

# *Publish/Subscribe Systems on Node and Link Error-Prone Mobile Environments*

**Wireless and Mobile Systems Workshop, Atlanta May 24<sup>th</sup> 2005**

**Sangyoon Oh**, Sangmi Lee Pallickara,  
Sunghoon Ko,  
Jai-Hoon Kim, Geoffrey C. Fox  
Community Grids Laboratory and Computer Science  
Indiana University,  
School of Information and Communication  
Ajou University

# Contents

---

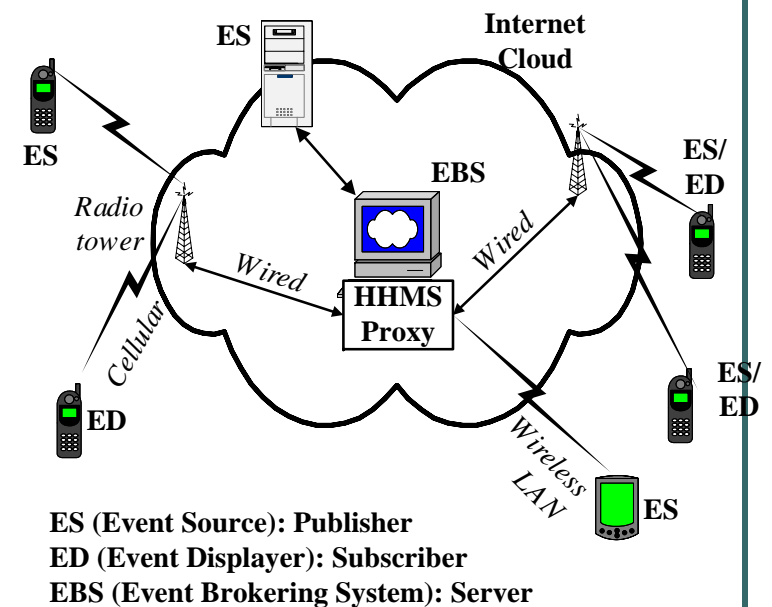
- Introduction
- Cost model
- Cost analysis
- Performance comparison
- Conclusion

<http://grids.ucs.indiana.edu/ptliupages/hhms/pub-sub.html>

# *Introduction*

# Introduction

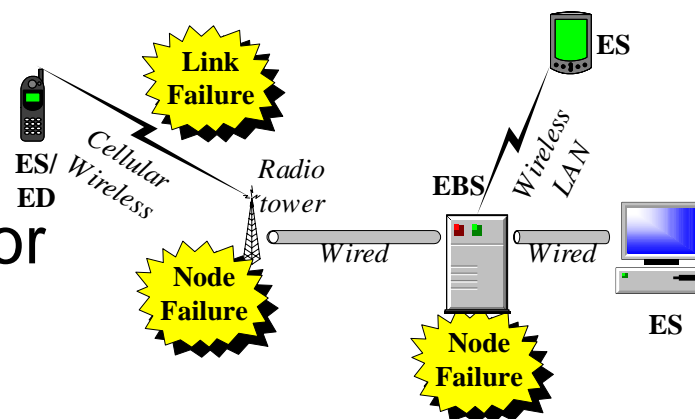
- Advantages of publish/subscribe systems in mobile computing
  - Intermittent and high-latency
  - Decoupling publisher and subscriber
  - Appropriate in mobile and ubiquitous environments
  - *Data dissemination services*
  - *Information sharing*
  - *Service discovery*



