Service-orientation and Jini

Strengths and Obstacles

Zoltan Juhasz
Pannon University
Veszprem, Hungary

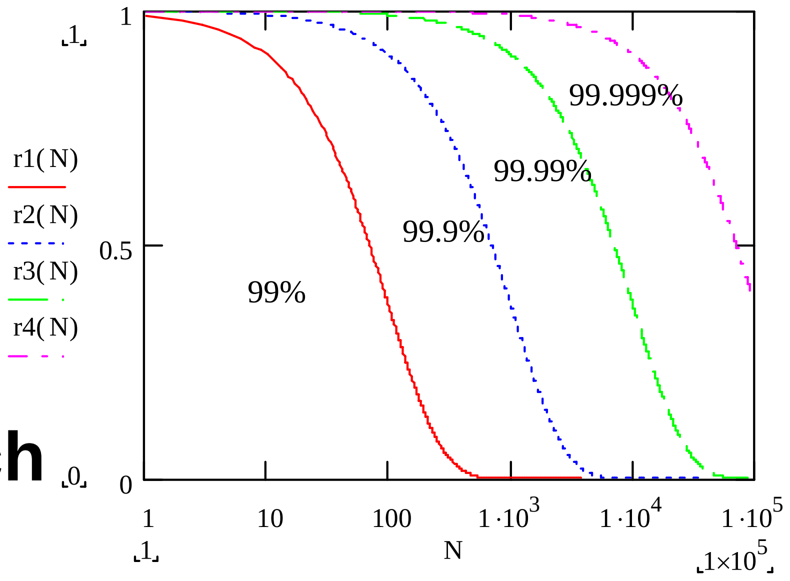
Motivation

- The JGrid project
  - Investigate whether Jini could help Grid Computing
  - Produced interesting results and services
- Not much interest within the grid community
  - Why?
  - Have we got something wrong?
  - Is it to do with service-orientation?
  - Is it about Jini?

Central problems

- Large-scale computations
- Heterogeneous platform
- Faults -> reliability, availability
- Security
- Assumptions
  - New problems require new techniques
  - **Service-oriented approach**

availability	downtime a year
99%	3.65 days
99.9%	8.76 hours
99.99%	52.56 minutes
99.999%	5.256 minutes



Service-orientation

- Perhaps Jini is not the “right” type of service orientation?
- What is a service? Sample definitions:
  - “A **service is a unit of work** done by a service provider to achieve desired end results for a service consumer.” [http://webservices.xml.com]
  - “A computing **service specifies a collection of operations** whose execution can be triggered by inputs from service users or the passage of time.” [F. Cristian, Understanding Fault-Tolerant Distributed Systems]
  - “**Services are what you connect together using Web Services**. A service is the endpoint of a connection.” [http://www.service-architecture.com]

Service-orientation cont’d

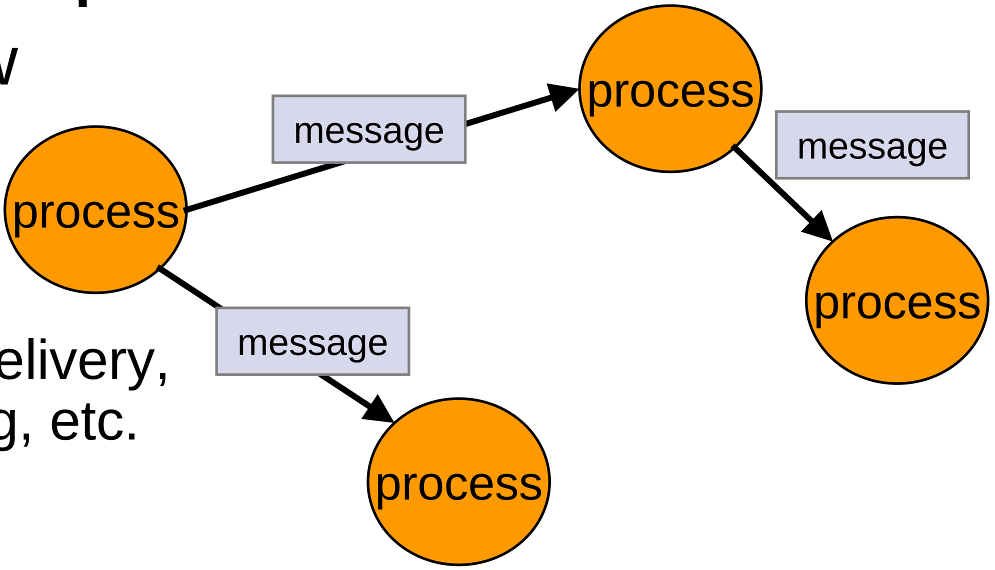
- “A **service is a set of actions** that form a coherent whole from the point of view of service providers and service requesters.” [http://www.w3.org/TR/2003/WD-ws-arch-20030808/]
- “A **service is functionality** that must be specified in the business context and in terms of contracts between the provider […] and its consumers. Implementation details should not be revealed. […] A **software service** is a type of service that is implemented by software and that **offers one or more operations**.” [Paul Allen, Service Orientation, Cambridge Univ. Press]
- “A **service is an entity** that can be used by a person, a program, or another service. A service may be a computation, storage, a communication channel to another user, a software filter, a hardware device, or another user.” [The Jini Architecture Specification]
- **Conclusion: a service is functionality, behaviour**

