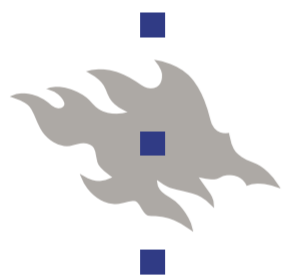


Jaiku: Jabber the Mobile

Mika Raento, mikie@iki.fi

Jaiku Ltd

Department of Computer Science, University of Helsinki
Basic Research Unit, Helsinki Institute for Information Technology HIIT



HELSINGIN YLIOPISTO

www.cs.helsinki.fi/group/context/



www.hiit.fi/



jaiku.com/

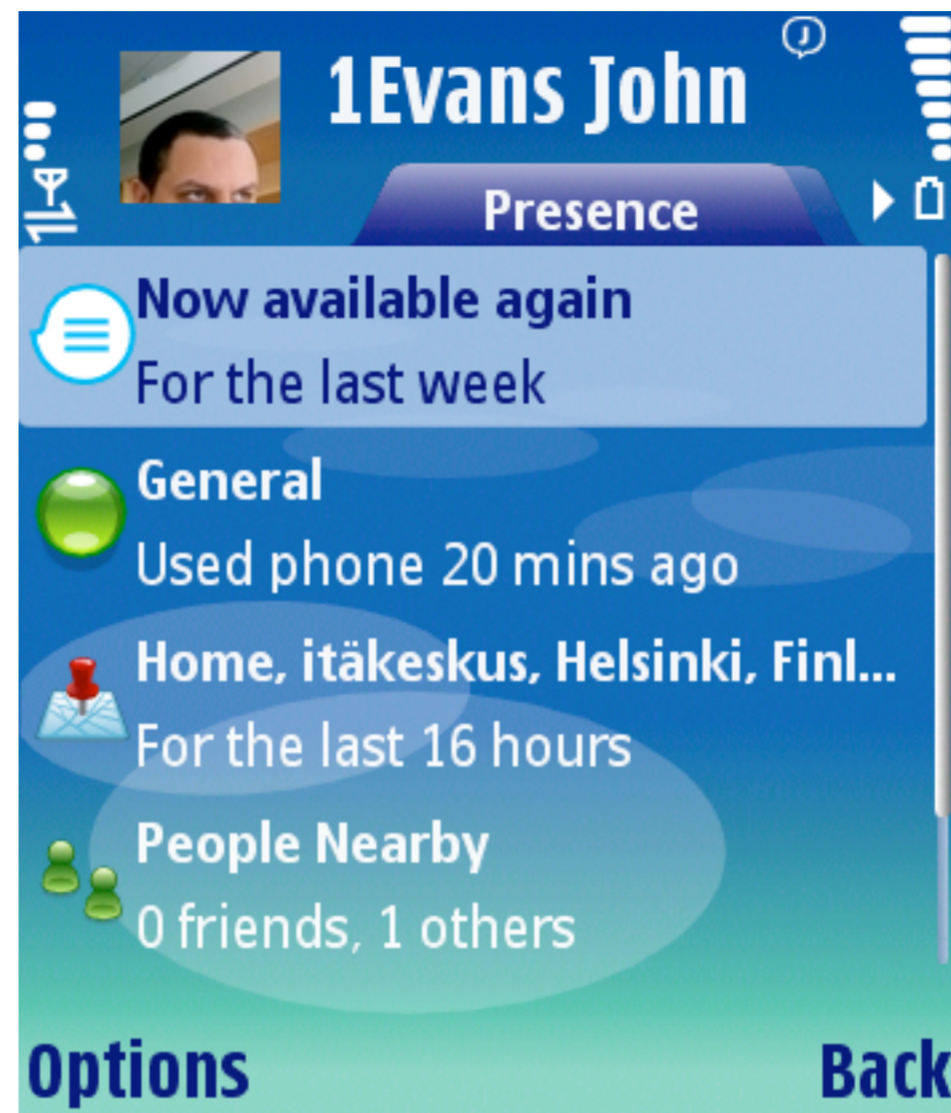
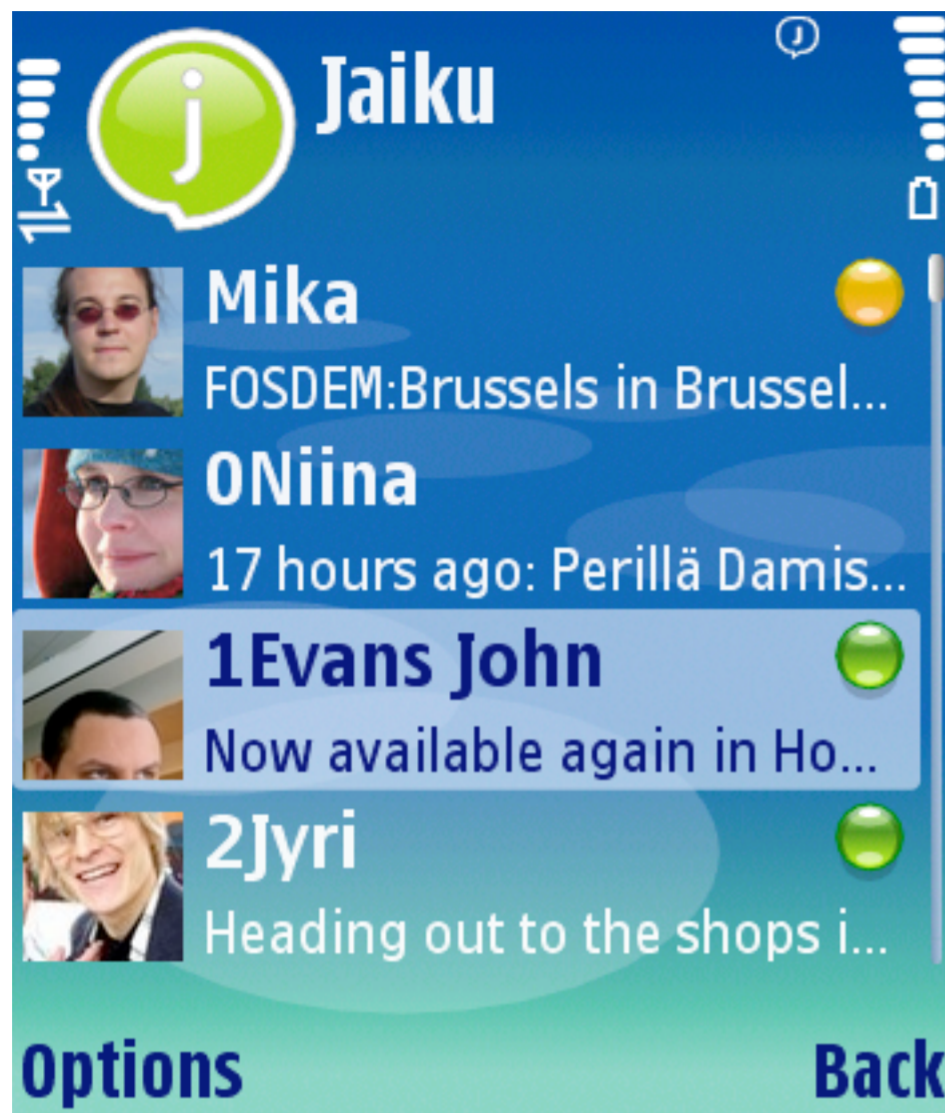
This talk

