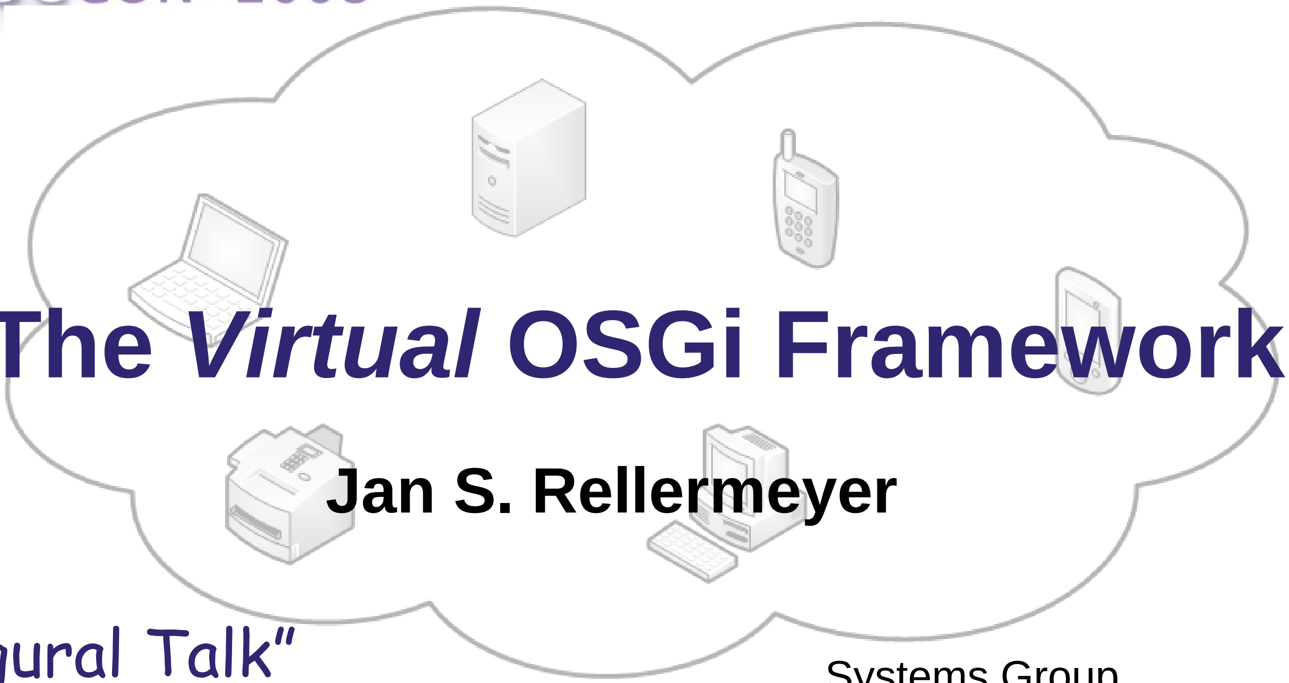
eclipseCON™ 2008



The *Virtual* OSGi Framework

Jan S. Rellermeyer

“Inaugural Talk”

Invited Researcher of
the OSGi Alliance

Systems Group
Department of Computer Science
ETH Zurich
8092 Zurich, Switzerland

OSGi Alliance

IKS Information and Communication Systems Research Group

ETH Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

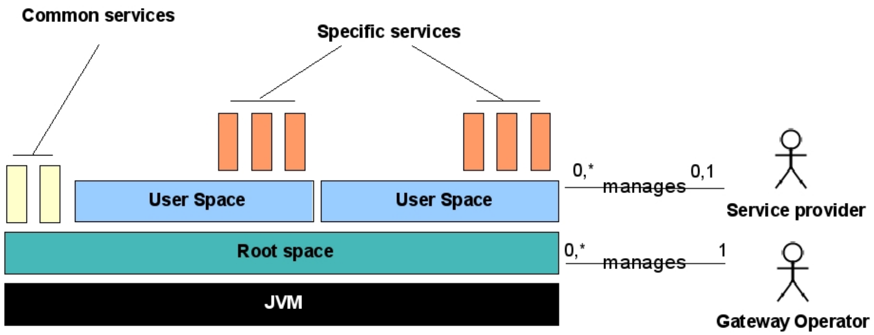
MICS

FNSNF
Fonds national suisse
Schweizerischer Nationalfonds
Fondo nazionale svizzero
Swiss National Science Foundation

March 19th 2008

Virtual OSGi? VOSGi?

- Unintended name clash
- VOSGi is work by Stephane Frenot et. al.
- Share services among virtual gateways



✗ Providers' services isolation
✗ Managament agent isolation

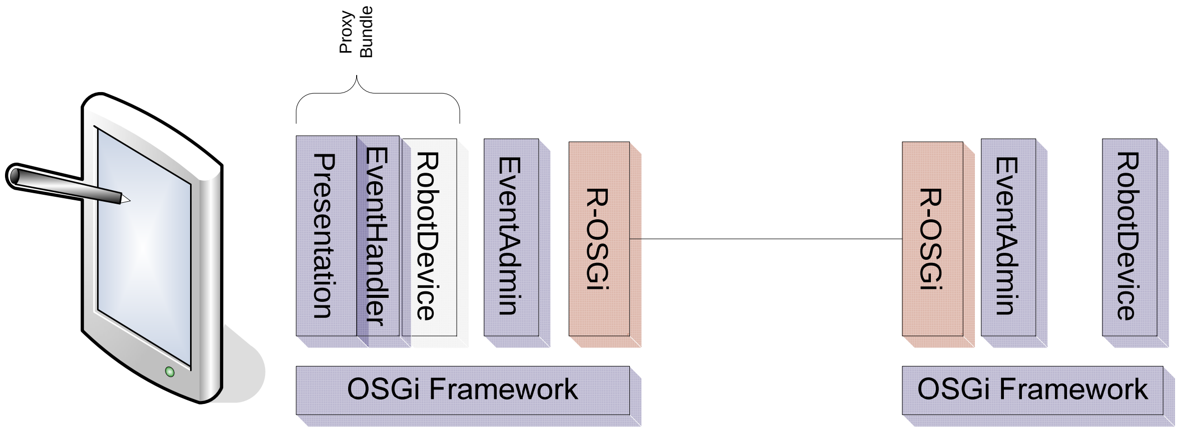
More like OS-Virtualization

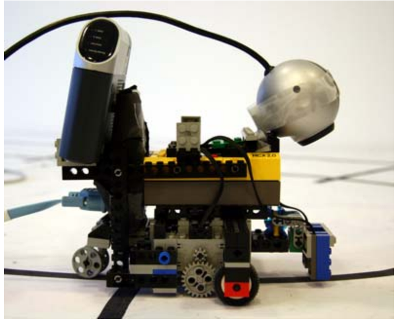
[Y. Royon, S. Frenot: *Un environnement multi-utilisateurs orienté service*. In: CFSE 2006]