# Motivations

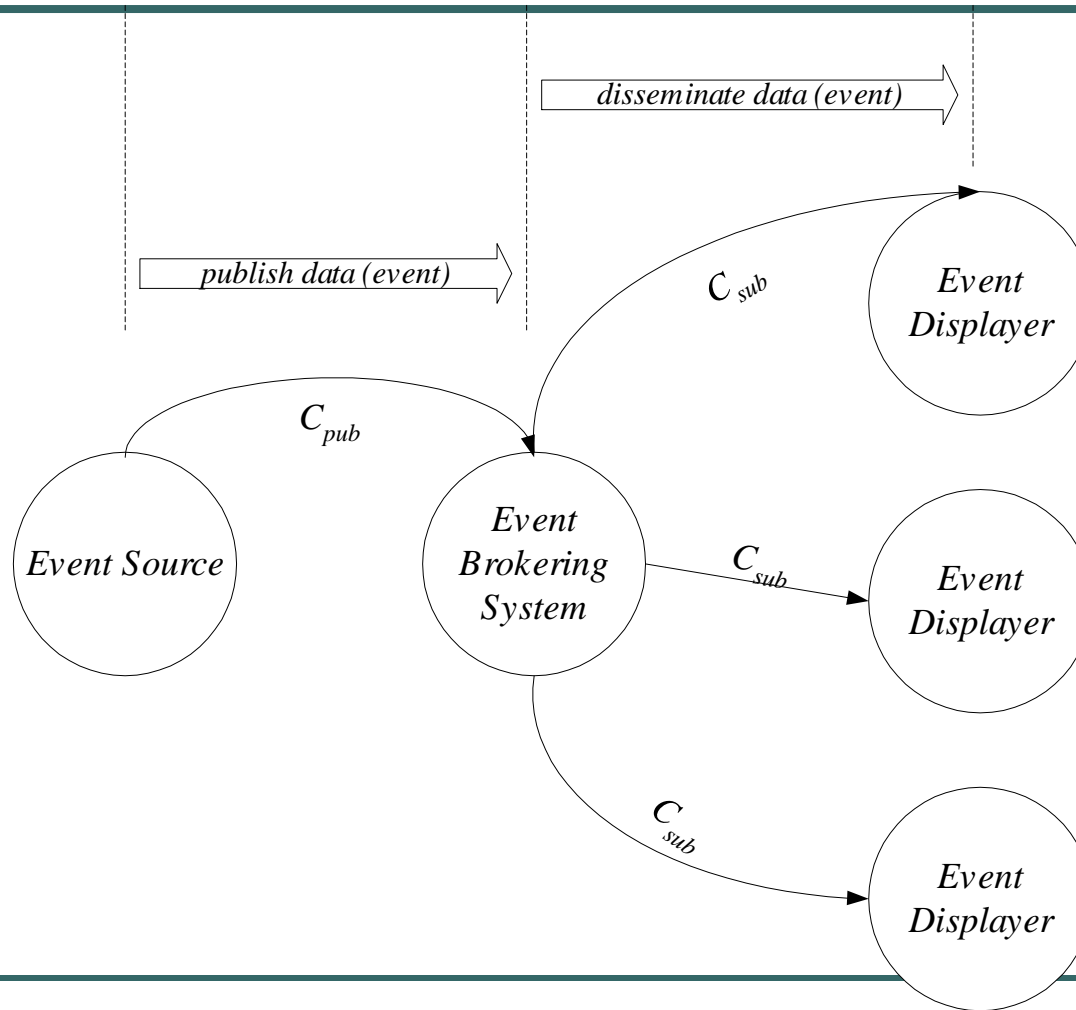
---

- Mobile environments are error prone
  - Wireless link disconnection, Power Exhaustion
- In the presentation, we analyze the influence of error
  - Mobile link and node
  - Comparison pub/sub to client-server and polling models



# *Cost Model*

# System model



# System parameters

---

$\alpha$  (publish rate)

$\beta$  (request rate or process  
(reference access) rate)

$c_{ps}(\alpha)$  (publish/subscribe cost per event)

$c_{rr}(\beta)$  (cost per request and reply)

$c_{poll}(\alpha, T)$  (cost of periodic publish or polling)

$s(n)$  (effect of sharing among  $n$  subscribers)

$t_{ps}$  (time delay for publish/subscribe)

$t_{rr}$  (time delay for request and reply)

$\lambda, \lambda_s, \lambda_c$  (failure rate of  
communication link, server, and client)

$\mu, \mu_s, \mu_c$  (recovery rate of  
communication link, server, and client)

# *Cost Analysis*

# Considerations and Assumptions

---

- Analyzing four models:  
Disconnection (failure) and reconnection (recovery) of link and node failure
  1. Publish/subscribe
  2. Event (broker) based request/reply
  3. Conventional request/reply
  4. Periodic polling
- Cost metric → time delay to transfer message and additional time required due to failure of communication links and nodes of servers and clients
- *Persistent files*  
for events are sharable among servers. When a server fails, the another server can take over the role and publish/server model can continue its transactions.
- *Durable database*  
saves events log and provides clients events history after clients recovers from failures.