- What Jaiku is about: ubiquitous social awareness
- How people use phones: ‘interaction in 4-second bursts’
- Connectivity on a phone: pervasive, flaky, battery-hungry
- What challenges these pose to XMPP



The aim is to talk about interesting and hopefully non-obvious aspects of mobile phone usage and connectivity as they relate to protocol (rather than implementation) design

What we are doing



jaiku.com/

Ubiquitous social awareness
runs on top of XMPP, but hacks rather than JEPs
Nokia S60, web, Java client, SMS interface
Research version GPL, current version not available (yet)
Communication pattern: people may be interested in what you are doing even if you are not motivated to say anything – and you may have a use for seeing what somebody is up to even if you don't realize it: peripheral, always-on
Intimate, group-oriented, companionship
Interacting with the big blue room: capturing that experience, but also informationalizing some of it
The data is about what is happening right `_now_`: something I might want to react to. I need to know whether it's up-to-date.
Undemanding messages: you can just post things without requiring attention from the receiver (a social construct triggered by some of the qualities of the tech)

Interaction in 4-second bursts



jaiku.com/

Even if a person is carrying out a task, they continuously switch their attention to the external world

Desktop interaction models may map poorly to mobile phones

On a desktop, the user may also multi-task, but they have control over: they can, if they want, focus on a single task, and they can gracefully switch out of a task

A person walking on the street is forced to switch their attention, maybe abruptly, so that they can cross streets, avoid cars

Voice calls take advantage of our natural abilities (separate audio thread), SMS are a good example of something that requires the use of general cognitive facilities and the resulting attention switching

Applications must have response times that are acceptable for the task

Early 2000s computer, shame about the user IO



400 MIPS processor
general purpose OS
40 MB RAM
2 GB disk
384 kbps networking



jaiku.com/

But small screen and slow text input
Normal one-to-one desktop IM data rate completely overwhelms a mobile user
But media instead! Pictures of the world the user moves within.

