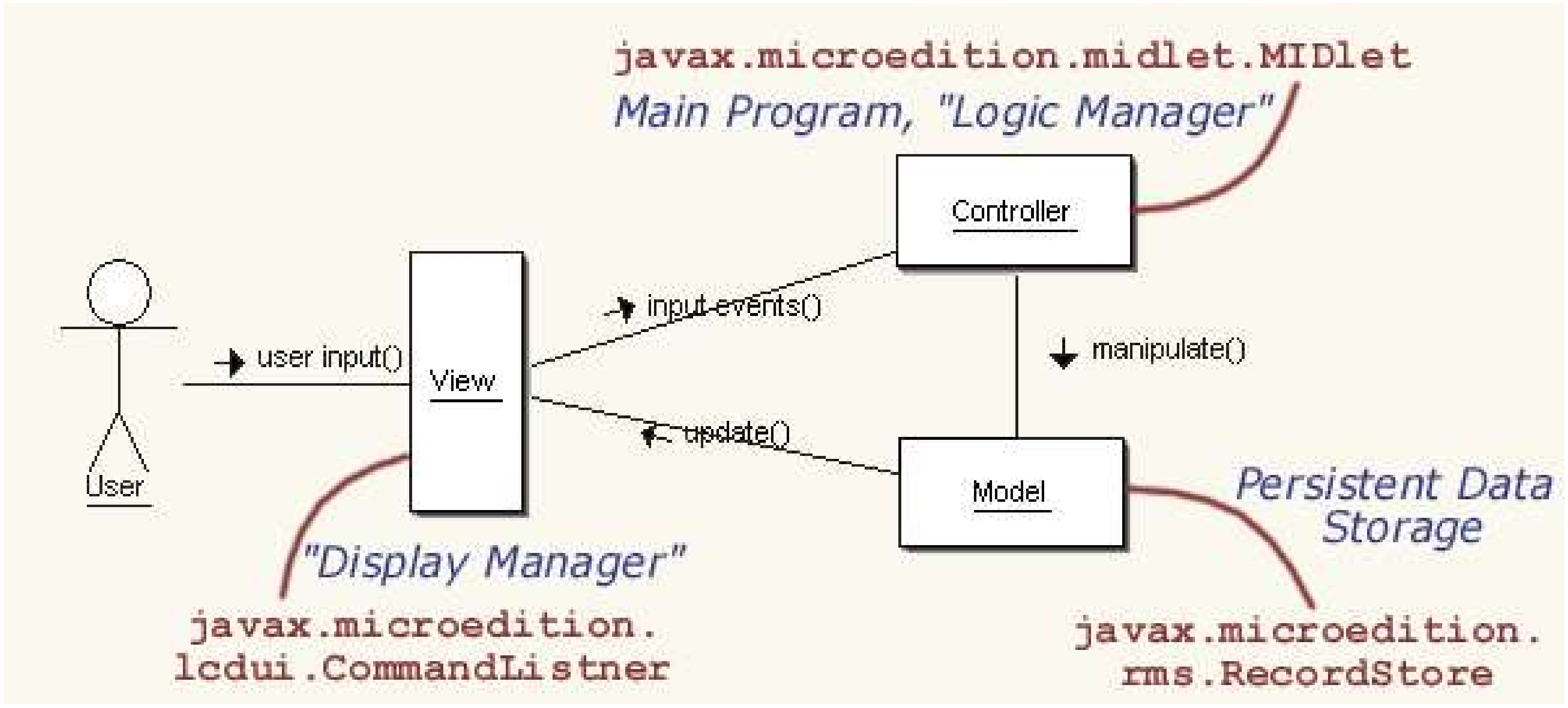
- XMPP Conformity
  - handle <message/>, <presence/>, <iq/> stanzas
    - Exchange messages with other users
    - Exchange presence information with other users
    - Manage subscriptions to/from other users
    - Manage items in a roster
    - Block communications to/from specific other users
- Auto-login option
- Small size
- User friendly, responsive GUI
- Preferably row memory consumption