# Characteristics on Failures

<i>Models</i>	<i>Types</i>	<i>Link failures</i>	<i>Server failures</i>	<i>Client failures</i>
<i>publish /subscribe</i>	<i>publish /subscribe</i>	<i>Wait until link reconnection, transaction is preserved</i>	<i>Server is replicated and transaction can be continued with sharable persistent file</i>	<i>After recovery, client can access events occurred during failure using events log on durable database</i>
<i>Event message based request/reply</i>	<i>request /reply</i>	<i>Wait until link reconnection, transaction is preserved</i>	<i>Server is replicated and transaction can be continued with sharable persistent file</i>	<i>After recovery, client can access events occurred during failure using events log on durable database</i>
<i>request/reply (RPC)</i>	<i>request /reply</i>	<i>Wait until link reconnection, transaction needs to restart</i>	<i>Transaction needs to restart after recovery</i>	<i>Lost event or data during client failure</i>
<i>periodic polling</i>	<i>any</i>	<i>Wait until link reconnection</i>	<i>Depends on system</i>	<i>Depends on system</i>

# Cost Analysis I

## Publish/subscribe model

- Cost of message transfer =  $(c_{pub} + n s(n)c_{sub})$ , plus delay time from link and/or node failure

- Failure of communication link

1. Probability of disconnection during transaction  $(\mu/(\lambda+\mu)(1 - \epsilon^{-(t_{pub}+t_{sub})\lambda}))$
2. Probability of communication link disconnection  $(\lambda/(\lambda+\mu))$

Probability of subscriber access event before reconnection

$$t_{pub} + t_{sub} + \left\{ \frac{\mu}{\lambda+\mu} (1 - \epsilon^{-(t_{pub}+t_{sub})\lambda}) + \frac{\lambda}{\lambda+\mu} \right\} \left\{ \frac{\beta}{\mu+\beta} \right\} (1/\mu).$$

Probability of disconnection

Aver. Delay time

- Failure of server

- Assumption: 1) Another server takes over the failed server, 2) ignores the cost required for transaction from failed server to backup server

$$t_{ps} = t_{pub} + t_{sub}$$

- Failure of client

- Assumption: Durable data  $\rightarrow$  Client get any information after recovery ( $n_{log}$ )
- Probability of  $i$  event occurred between failure and recovery  $\rightarrow \left( \frac{\alpha}{\alpha + \mu_c} \right)^i \frac{\mu_c}{\alpha + \mu_c}$
- If  $i > n_{log}$ ,  $i - n_{log}$  events are lost due to exceeding limitation of capacity for event log  
Thus, aver. number of lost events per client failure  $\rightarrow$

$$\sum_{i=n_{log}+1}^{\infty} \left( \frac{\alpha}{\alpha + \mu_c} \right)^i \frac{\mu_c}{\alpha + \mu_c} (i - n_{log}) = \left( \frac{\alpha}{\alpha + \mu_c} \right)^{n_{log}} \frac{\alpha}{\mu_c}$$

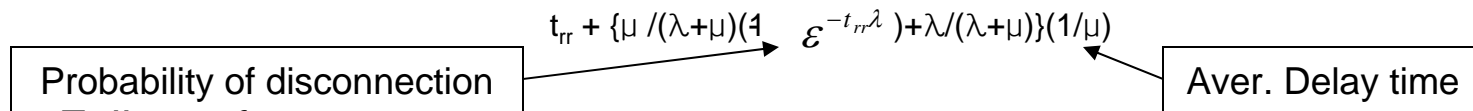
# Cost Analysis II

## Event (broker)-based request/reply

- Cost of message transfer =  $c_{rr}$ , plus delay time from link and/or node failure

- Failure of communication link

1. Probability of disconnection during transaction  $(\mu/(\lambda+\mu))(1 - e^{-t_{rr}\lambda})$
2. Probability of communication link disconnection  $(\lambda/(\lambda+\mu))$



- Failure of server

- Assumption: 1) Another server takes over the failed server, 2) ignores the cost required for transaction from failed server to backup server

$$t_{rr}$$

- Failure of client

- Assumption: Durable data → Client gets any information after recovery
- Probability of  $i$  event occurred between failure and recovery

$$\rightarrow \left( \frac{\alpha}{\alpha + \mu_c} \right)^i \frac{\mu_c}{\alpha + \mu_c}$$

# Cost Analysis III

## RPC Based Request/Reply model

- Assumption
  - No-persistent, No-durable
  - Additional overhead → ‘useless computation’ (re-generating request/reply, which is  $< t_{rr} + t_{proc}$ )
  - ‘Server recovery overhead’ is ignored like other models.