Web Services

- Let there be XML and SOAP
- In search of standards and protocols
  - Solving the heterogeneity, coupling, scalability, etc., etc. problems!?
- RPC style invocation in the beginning
- Move towards document-oriented (?) model
  - Passing of XML-encoded messages (documents)
  - RPC is not suited to data streaming, group communication; blocking, tightly coupled, etc
- Where and what is the service interface then?

Document-oriented Web services

- Processes
  - need minimal interfaces: “send”
  - must know messages *a priori*
    - Ontologies, agreed schemas -> protocols
  - Must have **command interpreters**
- Orchestration, workflow
  - Very much in fashion
  - But how to provide reliable operation?
    - Deadlocks, message delivery, detection, fault handling, etc.



Distributed System Models

- Object and Action model
  - aka: object-oriented systems
  - Objects represent processes, communication via RPC
- Process and Conversation model
  - aka: message-oriented systems
  - Communicating sequential processes
  - Communication via message passing
- Theory tells us they are equivalent (duals)
  - Lauer and Needham, On the Duality of Operating System Structures
  - Shrivastava, Mancini and Randell, The Duality of Fault-tolerant System Structures
- **We can map from one to the other!**

Common SOA (WS) buzzwords

- Asynchronous
  - No notion of time?
  - Blocking vs non-blocking calls?
- Loosely coupled
  - Blocking vs non-blocking calls?
  - Interface based?
  - Message passing?
- Industry standards based
  - Standards or protocols based?
- Platform and vendor neutral
  - Might be…

Distributed systems issues

- Back to basics
  - Resource sharing
  - Reliability
  - Privacy and security
  - Design tools and techniques (programming model)
  - Distribution and sharing (partitioning system and data)
  - User environment
- **Is there a technology that really helps?**

Fault tolerance

- Distributed systems are more complex
- Fault tolerance is an issue
- Redundancy is a necessary condition for fault tolerance
- Faults need to be detected -> not always possible
- Masking faults
  - Relies on Exceptions
  - Hierarchical masking
  - Group masking
  - Restart/activation
- Jini gives you
  - Java Exceptions, events, leasing, the LUS and activation

The Big Misunderstanding

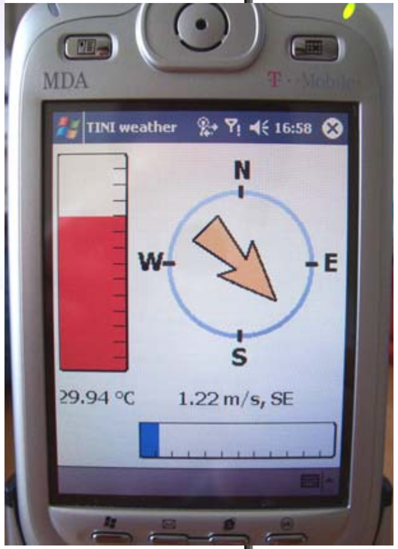
- Jini is not RMI!
  - Java supports mobile code
  - Client interacts with smart proxies
  - Using proxies is (almost) local Java
    - Interact locally, transfer control and/or as needed
- Rich choice of implementation and system partitioning options
- Data streaming, more complex communication IS possible
- Not to mention JavaSpaces for coordination, asynchronous messaging, workflow, etc.