Remember EclipseCon 2007…

- R-OSGi
- Originally motivated by ~~embedded~~ systems
- Service Discovery via SLP

Services were described by SLP Service URLs





[J. S. Rellermeyer, G. Alonso, and T. Roscoe: ***R-OSGi: Distributed Applications Through Software Modularization***. In: Middleware 2007]

R-OSGi Today

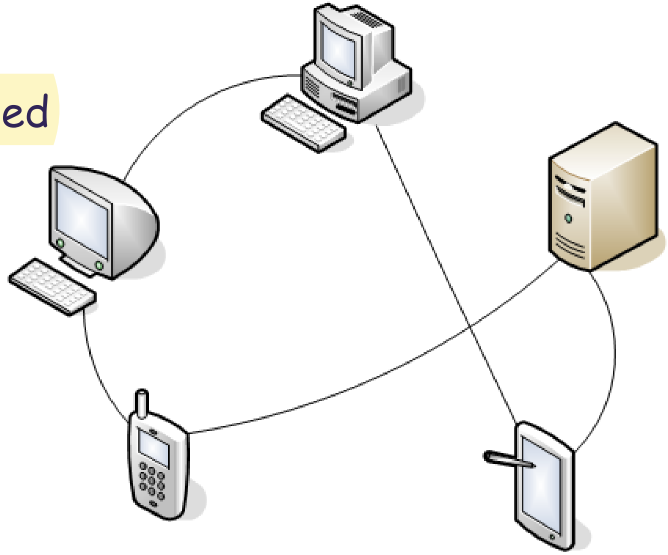
- Point to point
- Explicit connections
- Service Discovery is an optional part
- Closer to OSGi
  - `RemoteServiceReferences` known after lease exchange
  - Proxy is generated when the service is retrieved by a client

More feasible for remote services on the server side.

Gives “hints” where to connect to.

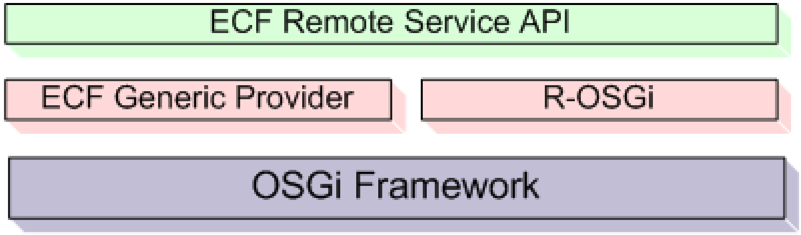
Already synchronized

Transparently “import” the remote service into the local framework



R-OSGi as an ECF Remote Service Provider

- ECF API on top of R-OSGi



ECF Remote Service API

ECF Generic Provider

R-OSGi

OSGi Framework

DSO on an ECF server

Transparent, point to point remote services

- Thereby, non-transparent access on top of a transparent service approach
- E.g., asynchronous service invocation

Motivation: Distributed Service Registry

- Original R-OSGi
  - Service Discovery
- R-OSGi 1.0.0.RCs
  - Pair wise joined Service Registry
  - Still service registry and remote service registry
- ECF Generic Provider
  - Server and DSOs

"The network is the service registry"

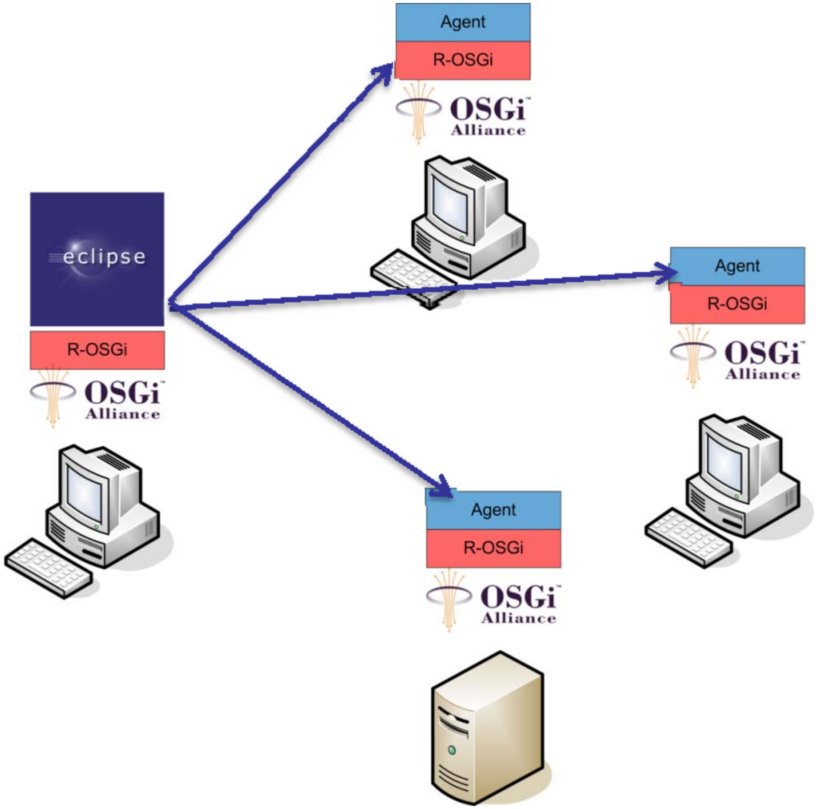
"Remote Service Registry = union of connected service registries"

"Centralized service registry"