- Failure of communication link

1. Link doesn't fail  
probability is  $1 - \mu/(\lambda+\mu)(1 - \epsilon^{-(t_{pub} + t_{sub})\lambda}) - \lambda/(\lambda+\mu) \rightarrow t_{rr}$
2. Link failed  
probability is  $\lambda/(\lambda+\mu) \rightarrow$  recovery cost  $1/\mu +$  restart  $c_{rr}$
3. Link failed during transaction  
probability is  $\mu/(\lambda+\mu)(1 - \epsilon^{-(t_{pub} + t_{sub})\lambda})$

Useless Computation:  $E(t_{rr} + t_{proc}, \lambda)$

The function E denotes amount of time, length x, and failure ratio of  $\lambda$

$$E(t_{rr} + t_{proc}, \lambda) = \int_0^x \frac{\lambda \epsilon^{-\lambda t}}{1 - \epsilon^{-\lambda t}} dt = \lambda^{-1} - \frac{x \epsilon^{-\lambda x}}{1 - \epsilon^{-\lambda x}}$$

$$c_{rr} = [1 - \{ \mu/(\lambda+\mu) (1 - \epsilon^{-(t_{pub} + t_{sub})\lambda}) + \lambda/(\lambda+\mu) \}] t_{rr} + \mu/(\lambda+\mu) (1 - \epsilon^{-(t_{pub} + t_{sub})\lambda}) \{ E(t_{rr} + t_{proc}, \lambda) + 1/\mu + c_{rr} \} + \lambda/(\lambda+\mu) \{ (1/\mu) + c_{rr} \}$$

$$\rightarrow c_{rr} = [ [1 - \{ \mu/(\lambda+\mu) (1 - \epsilon^{-(t_{pub} + t_{sub})\lambda}) + \lambda/(\lambda+\mu) \}] t_{rr} + \mu/(\lambda+\mu) (1 - \epsilon^{-(t_{pub} + t_{sub})\lambda}) \{ E(t_{rr} + t_{proc}, \lambda) + 1/\mu \} + \lambda/(\lambda+\mu) (1/\mu) ] / [ 1 - \{ \mu/(\lambda+\mu) (1 - \epsilon^{-(t_{pub} + t_{sub})\lambda}) + \lambda/(\lambda+\mu) \}]$$

# Cost Analysis III

## RPC Based Request/Reply model

- Failure of server

1. Server doesn't fail  
: probability is  $1 - \mu_s / (\lambda_s + \mu_s) (1 - e^{-(t_{pub} + t_{sub})\lambda_s}) - \lambda_s / (\lambda_s + \mu_s) \rightarrow t_{rr}$
2. Server failed  
: probability is  $\lambda_s / (\lambda_s + \mu_s) \rightarrow$  recovery cost  $1/\mu_s +$  restart  $c_{rr}$
3. Server failed during transaction  
: probability is  $\mu_s / (\lambda_s + \mu_s) (1 - e^{-(t_{pub} + t_{sub})\lambda_s})$

$$c_{rr} = [1 - \{ \mu_s / (\lambda_s + \mu_s) (1 - e^{-(t_{pub} + t_{sub})\lambda_s}) + \lambda_s / (\lambda_s + \mu_s) \}] t_{rr} + \mu_s / (\lambda_s + \mu_s) (1 - e^{-(t_{pub} + t_{sub})\lambda_s}) \{ E(t_{rr} + t_{proc}, \lambda_s) + 1/\mu_s + c_{rr} \} + \lambda_s / (\lambda_s + \mu_s) \{ (1/\mu_s) + c_{rr} \}$$