Development Environment

- Wireless Toolkit 2.1
  - Build, create obfuscate JAR, emulate, debug
- Eclipse IDE + SIPtech J2ME plug-in
  - Alternative: Sun One Studio 4 Mobile Edition
- Target device – Nokia 6600
  - MIDP 2.0, GPRS, Memory heap size 3MB
- Series 60 MIDP Concept SDK Beta 0.3.1, Nokia Edition (emulator)
- XMPP server
  - jabber.org
  - Antepo OPN server
- Exodus Jabber client for PC

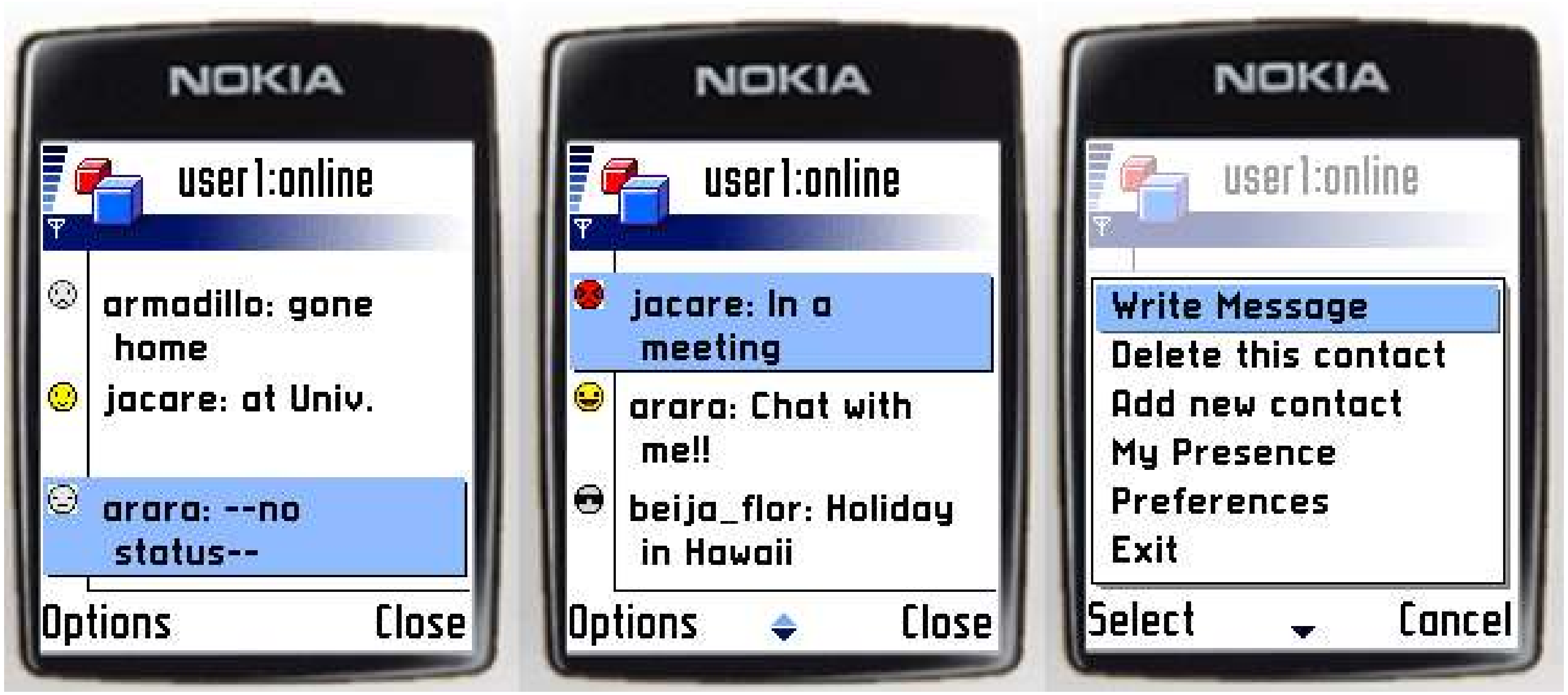
Software Design Pattern

- MVC: Model-View-Controller Pattern


javax.microedition.midlet.MIDlet
Main Program, "Logic Manager"
Controller
User
user input()
View
input-events()
manipulate()
update()
Model
Persistent Data Storage
"Display Manager"
javax.microedition.lcdui.CommandListner
javax.microedition.rms.RecordStore