- Other possibility: Unified service registry for both local and remote services

The Virtual OSGi Framework

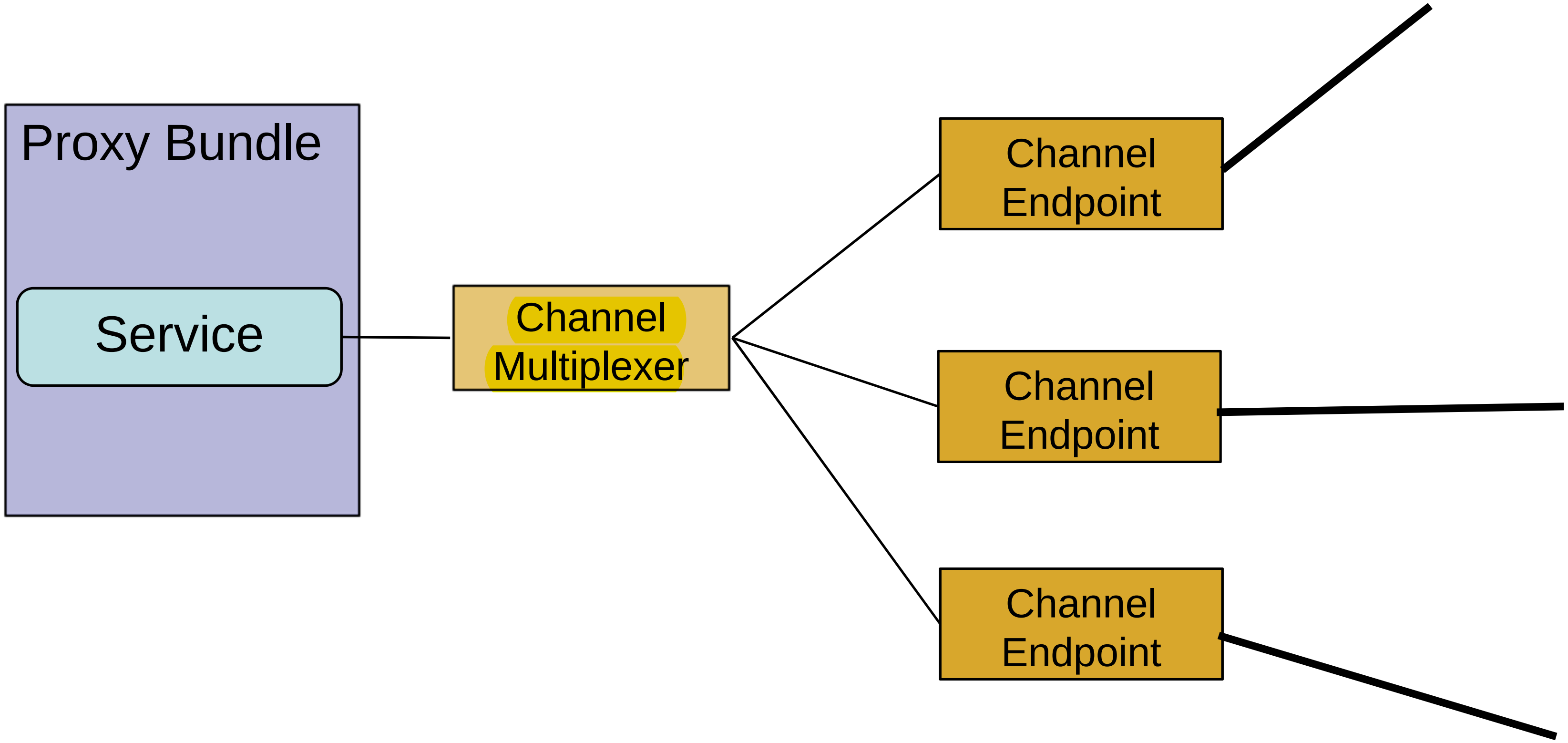
Motivation: Tool for "Orthogonal Distribution"



Demo

[Jan S. Rellermeyer, Gustavo Alonso, Timothy Roscoe: ***Building, Deploying, and Monitoring Distributed Applications with Eclipse and R-OSGi***. In: Eclipse Technology eXchange (ETX) Workshop (in conjunction with OOPSLA 2007)].

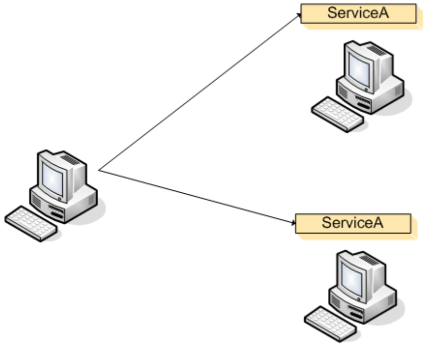
How does it work?

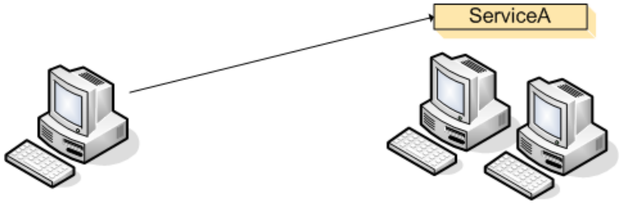


The Problem of State

- Failover
- Load balancing
- Real Services are not always stateless.

Think of the web





- Couldn’t state be preserved?
- Service replicas instead of just copies?

Task for the middleware

The Virtual OSGi Framework

Motivation: Sensor Nodes as OSGi Services

[with Michael Duller]

- TMote Sky
  - TI MSP430F1611 microcontroller at up to 8 MHz
  - 10k SRAM, 48k Flash + 1024k serial storage
  - 250kbps 2.4 GHz Chipcon CC2420 IEEE 802.15.4 Wireless Transceiver
- Cannot even run an OS
  - Runs TinyOS
- But it can be an R-OSGi service…

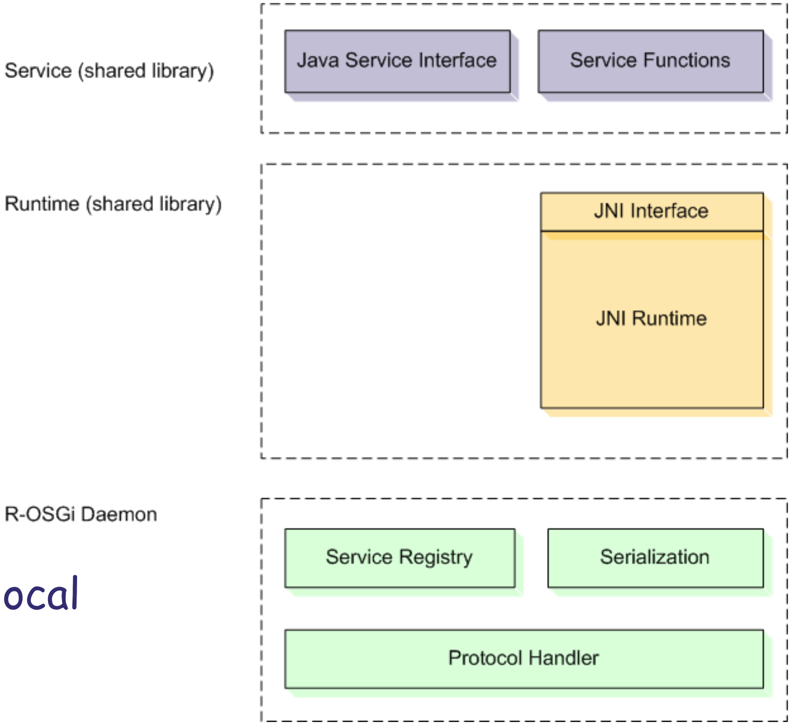
Demo

[J.S. Rellermeyer, M. Duller, and G. Alonso. ***Using Non-Java OSGi Services for Mobile Applications***. Demo at: MiNEMA 2008 Workshop in conjunction with EuroSys 2008].

