TCP, HTTP and Java over Wireless Links

Kimmo Raatikainen

kimmo.raatikainen@cs.helsinki.fi

Lessions Outline

- Problems in TCP Over Wireless
- Additional Problems Due to HTTP
- Java RMI Over Wireless

Problems in TCP Over Wireless

Problems in TCP/IP over Wireless Link - 1

- Overhead due to protocol headers
  - TCP headers take c. 60 bytes
- High latency
  - “extra” round trips should be avoided
- TCP slow start
  - full bandwidth not utilized
- Timers will not work as intended
  - If packet delivery times vary, then TCP timers get confused

Problems in TCP/IP over Wireless Link - 2

- Inefficient recovery from packet losses
- Simultaneous TCP connection interfere with each other
- No support for disconnected state

Should TCP be used – facts against

- It is generally recognized that TCP does not perform well in the presence of significant levels of non-congestion loss.
  - TCP detractors argue that the wireless medium is one such case, and that it is hard enough to fix TCP. They argue that it is easier to start from scratch.
- TCP has too much header overhead.
- By the time the mechanisms are in place to fix it, TCP is very heavy, and ill-suited for use by lightweight, portable devices.

Should TCP be used – facts in favior – 1/2

- It is preferable to continue using the same protocol that the rest of the Internet uses for compatibility reasons.
  - Any extensions specific to the wireless link may be negotiated.
- Legacy mechanisms may be reused (for example three-way handshake).
- Link-layer FEC and ARQ can reduce the BER such that any losses TCP does see are, in fact, caused by congestion (or a sustained interruption of link connectivity).
  - Modern W-WAN technologies do this (CDPD, US-TDMA, CDMA, GSM), thus improving TCP throughput.

Should TCP be used – facts in favior – 2/2

- Given TCP's wealth of research and experience, alternative protocols are relatively immature, and the full implications of their widespread deployment not clearly understood.
- Handoffs among different technologies are made possible by Mobile IP [RFC2002], but only if the same protocols, namely TCP/IP, are used throughout.

What to Improve - 1/2

- TCP: Current Mechanisms
  - Slow Start and Congestion Avoidance
  - Fast Retransmit and Fast Recovery
- Connection Setup with T/TCP [RFC1397, RFC1644]
- Slow Start Proposals
  - Larger Initial Window
  - Growing the Window during Slow Start
    - ACK Counting
    - ACK-every-segment
  - Terminating Slow Start
  - Generating ACKs during Slow Start
- ACK Spacing

What to Improve - 2/2

- Delayed Duplicate Acknowlegements
- Selective Acknowledgements [RFC2018]
- Detecting Corruption Loss
  - Without Explicit Notification
  - With Explicit Notifications
- Active Queue Management
- Scheduling Algorithms
- Split TCP and Performance-Enhancing Proxies (PEPs)
- Header Compression Alternatives
- Payload Compression
- TCP Control Block Interdependence

Future Readings

- RFC 2757: Long Thin Networks
- Other IETF PILC WG RFCs and Ids
- Particularly, TCP over Second (2.5G) and Third (3G) Generation Wireless Networks
  – draft-ietf-pilc-2.5g3g-10

Additional Problems Due to HTTP

HTTP Performance Problems

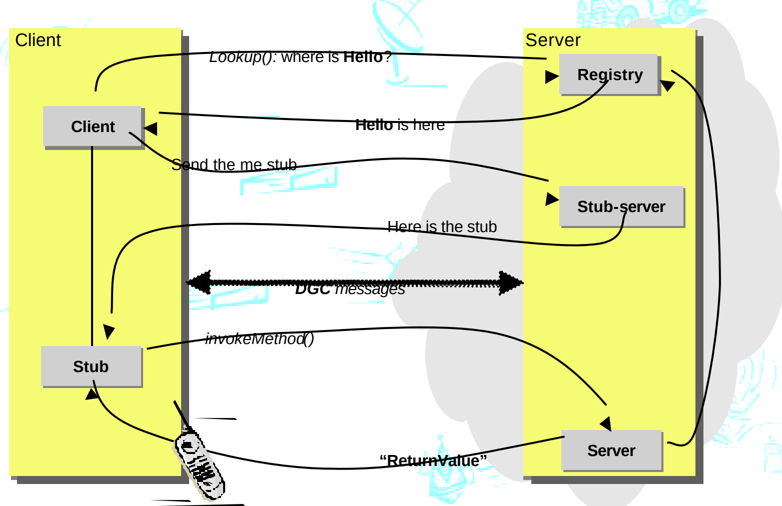
- HTTP requires an excessive number of round-trips
  - In HTTP/1.0 each hypertext document involves creation and deletion of several TCP connections
  - one for each embedded document object
- Connections are typically short
  - large overhead
  - three-way handshake
  - TCP slow-start