Program Flow

- Roster as the main screen
- Status icon – name – status description

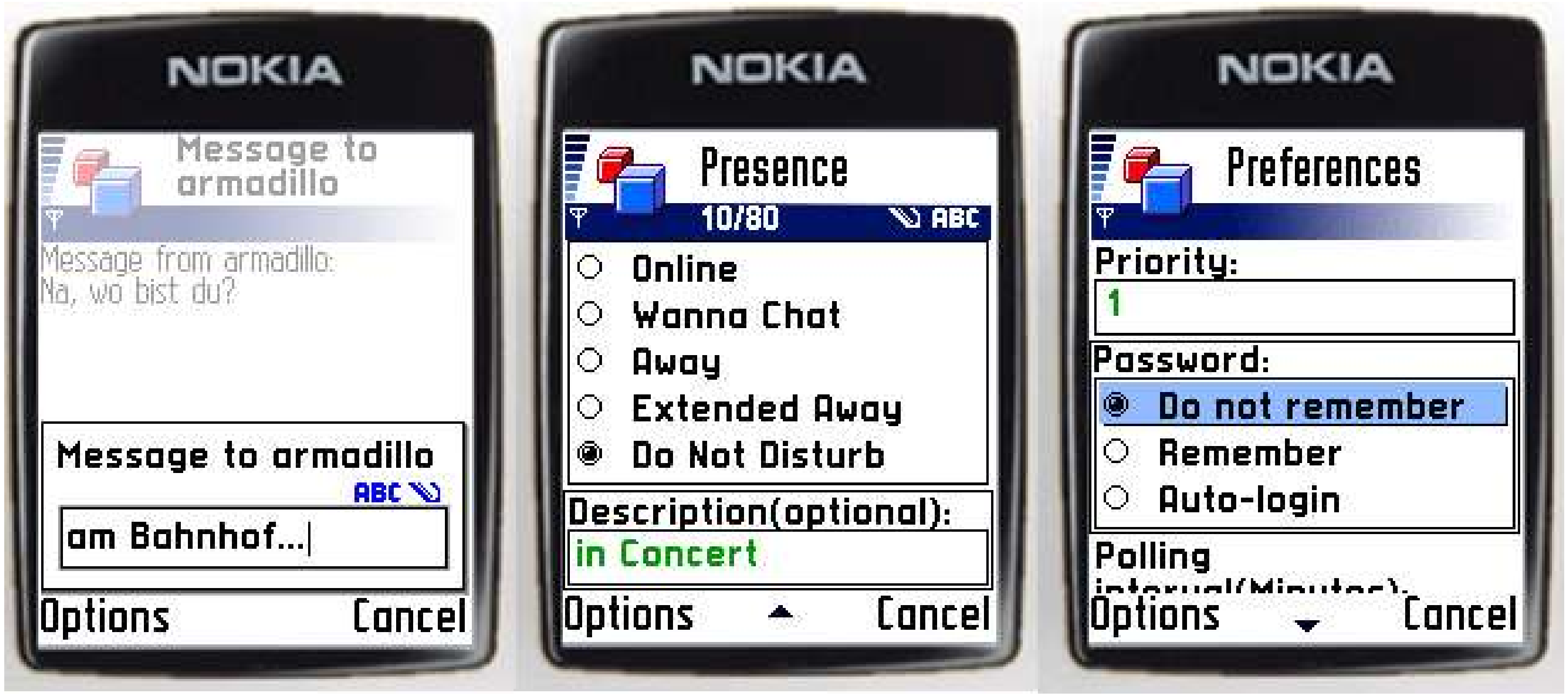


armadillo: gone home
jacare: at Univ.
arara: --no status--

jacare: In a meeting
arara: Chat with me!!
beija_flor: Holiday in Hawaii

Program Flow

- Presence setting screen: status, free-text description
- Preference screen: priority, remember password, blacklist function



Message from armadillo:
Na, wo bist du?

Message to armadillo
ABC
am Bahnhof...