Extending the Idea of OSGi Services

Why should a remote OSGi service have to be written in Java?

- C/C++
- CLDC
- Over Bluetooth, …



Would be nice to have this for local services as well

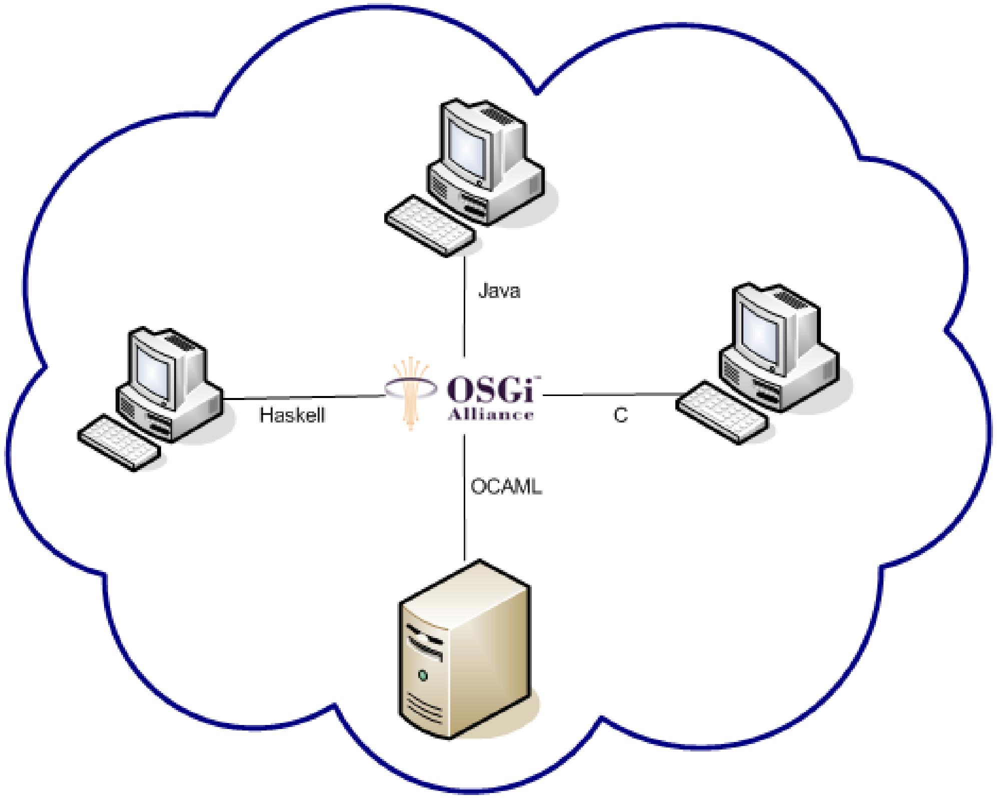
The Virtual OSGi Framework

[J.S. Rellermeyer, M. Duller, K. Gilmer, D. Maragkos, D. Papageorgiou, and G. Alonso: ***The Software Fabric for the Internet of Things***. In: Internet of Things 2008].

What about consuming services

- The consumer has to be an OSGi framework
- But, …
- Couldn't it be

The Virtual OSGi Framework



The Virtual OSGi Framework

- OSGi on the cloud
  - Have a network full of machines running OSGi
  - Don’t care where they are
  - Don’t care where bundles are installed
  - Don’t care where services are provided
  - Access them from anywhere

Bundles and services are becoming virtual

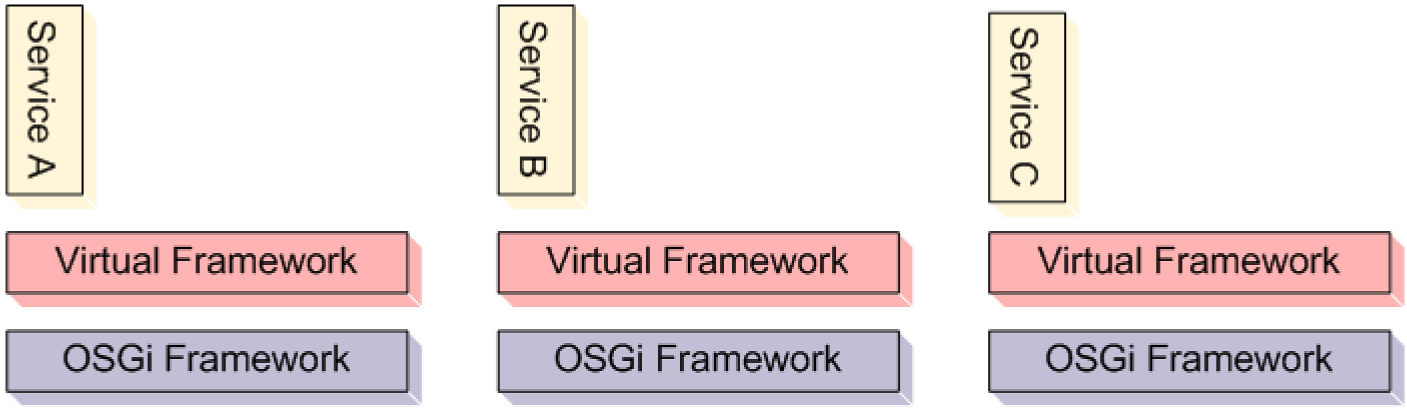
Access them transparently

- Fluid OSGi
  - Have a replica where you need it
  - Read any / write any

From a peer’s perspective, services “flow” through the network

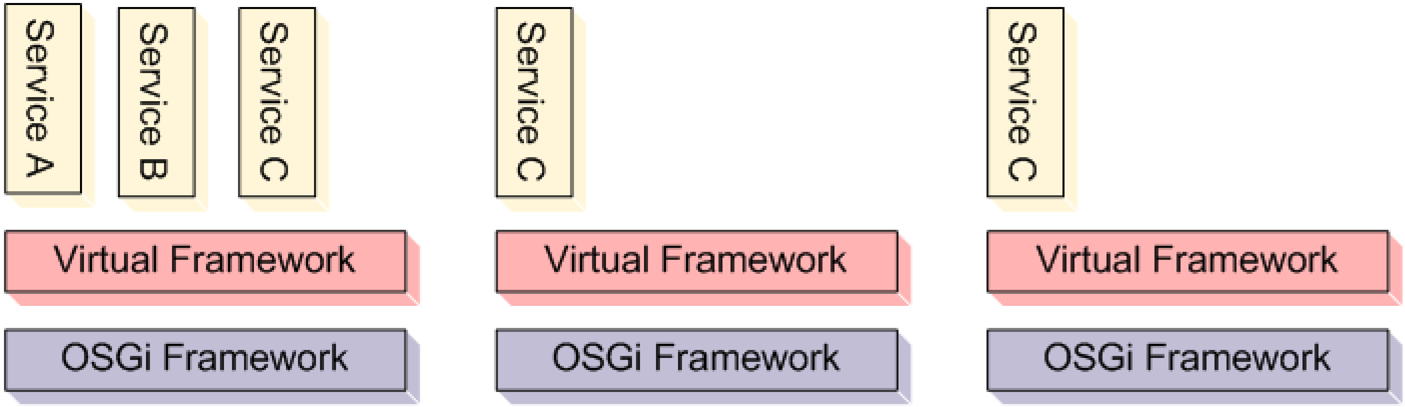
Architecture

- Unifying local and remote services
- As an extension, non-invasive against the framework