TCP/IP over 3G: the irritants

- Bringing an interface up can take a minute
- First packet out of idle takes at least 5 seconds
- Bandwidth is not a problem for messaging, does define interactional limits for media
- Network is always available in civilized countries
- But the connection is continuously dropped or suspended by calls, 3G shadows
- The GPRS gateway buffers data and acknowledges packets when the terminal is unavailable



The technology is not there yet

In the UK, you lose signal in an elevator or in a pub

In Finland ping-pong between 2G and 3G cells can in effect disable the connection

The existence of the TCP connection only means that the gateway is still happy to buffer your data

TCP/IP over 3G: the catastrophes

- GSM signaling is extremely well optimized: 10 days stand-by time
- A phone that is transmitting or receiving data over 3G requires two orders of magnitude more power than when idle: 1 day of stand-by time with a TCP connection +heartbeat
- 2G is much better, but politically infeasible (three times the battery life with same data transfer)
- 3G assumes everything is web browser: tens of KBs of data every 30 seconds



The technology is not there yet

It's not about bandwidth, it's about battery life

Ping-pong between 2G and 3G cells can kill the battery in 40 minutes

Several failure modes that require a reboot of the device (name resolver, connection between the PDA and the GSM side, transient errors sometimes considered permanent with no way to reset the state)

The phone communication innovation space will explode when always-on IP starts to work (think if you had to open a separate app and poll to see whether you've got an incoming call). Get ready.

XMPP, then

- Disclaimers
- ‘Hello. My name is Mika and I misuse presence.’
- We did not have resources to build a server: had to live with what was available, but could do a front-end proxy that did some transformations
- Designed in 2003--2004
- Initially a research prototype for field experiments: did not care about interoperability
- We have hacked these extensions, but would be willing to look into starting to use standardized versions or standardizing things that aren't yet



TCP connection <> availability

- The TCP connection is not under (conscious) control of the user
- Connection breaks often, mostly transiently
- Presence should not change based on transient connectivity breaks (false alarms, unnecessary traffic)
- But we do normally want to maintain a connection because of latency
- If the phone is idle, updates are not shown and should not be sent by the server
- Active/idle much more useful



We don't create 'unavailable' presence from connection drops

The client suspends the server connection when the phone is idle, resumes when the user touches a key

Pace of messages varies, but tends to always be slower than on the desktop

We don't want to poll due to large latency, rather keep a connection open – also more traffic than notifications

Presence is timestamped, and receiver can see how up-to-date it is and make inferences based on that

Reliable messaging

- Since attention is switched off the mobile both by sender and receiver, the normal human level ack/retry doesn't work
- Transient connection breaks are much, much more frequent than on the desktop, but a connection may also stay down for days (trip abroad)
- Some high-value messages ('I love you', photos)
- Very much XEP-0079 (AMP) like, but implemented like XEP-0198 (stanza acking) with fixed rules about expiry per message type
- No presence-probes: the server knows what updates have happened since the client last received some



Desktop IM is very often semi-synchronous: you are willing to wait whether the other person answers, and possibly fall back to some other media if they don't

Store-and-forward rather than real-time message routing, but with the goal of having low enough latency to support chat

Can't afford to send the presence of 100 people every time a connection is re-established

All clients of the user receive the same data by default

Structured messages and other data

- Structured presence (gsm-cell, phone ring volume), buddy images, Jaiku interal communication
- No standards for many things yet
- Important for rich interaction on the mobile client (send an ISBN for a book I'm reading, automatically store mappings between cell-id and location name, presentation rules for phone profile data)
- We just put our own stanzas in the jabber stream now
- We will most likely keep doing this for our own clients, but would be nice to degrade gracefully
- Potential ways: mime-types, inline mime-typed data with alternative representations, URLs, extra URI schemes, pushing presentation layer data to clients (html for presence?)
- Ideas?



Standardization is one way

The other way is like flickr: provide decent documentation and let people integrate the things they find compelling

Fast-moving target (new devices come out with new capabilities and sensors, we want to change schema, new kinds of messages added to Jaiku)

Feedback to the user

- Since delivery is expensive, we postpone it until the user is there
- So there may be quite a bit to deliver
- Bandwidth is still lowish, and variable
- The user needs to be notified of how much data there is to come
- Protocol-level progress indicator: coming out of suspend, the server tells the client how many stanzas it has queued. Client shows a progress bar based on that and reception of the stanzas.