Online
Wanna Chat
Away
Extended Away
[✓ Do Not Disturb]

Description(optional):
in Concert

Priority:
1

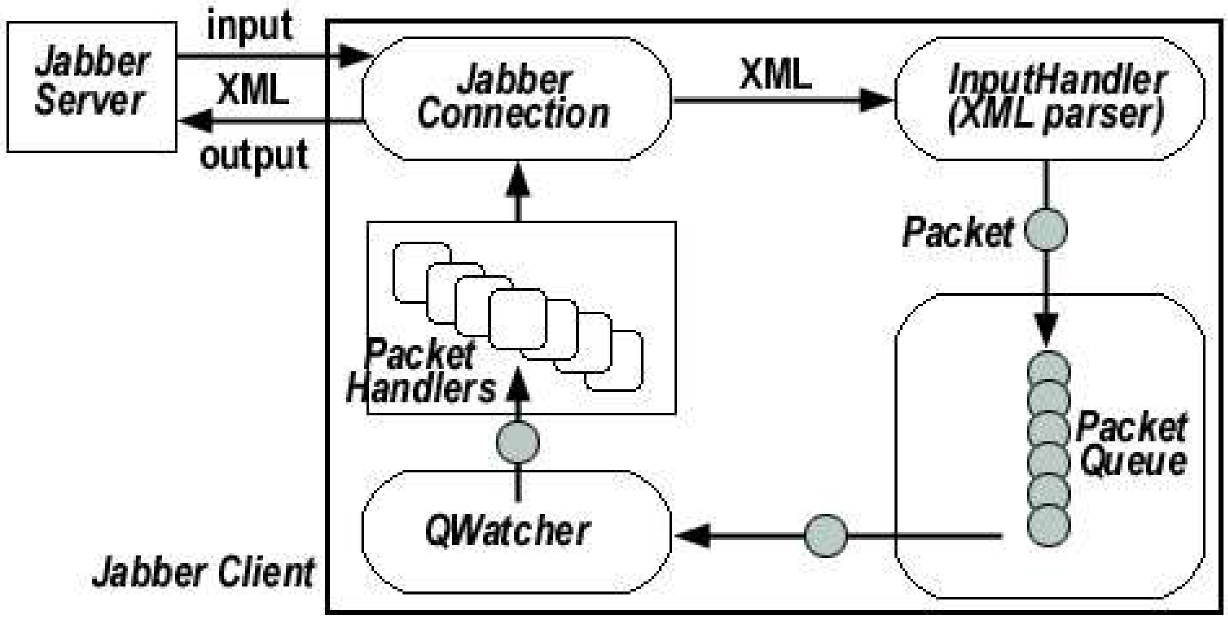
Password:
[✓ Do not remember]
Remember
Auto-login