Architecture

- Unifying local and remote services
- As an extension, non-invasive against the framework



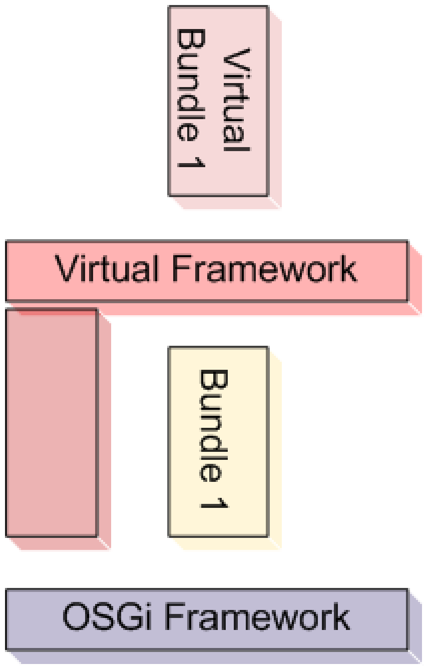
Equivalent for a peer on the cloud

Virtualized Module Layer

[Dimitrios Papageorgiou]

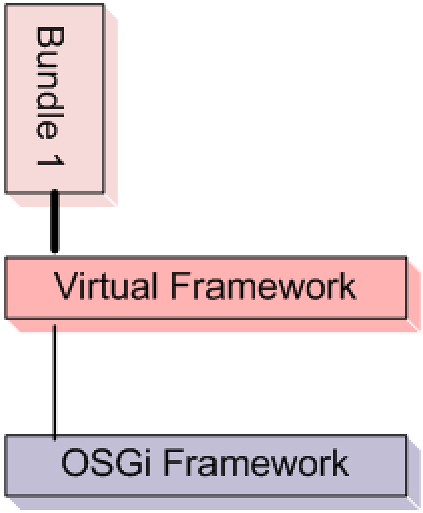
- The Virtual Framework runs as a bundle on the host framework
- Virtual Bundles are installed on the host framework
- Virtual Bundles are started on the virtual Framework

Host framework



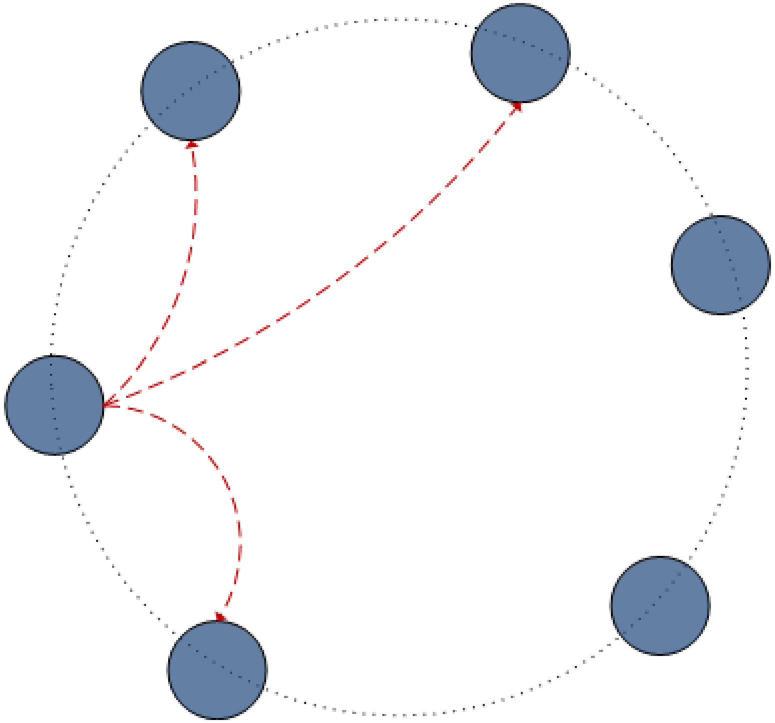
Virtual Bundles

- Installation of the bundle
  - Install on the host framework
  - Pass back a `VirtualBundle` instead of the host framework’s Bundle implementation
- Starting the bundle
  - Called through a `VirtualBundle`
  - get the Activator from the host framework
  - Call it with a `VirtualBundleContext`
  - Handle the virtual state of the bundle within the virtual framework
  - Subtile: ensure BundleID consistency



Distributed Registries

- Centralized registries are replaced by a distributed registry
- Prototype system: kind of DHT
  - Can store pointers to bundles
    - Supports constraints
  - Can store pointers to services + attributes
    - Supports filters



Challenges

- Mapping the class space model to the DHT

Optimization for resolving

- `getAllServices` becomes a very expensive operation

Is there a good Tradeoff?