$$\rightarrow c_{rr} = [ [1 - \{ \mu_s / (\lambda_s + \mu_s) (1 - e^{-(t_{pub} + t_{sub})\lambda_s}) + \lambda_s / (\lambda_s + \mu_s) \}] t_{rr} + \mu_s / (\lambda_s + \mu_s) (1 - e^{-(t_{pub} + t_{sub})\lambda_s}) \{ E(t_{rr} + t_{proc}, \lambda_s) + 1/\mu_s \} + \lambda_s / (\lambda_s + \mu_s) (1/\mu_s) ] / [ 1 - \{ \mu_s / (\lambda_s + \mu_s) (1 - e^{-(t_{pub} + t_{sub})\lambda_s}) + \lambda_s / (\lambda_s + \mu_s) \}]$$

- Failure of client

- Assumption  
current data only and no-logging data
- Only the data occurred during the failure will be lost.

Useless Computation:  $E(t_{rr} + t_{proc}, \lambda_s)$

The function E denotes amount of time, length x, and failure ratio of  $\lambda$

$$E(t_{rr} + t_{proc}, \lambda) = \int_0^x \frac{\lambda e^{-\lambda t}}{1 - e^{-\lambda t}} dt = \lambda^{-1} - \frac{x e^{-\lambda x}}{1 - e^{-\lambda x}}$$

# Cost Analysis IV

## Periodic (Polling) model

- Assumption

- Polling is delayed until communication link is recovered.
- the persistent file and durable database

Probability of disconnection

- Link was disconnected or is disconnected during the transaction

- Time delay =

$$t_{poll}(\alpha, T) + \left\{ \frac{\mu}{\lambda + \mu} \varepsilon^{-t_{poll}(\alpha, T)\lambda} + \frac{\lambda}{\lambda + \mu} \right\} (1/\mu)$$

- Conceptual cost =

$$c_{poll}(\alpha, T) + [1 - \left\{ \frac{\mu}{\lambda + \mu} \varepsilon^{-t_{poll}(\alpha, T)\lambda} + \frac{\lambda}{\lambda + \mu} \right\}] c_{delay}(\alpha, T) + \left\{ \frac{\mu}{\lambda + \mu} \varepsilon^{-t_{poll}(\alpha, T)\lambda} + \frac{\lambda}{\lambda + \mu} \right\} c_{delay}(\alpha, T + 1/\mu)$$

- Failure of communication link

$$t_{rr} + \left\{ \frac{\mu}{\lambda + \mu} (1 - \varepsilon^{-t_{rr}\lambda}) + \frac{\lambda}{\lambda + \mu} \right\} (1/\mu)$$

- Failure of server

$$t_{rr}$$

- Failure of client

$$\left( \frac{\alpha}{\alpha + \mu_c} \right)^i \frac{\mu_c}{\alpha + \mu_c}$$

# *Performance Comparisons*



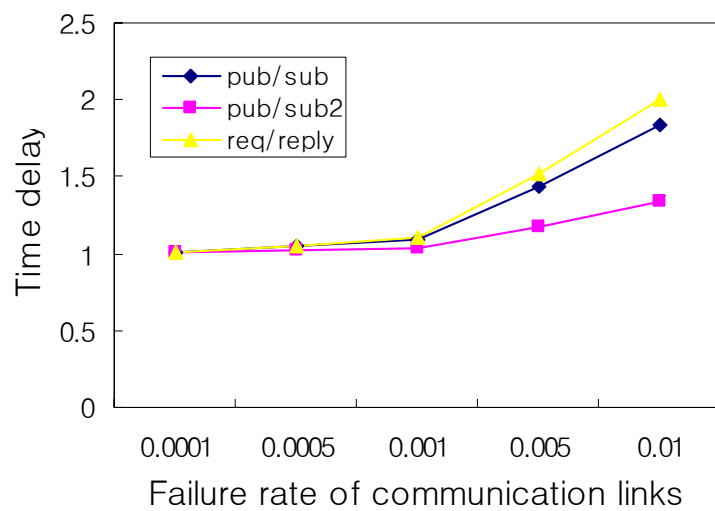
# Performance Comparisons by parametric analysis

- Publish/subscribe system requires less communication delay.
- Assumption

Models	Delays
Publish/subscribe system & Event-based request/reply	Only need communication delay ← Backup server
RPC-based request/reply system	Need additional time delay and lost communication ← Server failure