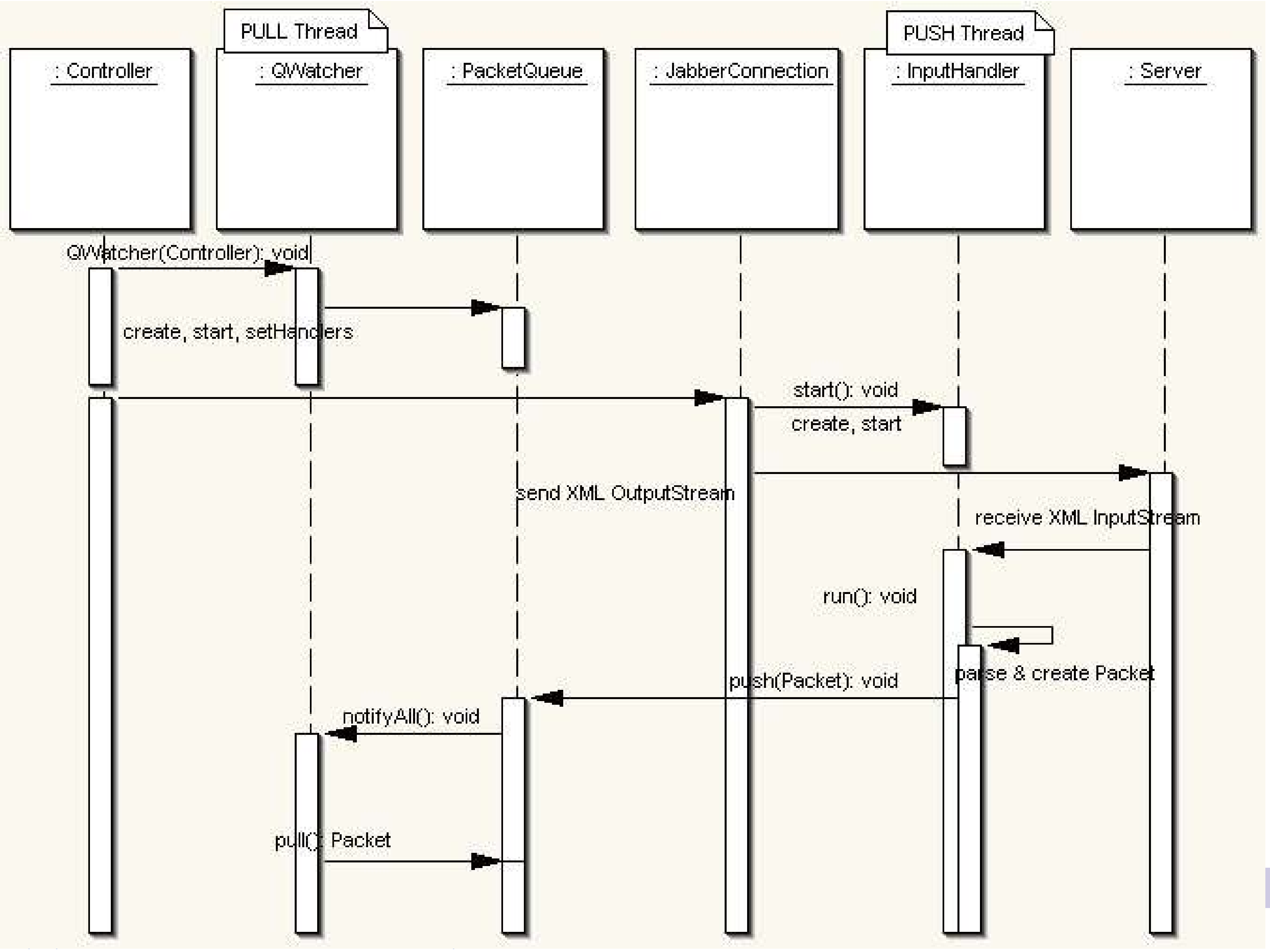
Polling
interval(Minutes):