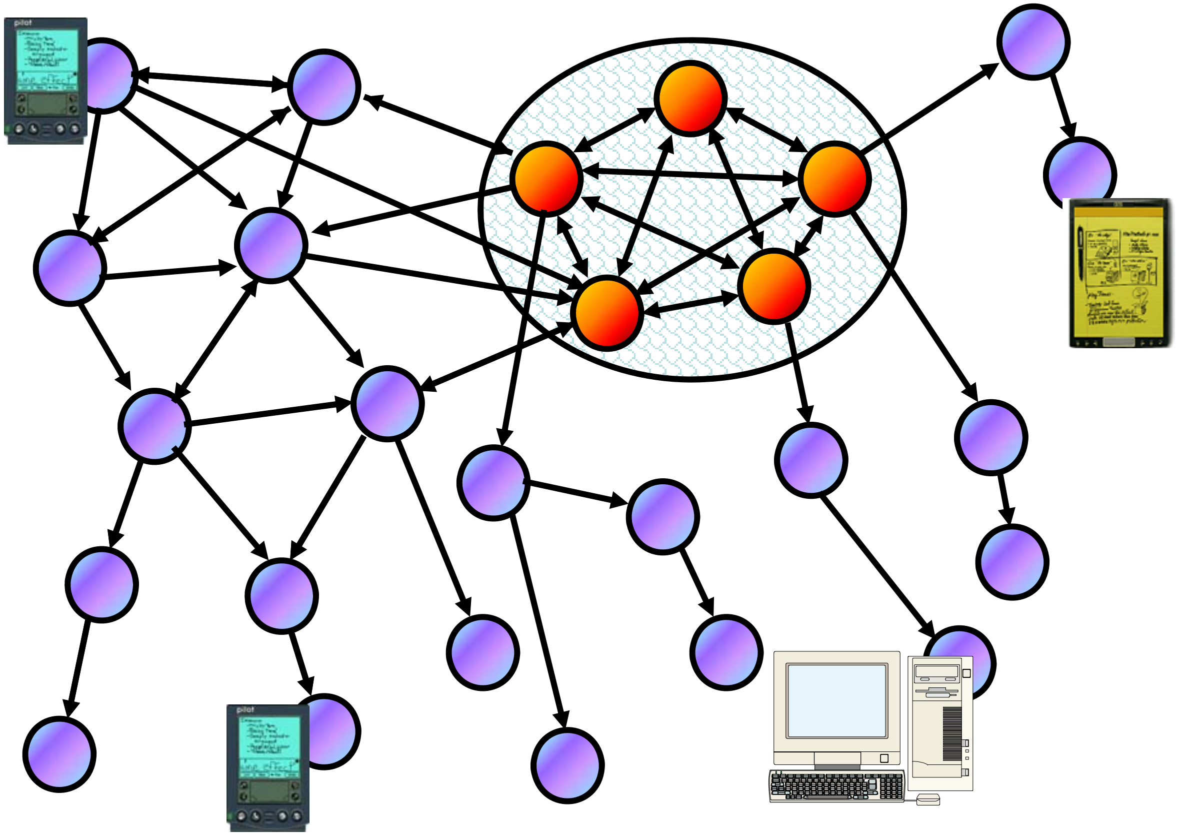
- Maintaining replicas of DHT nodes

Transactional model?

- Scalability?
  - Can it scale to massively distributed systems?
  - Can it scale to the diameter of the internet?

Currently not our focus!

OceanStore?

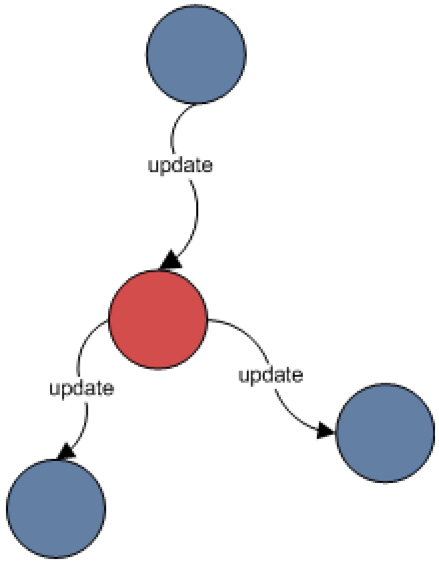


[J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao: ***OceanStore: An Architecture for Global-Scale Persistent Storage***. In: ASPLOS 2000]

Service Replication

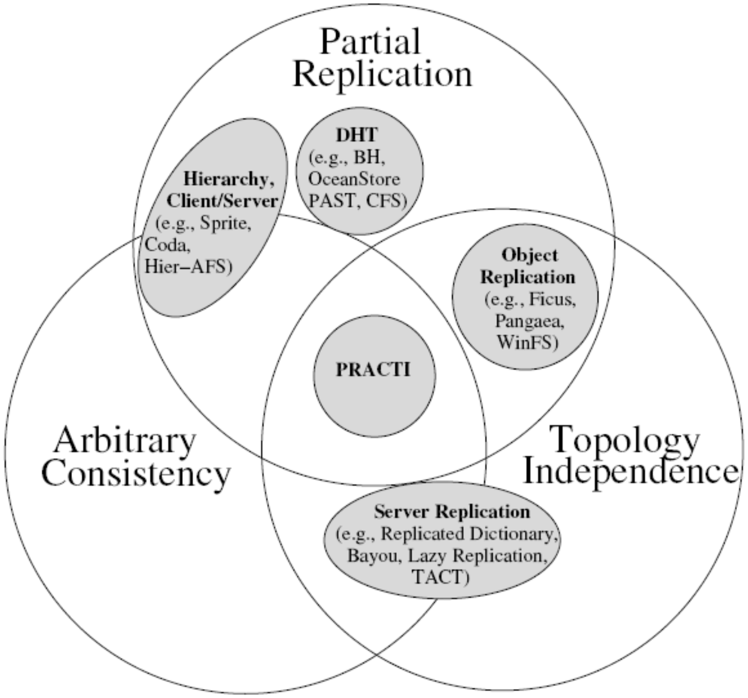
[Damianos Maragkos]

- Fluid Replication
  - Place a replica of the service where ever it is needed
- Preserve the state between service replicas
- Prototype: Communication model through the DHT
- Coordinator nodes
  - For update propagation
  - For using different consistency levels within the same virtual framework



PRACTI?

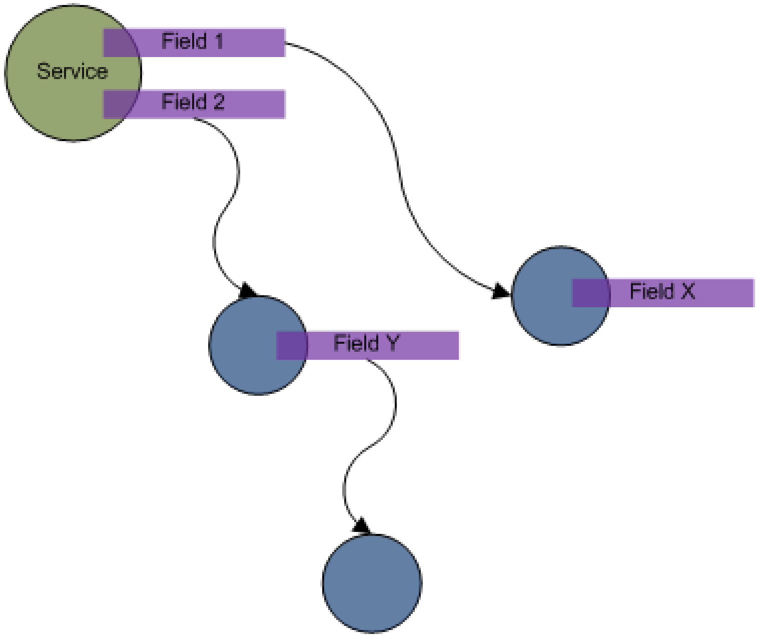
- Partial Replication
- Arbitrary Consistency
- Topology Independence