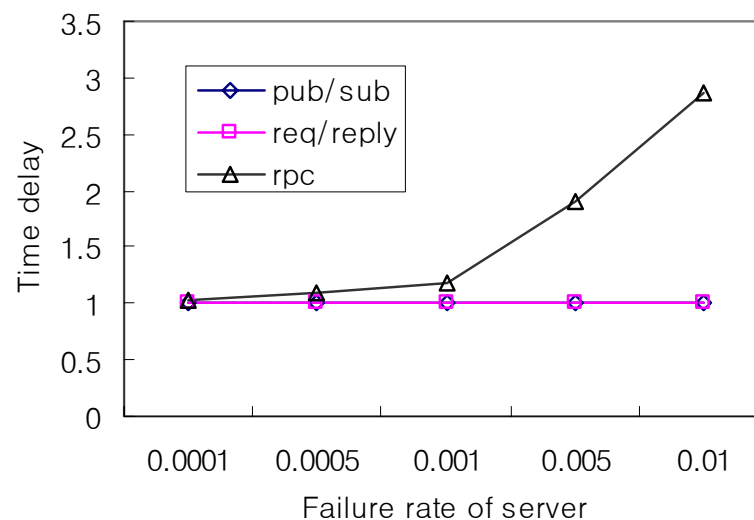
System parameters	
Param.	Values
$\alpha, \beta$	0.5
$C_{ps}, C_{rr}$	2
$C_{pub}, C_{sub}$	1
$C_{poll}(\alpha, T)$	1 or $\alpha T$
$C_{delay}(\alpha, T)$	0, T, or $\alpha T$
$s(n)$	$1/n - 1$
$\lambda, \lambda_s, \lambda_c$	0.0001 – 0.5
$\mu, \mu_s$	0.1
$\mu_c$	0.05 - 0.1
$t_{ps}, t_{rr}$	1
$t_{proc}$	1 or 5
$t_{poll}(\alpha, T)$	1, T, or $\alpha T$

# Performance comparisons II



## Time delay per transaction

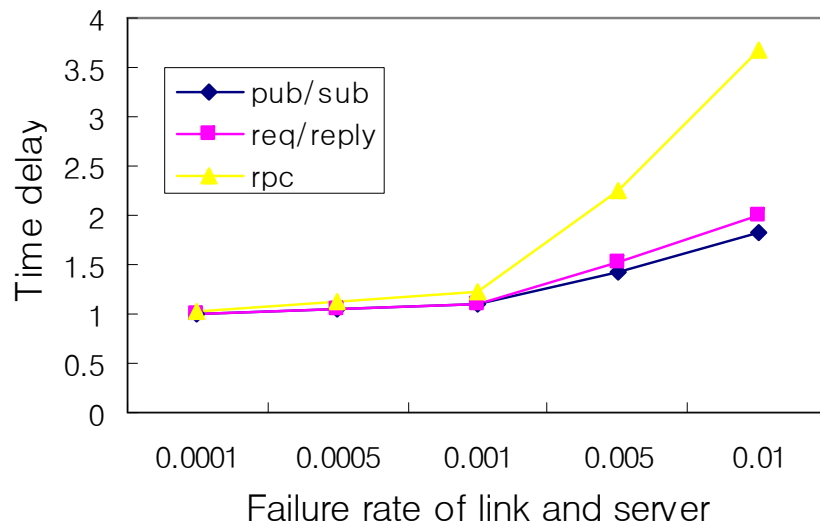
( $\alpha=0.5$ ,  $s(n)=1$ ,  $t_{ps}=1$ ,  $t_{rr}=1$ ,  $\mu=0.1$ ,  
and  $\beta=0.5$  for pub/sub and req/reply  
and  $\beta=0.2$  for pub/sub2)