Java RMI Over Wireless

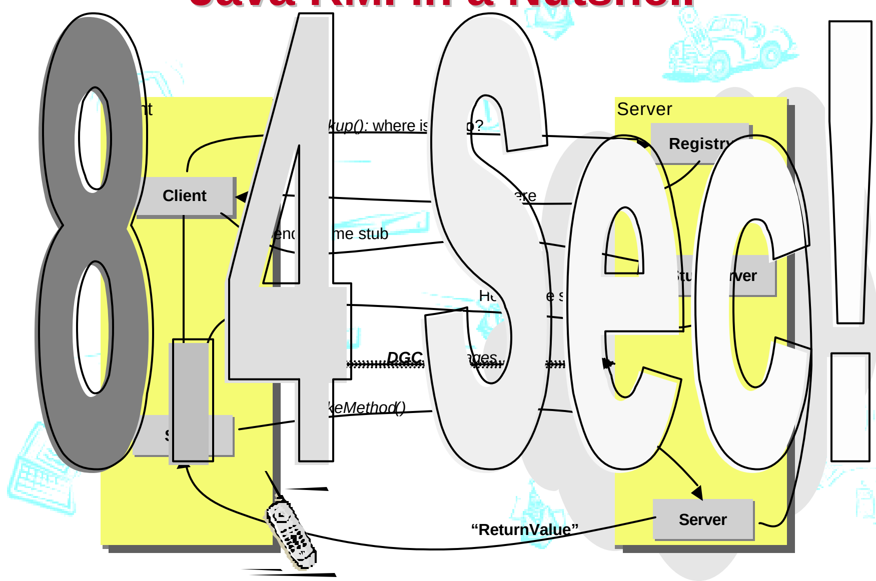
Remote Method Invocation

- RMI protocol interface lets Java objects on different hosts communicate with each other in a transparent way
- Clients can invoke methods of a remote object as if they were local methods
- Preserve the object oriented paradigm in distributed computing

Java RMI in a Nutshell

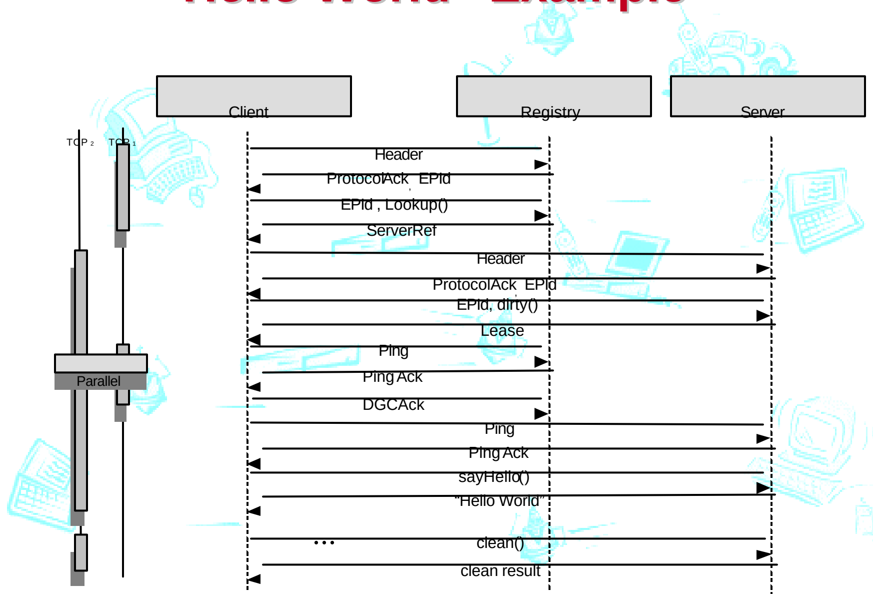


Java RMI in a Nutshell



Petrozavodsk, September 8, 2004 Kimmo Raatikainen 17

"Hello World" Example



Petrozavodsk, September 8, 2004 Kimmo Raatikainen 18

Data traffic analysis

	Client to Server and Registry (bytes)	Server and Registry to Client (bytes)	Total (bytes)
Registry Lookup	55 (6%)	276 (42%)	331 (20%)
Invocation Data	41 (4%)	37 (6%)	78 (5%)
DGC Data	831 (85%)	305 (46%)	1136 (69%)
Protocol Overhead	52 (5%)	40 (6%)	92 (6%)
Total	979 (100%)	658 (100%)	1637 (100%)