Partial Replication
Hierarchy, Client/Server (e.g., Sprite, Coda, Hier–AFS)
DHT (e.g., BH, OceanStore PAST, CFS)
Object Replication (e.g., Ficus, Pangaea, WinFS)
PRACTI
Arbitrary Consistency
Topology Independence
Server Replication (e.g., Replicated Dictionary, Bayou, Lazy Replication, TACT)

[N. Belaramani, M. Dahlin, L. Gao, A. Nayate, A. Venkataramani, P. Yalagandula, and J. Zheng: ***PRACTI Replication***. In: NSDI 2006]

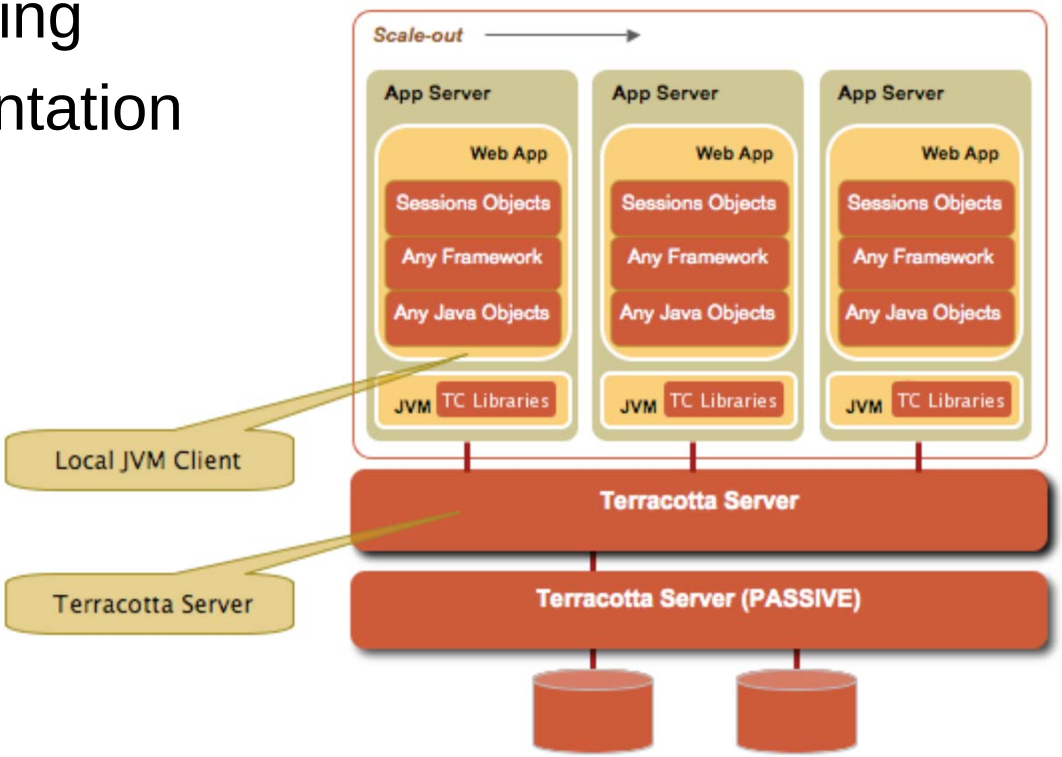
Replication of Services

- What is state?
  - Model: Only services have state
  - State is contained in fields
- Capturing state?
  - Update propagation
  - Arbitrary consistency
- Goal: Transparent replication
  - Run with every OSGi Service
  - Requires no changes



OpenTerracotta?

- Transparent clustering
- Load time instrumentation
- Distributed locking



[http://www.terracotta.org]