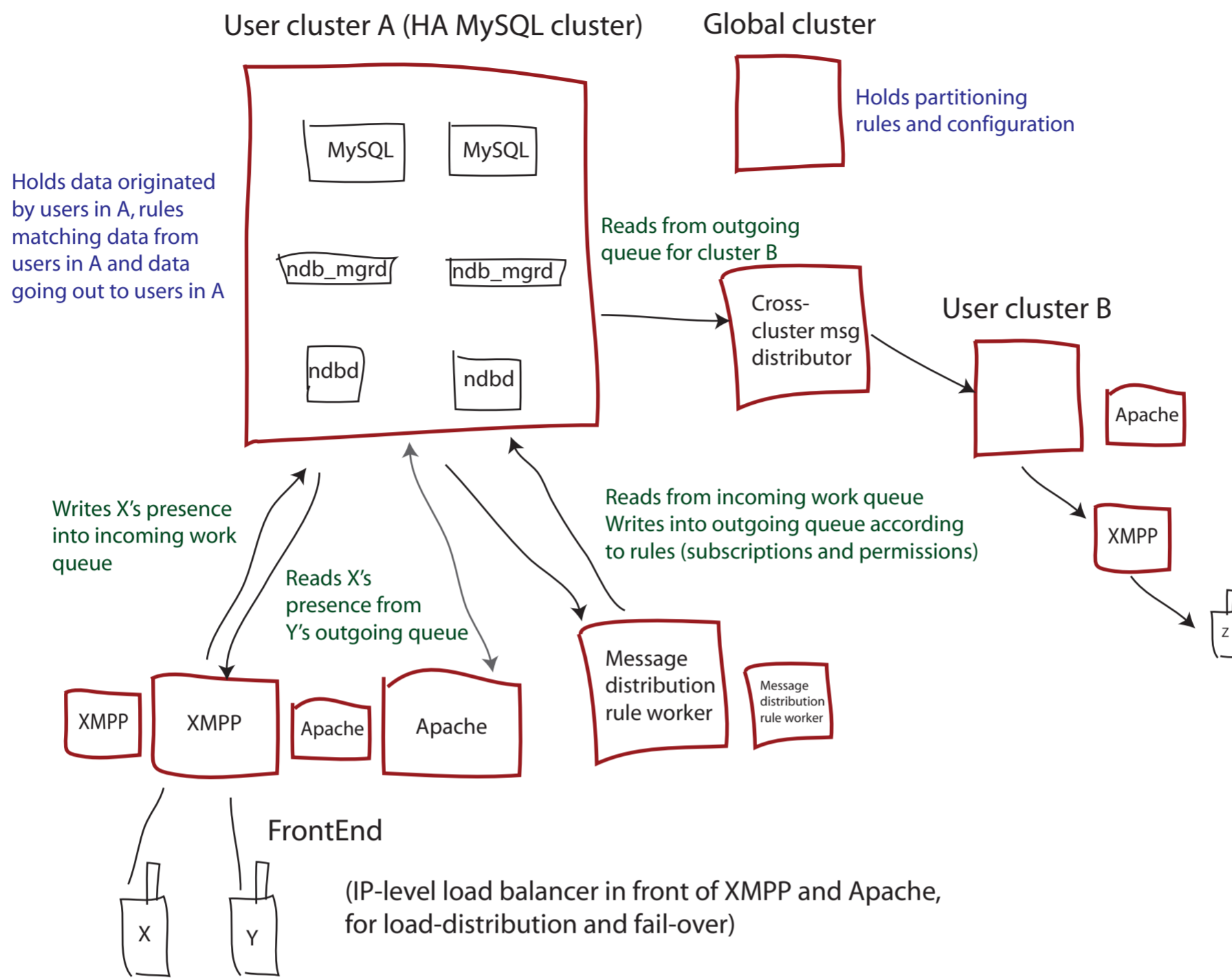


Why would you care?

- Anybody who wants to bring presence and IM to the mobile will have to solve at least a significant subset of these things
- There are some really cool applications to be done with mobile phones and reliable structured messaging
- Jaiku is a company, but we are quite willing to play nicely with you guys
- We will be providing (XMPP-based) hooks into the system both at the network and in the terminal, if you want to play with rich presence, context-aware applications, emotive software and intimate relationships
- Talk to me, Ralph and Andy about what you think you could do with these capabilities



Jaiku service architecture - the Plan



The only real idea: all hops are reliable store-and-forward. The client 'never' deletes anything before the server acks it. The server never acks anything before committing it to durable storage. Routing is done from durable storage, so failures of routing don't break the client-server communication.

Feedback to the user will be crucial.