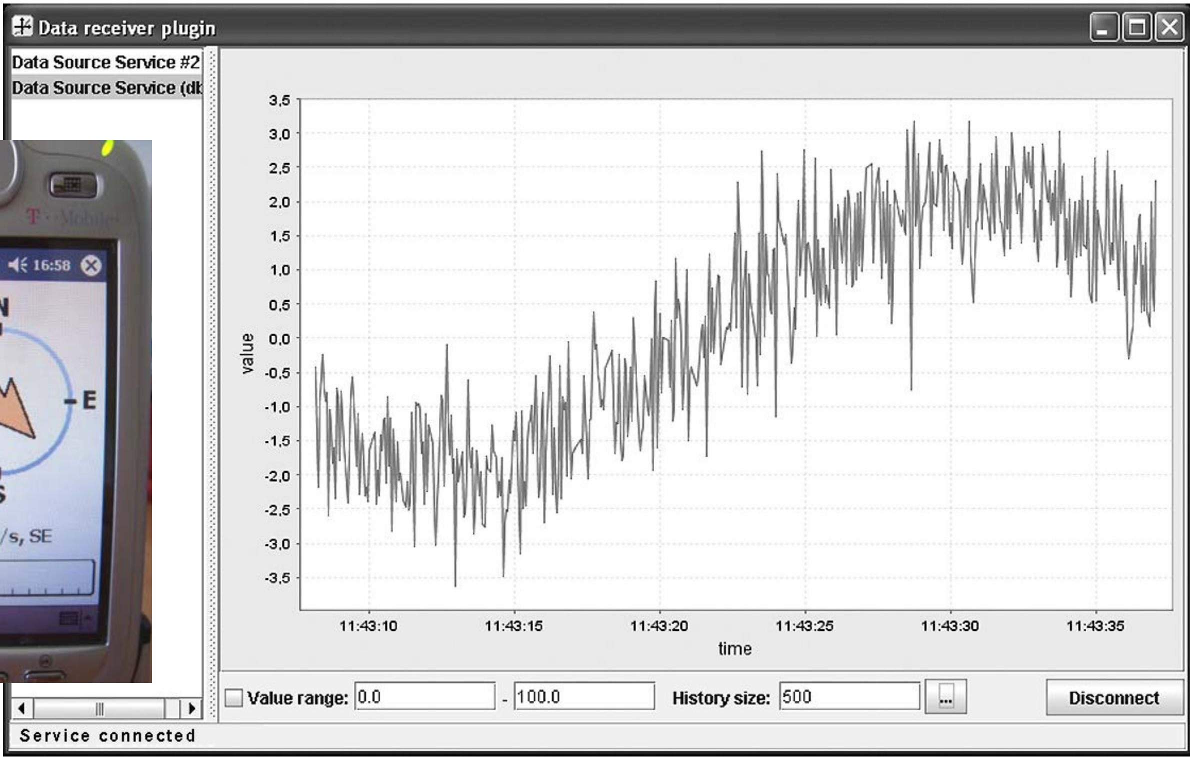
Other features

- Security
- Programming model
  - Language and infrastructure support
- Configuration
  - Wide range of deployment, system partitioning options
- User experience
  - Rich clients
  - Alternative user interfaces