RMI Optimization

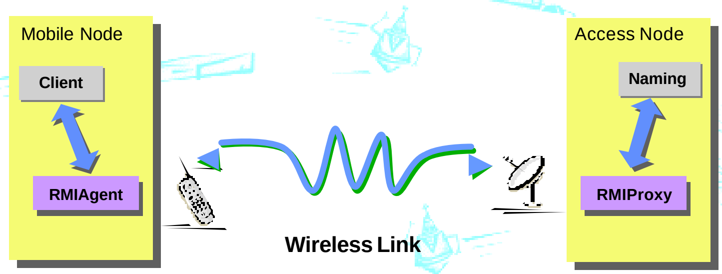
- Maintain compatibility with Java RMI specifications
- Avoid redundancy in communication protocol
- Use compression and caching to minimize data transmission

Java RMI Optimization

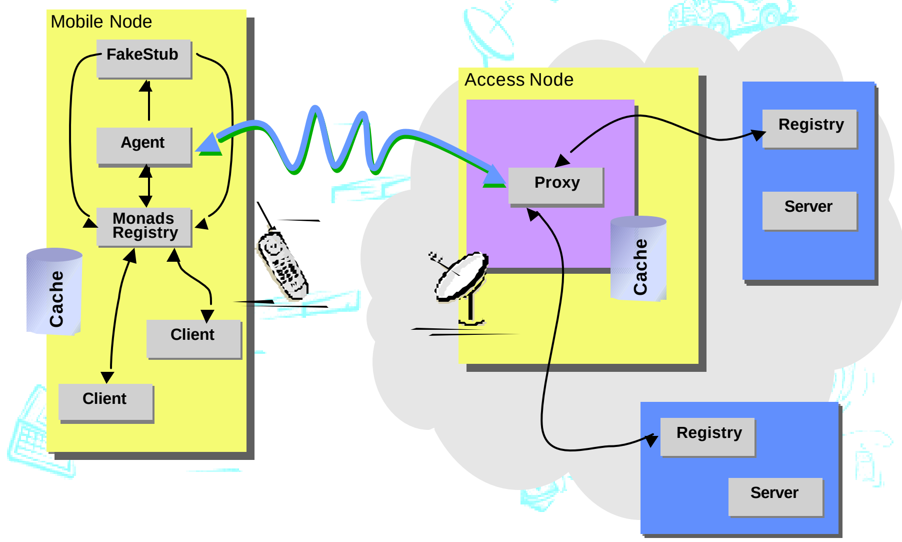
- Protocol
  - Use of Mediators to minimize the exchange of data through the wireless link.
- Data Communication
  - Optimized Communication: Compress and Optimize data communication
- Stub&Class Loading
  - If possible, avoid to download stubs

Protocol Optimization

- The idea is to de-couple the connection between the client and the server using mediators.