## Time delay per transaction

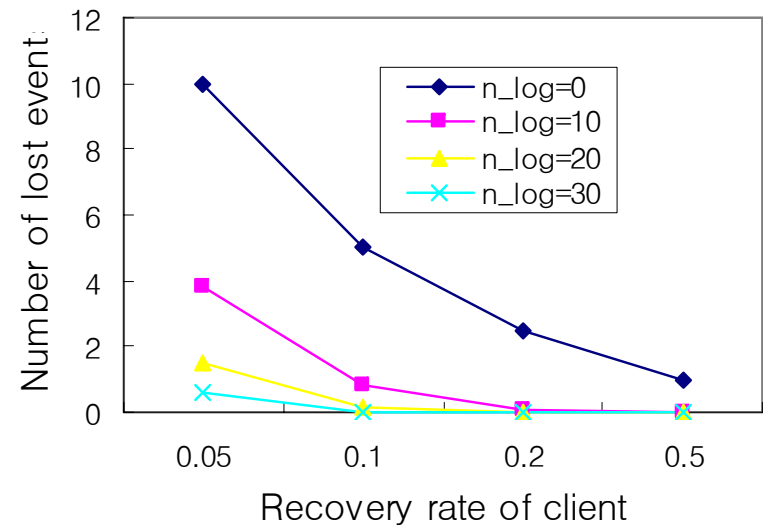
( $\alpha=0.5$ ,  $t_{ps}=1$ ,  $t_{rr}=1$ ,  $\beta=0.5$ , and  
 $\mu s=0.1$ )

# Performance comparisons III



## Time delay per transaction

( $\alpha=0.5$ ,  $t_{ps}=1$ ,  $t_{rr}=1$ ,  $t_{poll}=1$ ,  $\beta=0.5$ ,  
 $\mu=0.1$ ,  $\mu_s=0.1$ )



## Number of lost events per client failure by varying recovery rate of client

(effectiveness of durable database which logs events)

# Experimental Setup

---

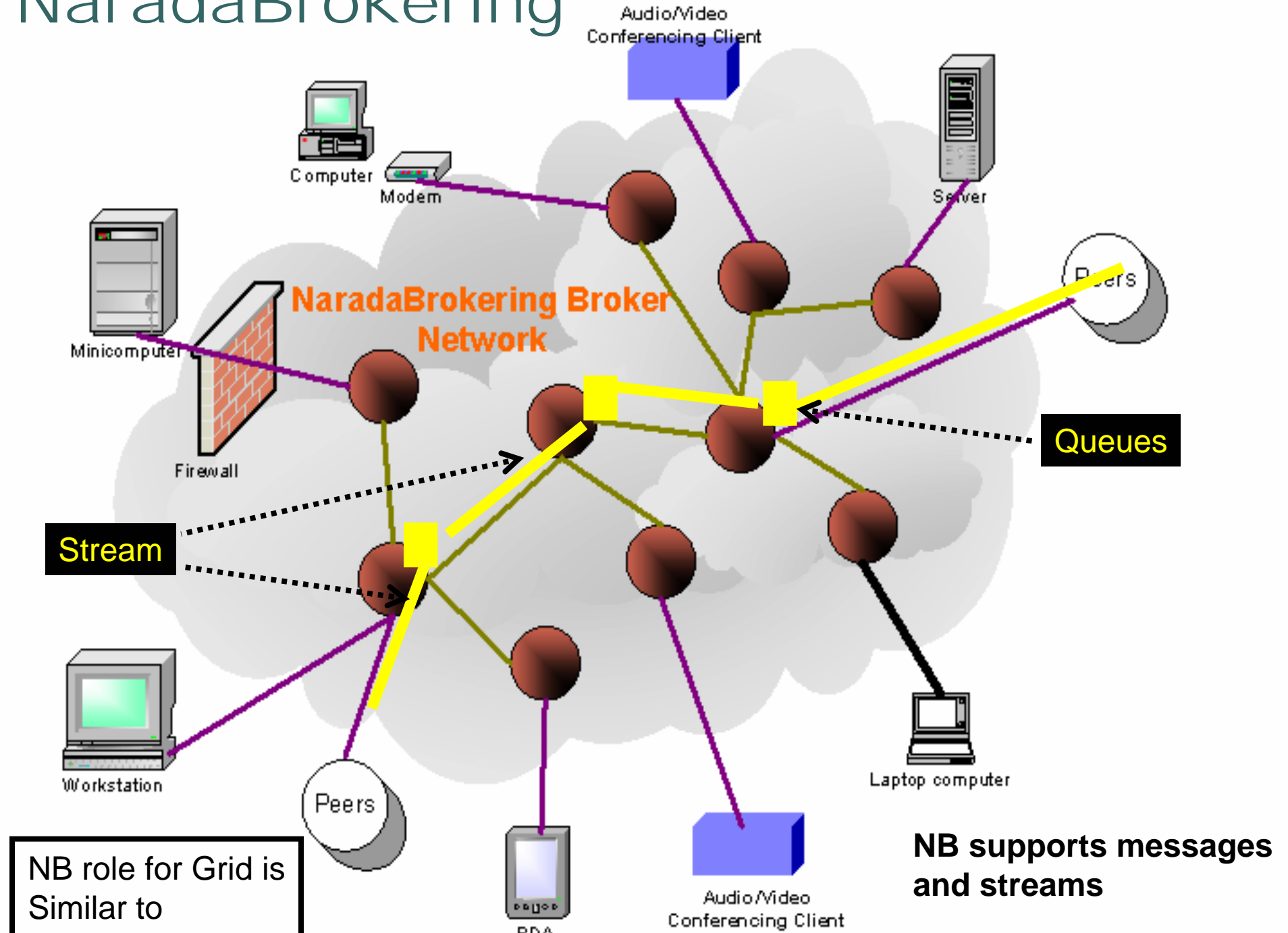
- Using NaradaBrokering as message brokering system -- MOM (Message Oriented Middleware) **for publish/subscribe**
- Using HHMS (Handheld Message Service) as primary application level transport protocol **for publish/subscribe** between mobile device and conventional wired environment
- Conventional RPC code using J2SE and J2ME MIDP 2.0 **for request/reply**
- Experimental Specifications
  - Treo600:
    - PalmOS 5.2 144MHz ARM, 32MB, Sprint PCS Service (<14.4kbps)
  - HHMS Gateway and NaradaBrokering:
    - Linux 7.3, Pentium III 1GHz, 512MB
  - Timer: Linux native timer by JNI