Data streaming

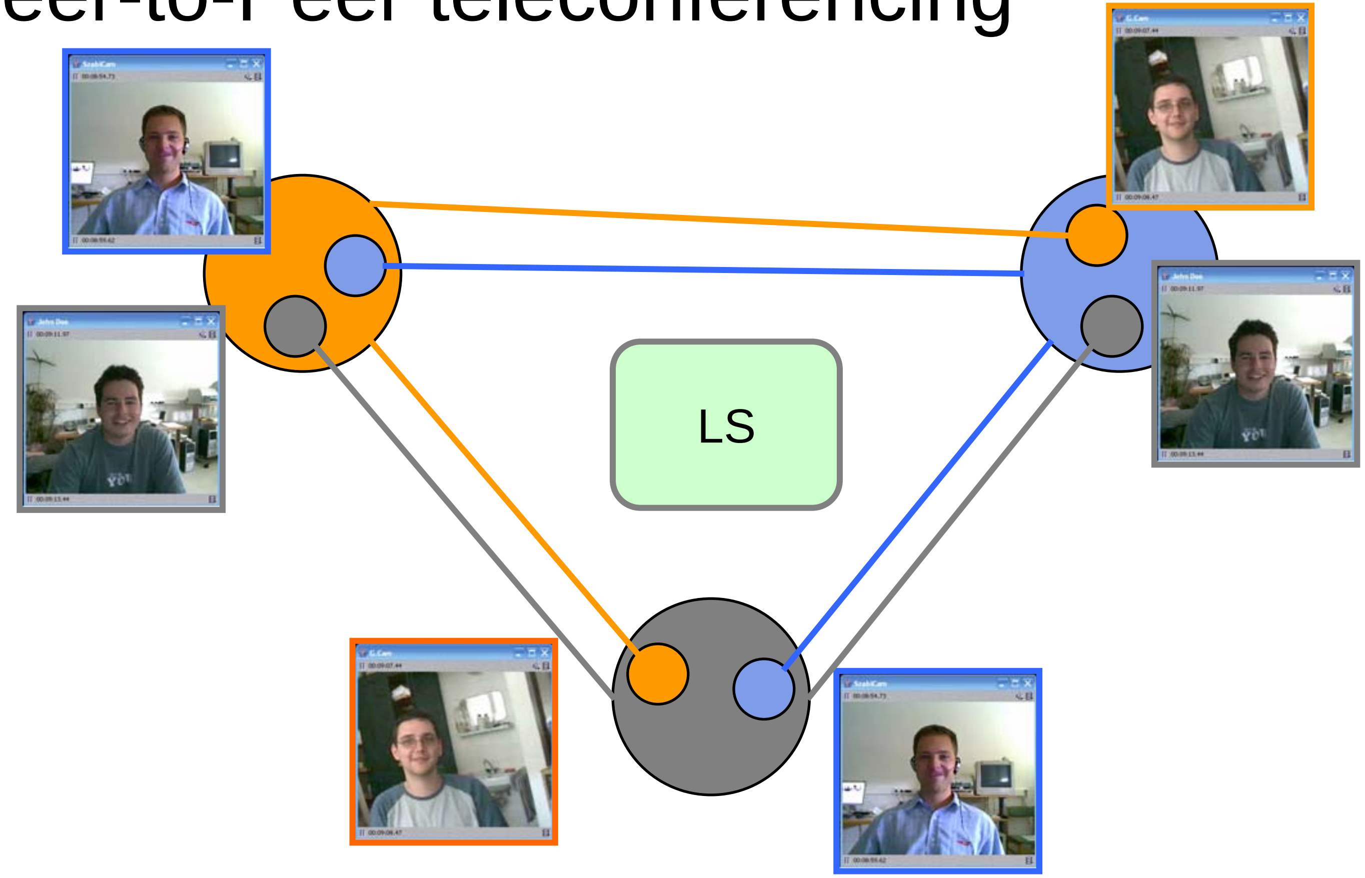
- Can represent instruments, data feed
- Weather service





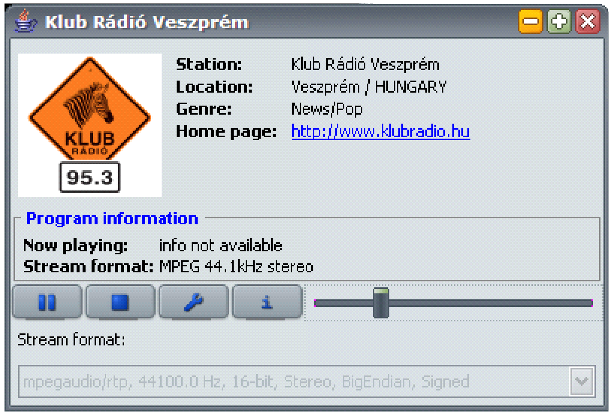
Instant Sharing

- Peer-to-Peer teleconferencing



Other examples

- Media delivery
- Control over remote jobs
- Remote file space

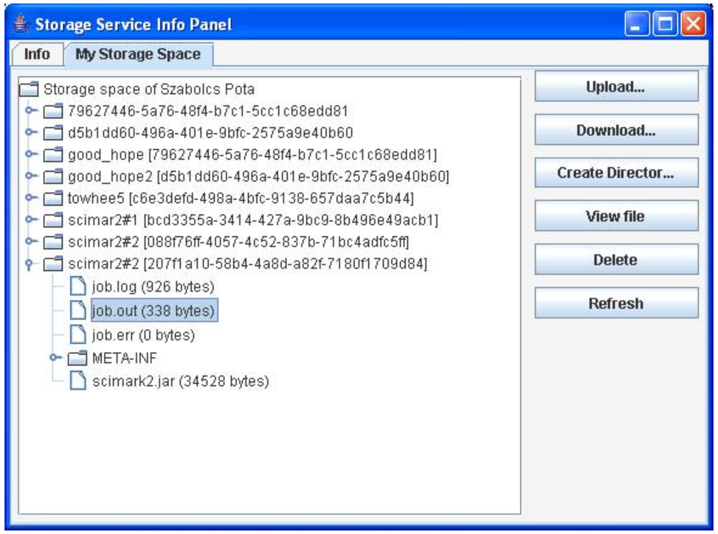

Klub Rádió Veszprém

Station:	Klub Rádió Veszprém
Location:	Veszprém / HUNGARY
Genre:	News/Pop
Home page:	http://www.klubradio.hu