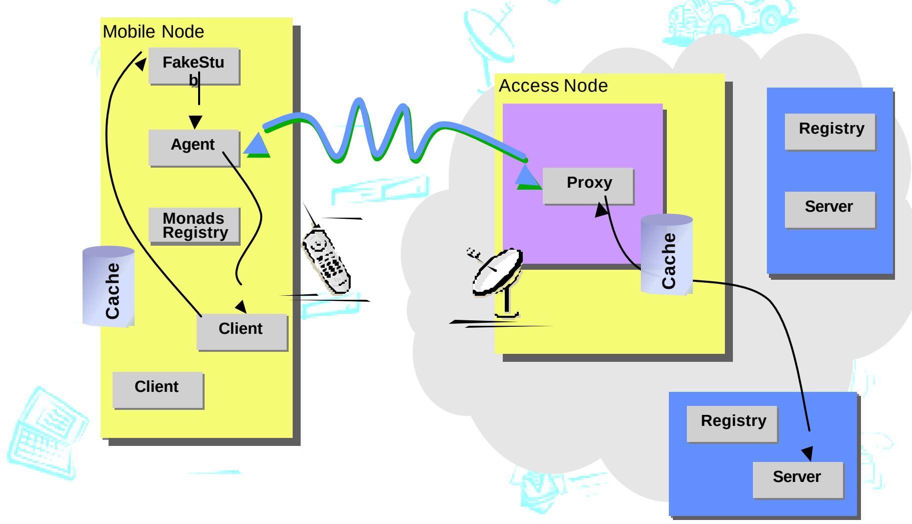


Optimized RMI



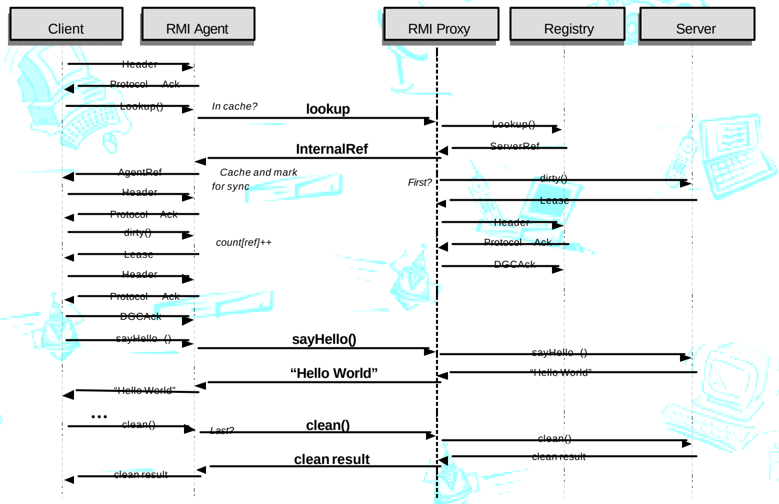
Petrozavodsk, September 8, 2004 Kimmo Raatikainen 23

Optimized RMI



Petrozavodsk, September 8, 2004 Kimmo Raatikainen 24

Optimized Remote Invocation



Comparison between Normal RMI and Optimized RMI

	Registry Invocation	Remote Invocation	Total
Java RMI	7.1 sec	1.3 sec	8.4 sec
Optimized RMI	1.7 sec	0.6 sec	2.3 sec
Improvement	417%	216%	365%

Conclusions

- Designers cannot just “plug-in” wireless communication to existing solutions
- Wireless issues extend their influence also to middleware component and eventually to applications
- Solutions are there, just mostly ignored

Wireless Java RMI Background

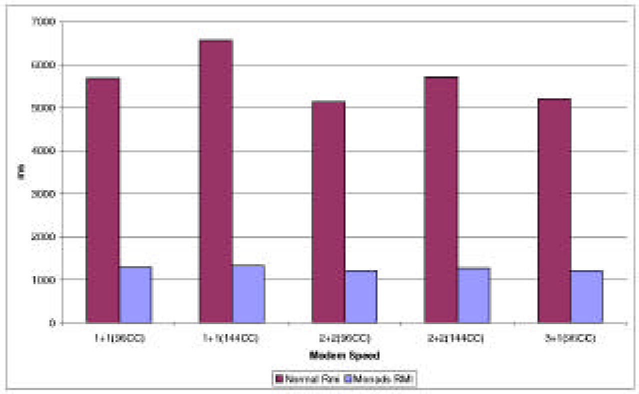
Test Arrangments

- Operating Systems
  - Clients:
    - Windows98
    - Linux (Red Hat 6.1, kernel 2.2.14)
  - Server
    - Windows NT (SP 6)
    - Linux (Red Hat 6.1, kernel 2.2.14)

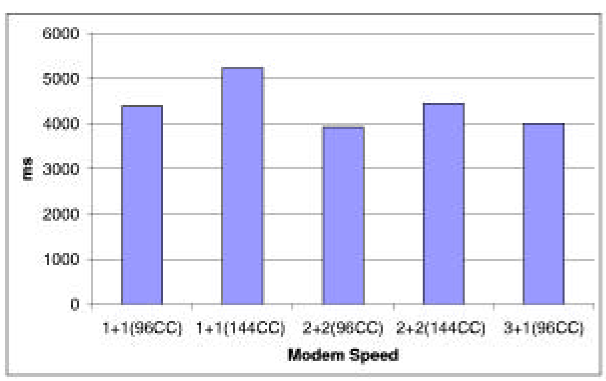
Test Arrangements

- Java Virtual Machine
  - Sun JDK 1.2.2 (Linux and Windows)
- Wireless communication
  - GSM HSCSD (5 configurations)
- Benchmark Suite
  - KaRMI from University of Karlsruhe

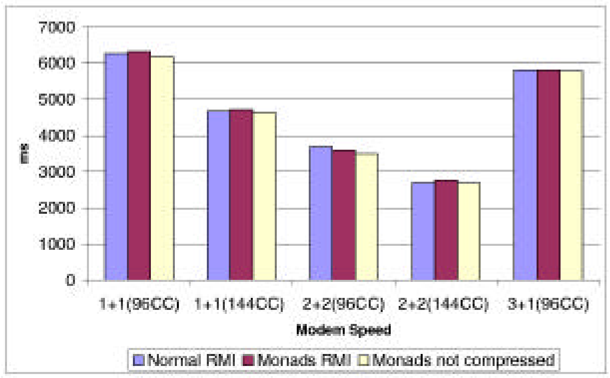
Lookup Results (windows)



Lookup Differences



Invocation Results
Image Uplink (Linux)



Invocation Results
Two-way Text uplink