# NaradaBrokering

---

- Developed by Community Grids Laboratory of Indiana University
- Message Oriented Middleware (MOM)
  - **Multiple protocol transport support:** In publish-subscribe Paradigm with different Protocols on each link
  - **Subscription Formats**
  - **Reliable delivery**
  - **Ordered delivery**
  - **Recovery and Replay**
  - **Security**
  - **Message Payload options**
  - **Messaging Related Compliance**
  - **Grid Feature Support**
  - **Web Services supported**

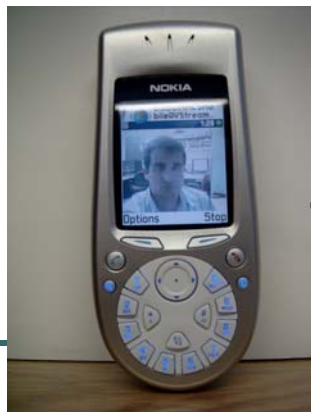
# NaradaBrokering



# Handheld Messaging Service

---

- Light-weight publish/subscribe message service framework for mobile devices
- Optimized application level transport protocol using byte message format
- Provide core-subset of JMS API



# Experiment results

---

- Simple server → sending 10 bytes messages
- Echo client → return the message back
- Round Trip Time (RTT)
  - ← median is chosen among 2000 iteration
- It shows matching result with our parametric analysis

$$t_{ps} = t_{rr} = 1$$

## Delay time of sending message

	<i>Wireless</i>	<i>Wired</i>	<i>Total (msec.)</i>
<i>RPC</i>	1290.7 ( <i>client – gateway</i> )	39.9 ( <i>gateway – server</i> )	<b>1330.6 (<math>t_{rr}</math>)</b>
<i>Pub/Sub</i>	1448.4 ( <i>ED – EBS</i> )	89.7 ( <i>EBS – ES</i> )	<b>1538.1 (<math>t_{ps}</math>)</b>

*Conclusion*

# Conclusion

---

- Advantages of publish/subscribe system in push based mobile applications.
- Problem: Error-prone mobile environments
  - Link disconnection
  - Power exhaustion
- Publish/subscribe system has improved performance and effectiveness on failure of client and server node, and disconnection of communication link.
- Our analysis shows publish/subscribe system is more durable than client/server model in mobile environments