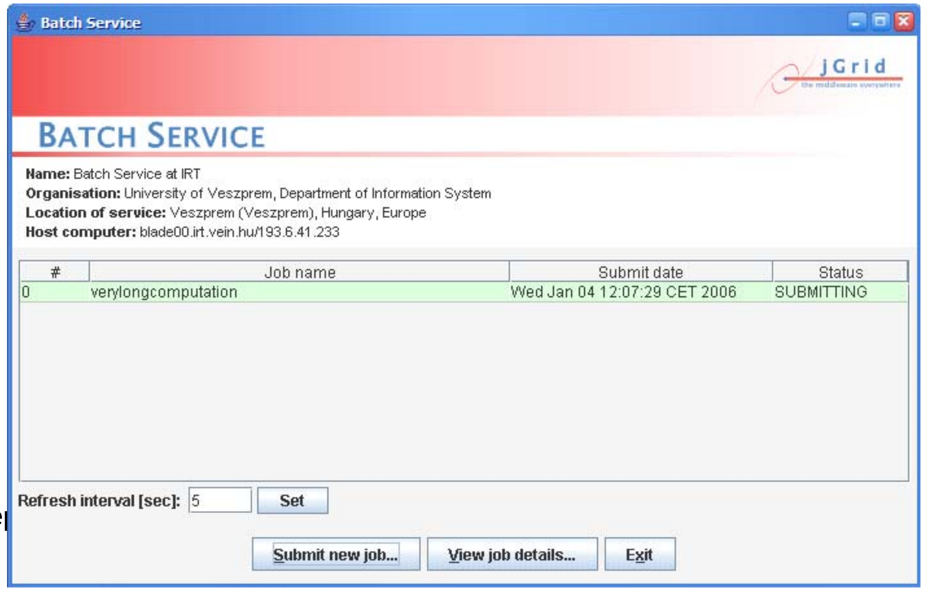
Program information

Now playing: info not available
Stream format: MPEG 44.1kHz stereo

Stream format:
mpegaudio/rtp, 44100.0 Hz, 16-bit, Stereo, BigEndian, Signed


Storage Service Info Panel

```
Storage space of Szabolcs Pota
  79627446-5a76-48f4-b7c1-5cc1c68edd81
  d5b1dd60-496a-401e-9bfc-2575a9e40b60
  good_hope [79627446-5a76-48f4-b7c1-5cc1c68edd81]
  good_hope2 [d5b1dd60-496a-401e-9bfc-2575a9e40b60]
  towhee5 [c6e3defd-498a-4bfc-9138-657daa7c5b44]
  scimar2#1 [bcd3355a-3414-427a-9bc9-8b496e49acb1]
  scimar2#2 [088f76ff-4057-4c52-837b-71bc4adfc5ff]
  scimar2#2 [207f1a10-58b4-4a8d-a82f-7180f1709d84]
    job.log (926 bytes)
    job.out (338 bytes)
    job.err (0 bytes)
    META-INF
    scimark2.jar (34528 bytes)
```


Batch Service

BATCH SERVICE

Name: Batch Service at IRT
Organisation: University of Veszprem, Department of Information System
Location of service: Veszprem (Veszprem), Hungary, Europe
Host computer: blade00.irt.vein.hu/193.6.41.233

#	Job name	Submit date	Status
0	verylongcomputation	Wed Jan 04 12:07:29 CET 2006	SUBMITTING

Refresh interval [sec]: 5

Potential problems

- Technical problems
  - Firewalls, port usage
  - Class versioning
  - Class loading and the required infrastructure
  - Discovery (description, proxies, ServiceUI)
  - Service handles
  - Restricted devices
- Social and business issues
  - Strong technological competition
  - Lack of in-depth knowledge
  - Interoperability requirements, integration
  - Assessing risks
  - Following the crowd effect
- Others?

Conclusions

- Distributed systems is hard, do not pretend it isn’t
- Jini is still misunderstood by many
- Jini is a technology that
  - focuses on the real problems of distributed systems
  - helps creating reliable distributed system by giving the right tools
  - increases our knowledge and understanding of distributed systems
- Other technologies may look more successful but how will they compare?
- These benefits will not be visible until we start building and using **large-scale** service-based environments
- The role of the Community

Acknowledgements

- Support of the Hungarian National Office for Research and Technology under Grant GVOP-3.1.1-2004-05-0035/3.0
- Sun Academic Equipment Grant
- Team members: Szabolcs Pota, Akos Pasztory, Gabor Bognar

- For more info email to: juhasz@irt.vein.hu