Background | Technology | **Application Development** | Conclusion

Tough Tasks I: XML Parsing

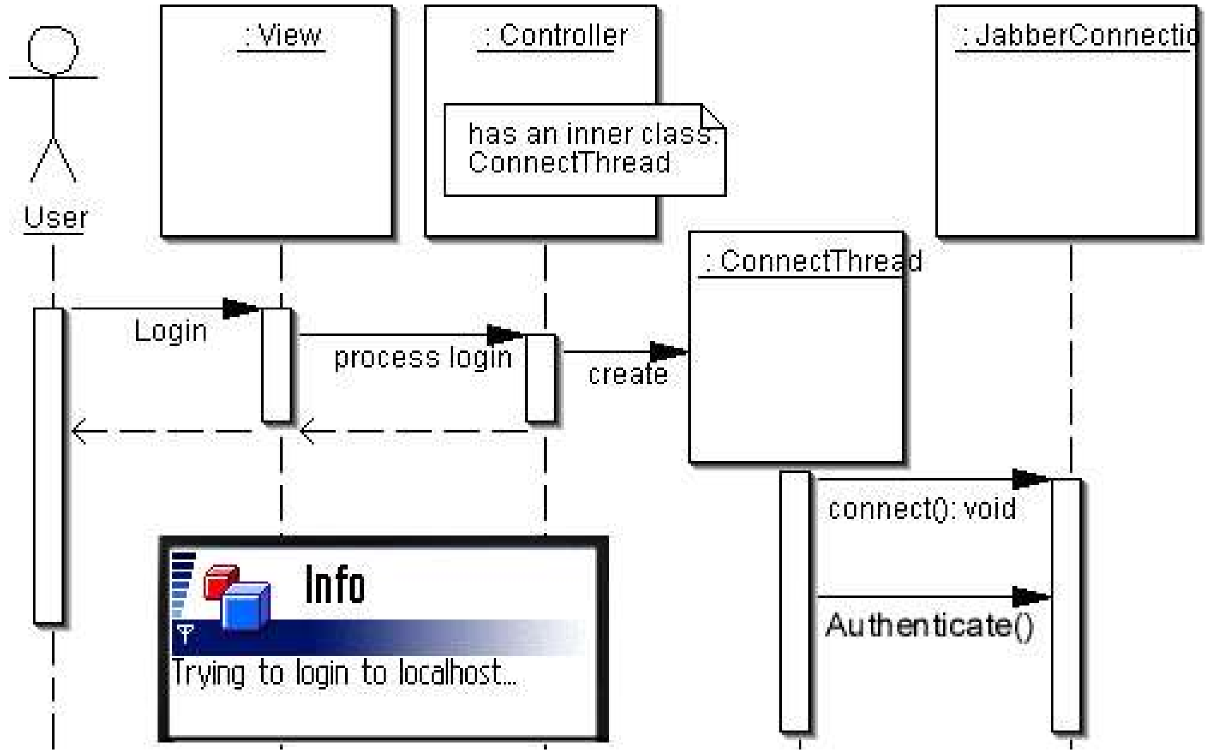
- Programming with J2ME limitations
- XML parsing subsystem
  - Examination of third-party APIs (NanoXML, TinyXML, kXML)
  - DOM Node-like "Packet" creation




PULL Thread
PUSH Thread
: Controller
: QWatcher
: PacketQueue
: JabberConnection
: InputHandler
: Server
QWatcher(Controller): void
create, start, setHandlers
start(): void
create, start
send XML OutputStream
receive XML InputStream
run(): void
parse & create Packet
push(Packet): void
notifyAll(): void
pull(): Packet