Instrumentation

- Symbolic Execution

Find out where state is accessed/changed

- Instrumentation to capture fields

P2P update propagation through group communication

- Also does distributed locking, distributed thread coordination

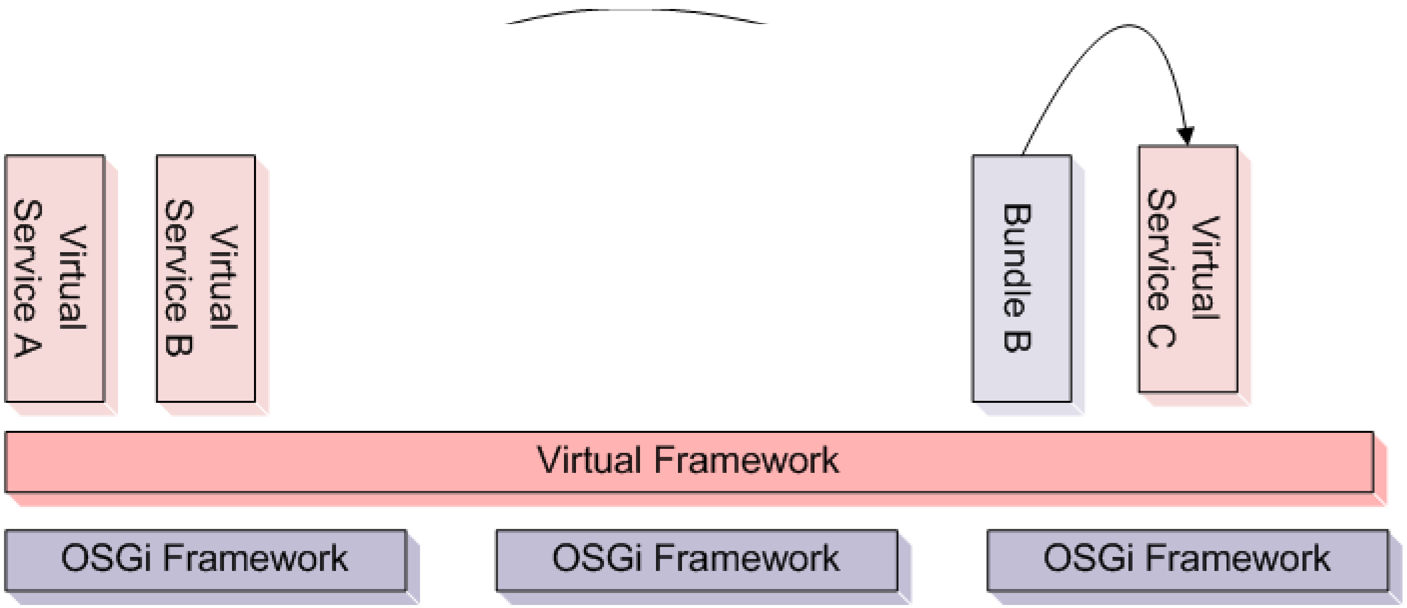
Seamless parallelization

- Also used for service migration

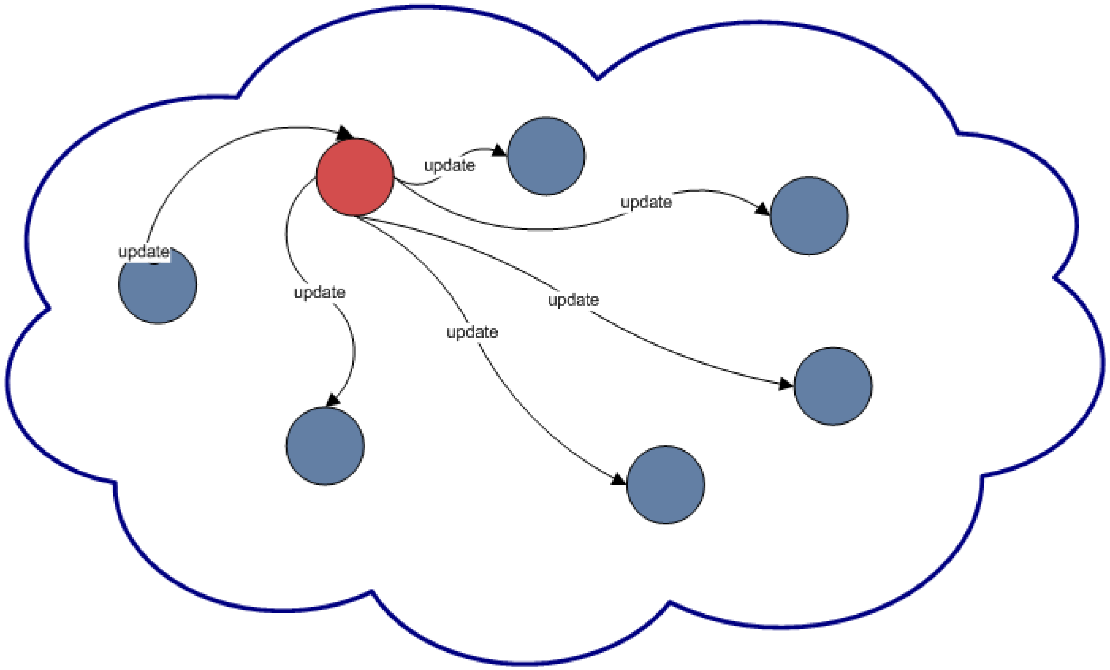
Can be considered as a temporal replication

But we also handle thread migration.

What we have now…



Coordination overhead

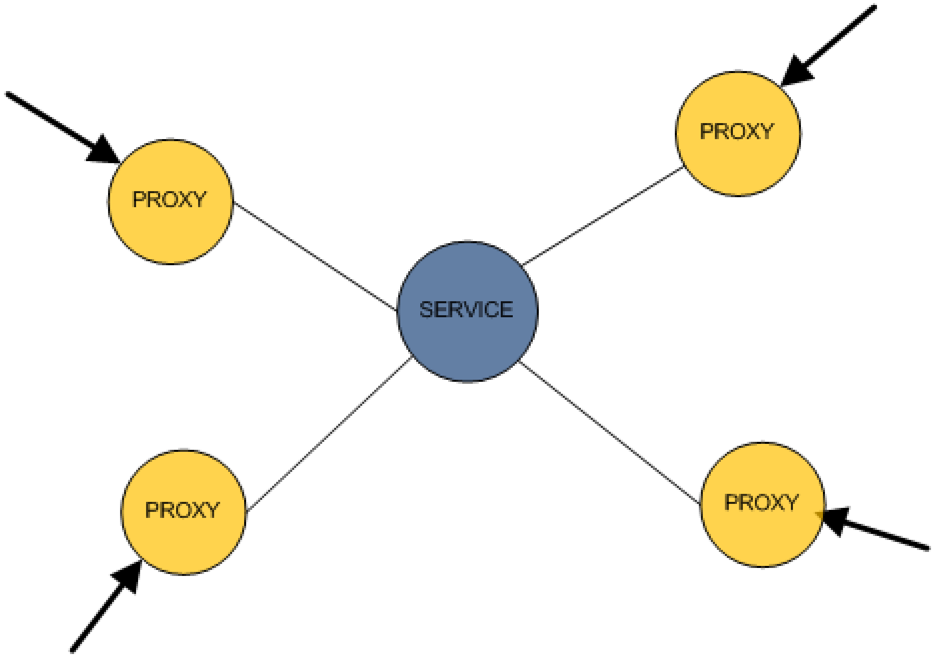


Duality between Service Replicas and Remote Services

- Coordinating all the replicas
- Affects scalability

Outlook: Autonomous Controller

[Ramon Küpfer, Dario Simone; starting soon]

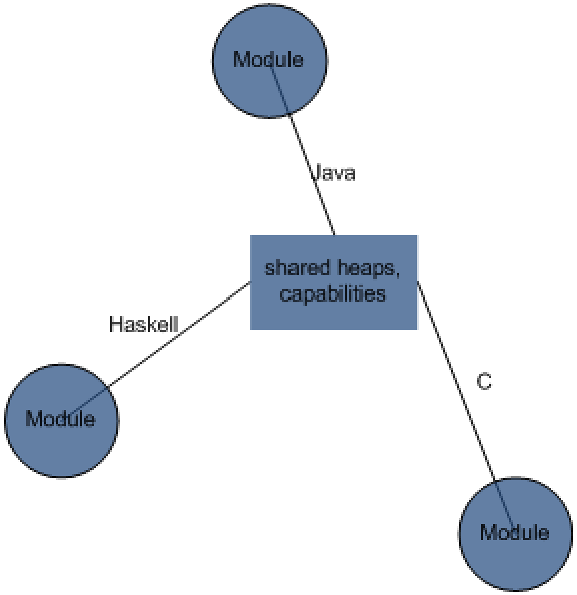


Outlook: Language-independent OSGi

- Soon a prototype for Barrelfish (new operating system at ETH)

[with Simon Peter, Adrian Schüpbach, Andrew Baumann, Timothy Roscoe]

  - Use the kernel-provided IPC model
  - Provide an application model (derived from OSGi)
  - Optimize for interactions within the same language
  - Provide generic type mappings for heterogeneous apps



Conclusions

- The Virtual OSGi Framework
  - Unifies local and remote services
  - Makes a (dynamic) group of machines appear as a single OSGi framework
  - Allows replication of services for load balancing or to increase failure resilience
  - Runs as a bundle on top of every framework
  - Uses the host framework for module layer operations
  - Intercepts/extends certain operations on the virtualization layer
  - Can relocate bundles/services
  - OSGi on the cluster/cloud

Welcome to the virtual world!

- Questions?