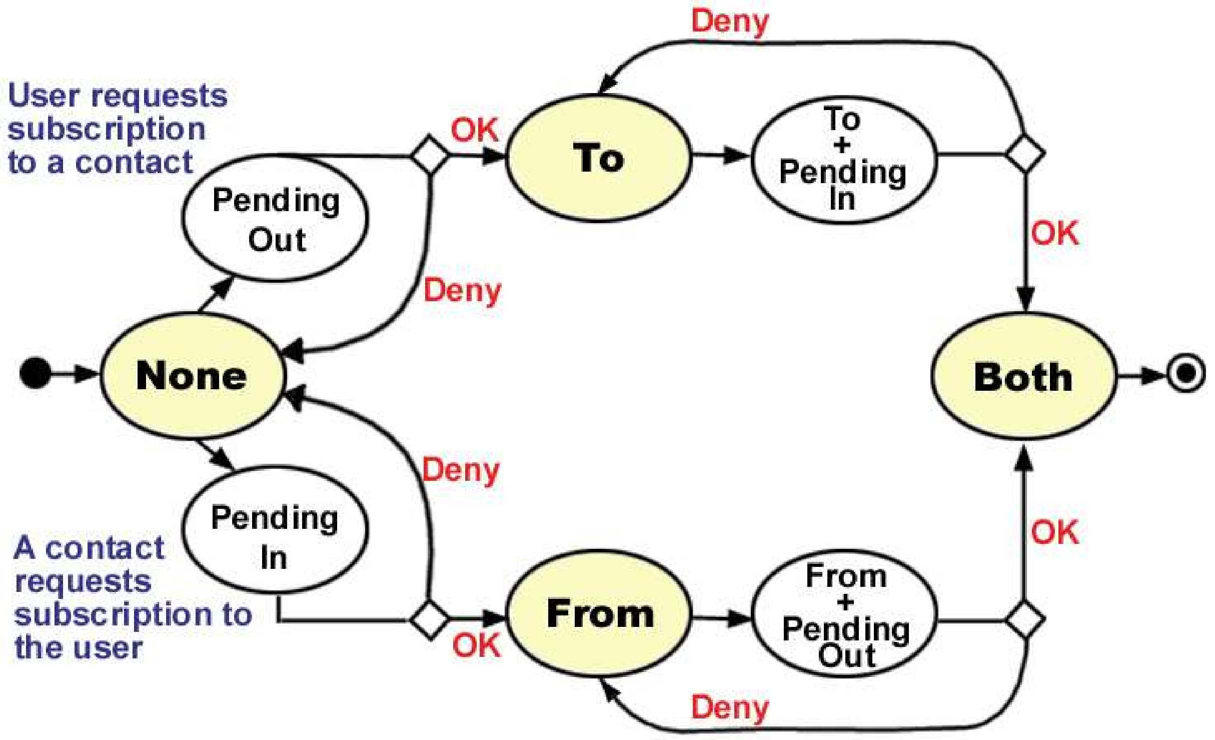
Tough Tasks II: Multi Threading

- Reduced use of keyword *synchronized*
- Start network operation on separate Thread
  - Essential to make UI responsive



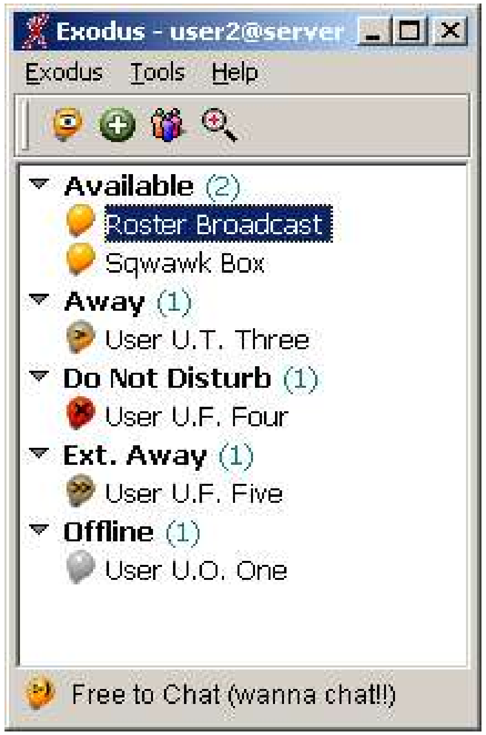
Tough Tasks III: Presence Subscription

- Presence subscription management
  - Transition: “none” → [“to” || “from”] → “both”
  - If “from”, try to subscribe that contact to create *mutual* subscription



Conclusion

- Achieved
  - All the basic IM functionality defined in XMPP
  - Blacklist
  - Password digest authentication (defined in JEP-0078)
  - Auto-login option
  - Configurable roster update interval
- Yet to be improved
  - Use of TLS, possibly SASL
  - User Interface
    - Line-by-line chat
    - Sort roster items (status / group)
    - More color (avatar)?



Available (2)
Roster Broadcast
Sqwawk Box
Away (1)
User U.T. Three
Do Not Disturb (1)
User U.F. Four
Ext. Away (1)
User U.F. Five
Offline (1)
User U.O. One
Free to Chat (wanna chat!)

Comparison to Existing Clients

- TipicME (released in April 2004)
  - Avatars, navigation through colorful icons
  - Limited to Tipic-Server
- mJabber (master thesis written in Jan. 2003)
  - MVC pattern, DOM Node-like parsing system
  - IMpulse supports more of XMPP ☺
    - All presence status: on/off // chat/away/xa/dnd
    - Free text status description
    - Password digest authentication
    - Blacklist support
    - Supporting roster item's <group/>
    - Handling of subscription “from” state

Future IM Tendency: Standardization

- Jabber/XMPP
  - Approved in May 2004: “Mapping XMPP to CPIM”
    - Other JEPs will follow IETF standardization process
- SIP/SIMPLE (Session Initiation Protocol for IM & Presence Leveraging Extensions)
  - Getting toward IETF “Proposed Standard”
    - Supporters: IBM, Microsoft, Sun, Novel
- JAIN (Java API for Integrated Networks)
  - JAIN Presence (JSR 186)
  - JAIN Instant Messaging (JSR 187)
    - Standard API for IM/Presence Application
    - Interoperable across different underlying protocols

Q and A

Thank you for your attention!

☺

Any Question?