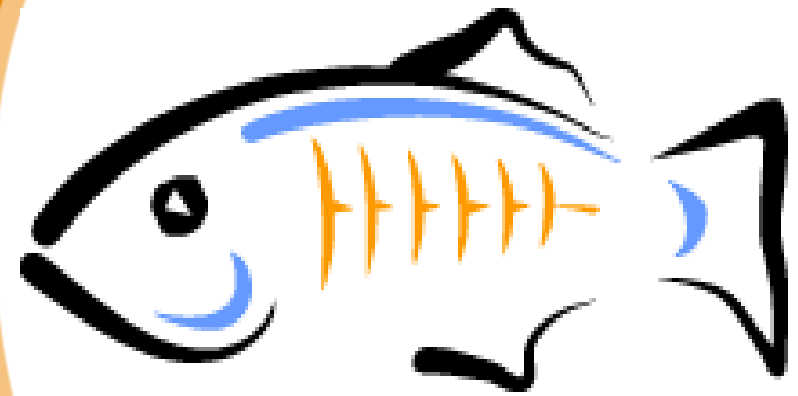


Opensourcowe projekty GlassFish oraz OpenSSO

Marek Sokołowski
Sun Microsystems Poland





Project GlassFish

Project
GlassFish



Marek Sokołowski
Sun Microsystems Poland

Java EE 5.0 = (J2EE 1.4).next

- Java EE 5 Theme: Ease of Development
- POJO-based programming
 - > More freedom, fewer requirements
- Extensive use of annotations
 - > Reduced need for deployment descriptors
 - > Annotations are the default
- Configure by exception
 - > Reasonable defaults wherever possible
- Resource Injection
- New APIs and frameworks
 - > EJB3, JAXB 2, JAX-WS, JSF, ...

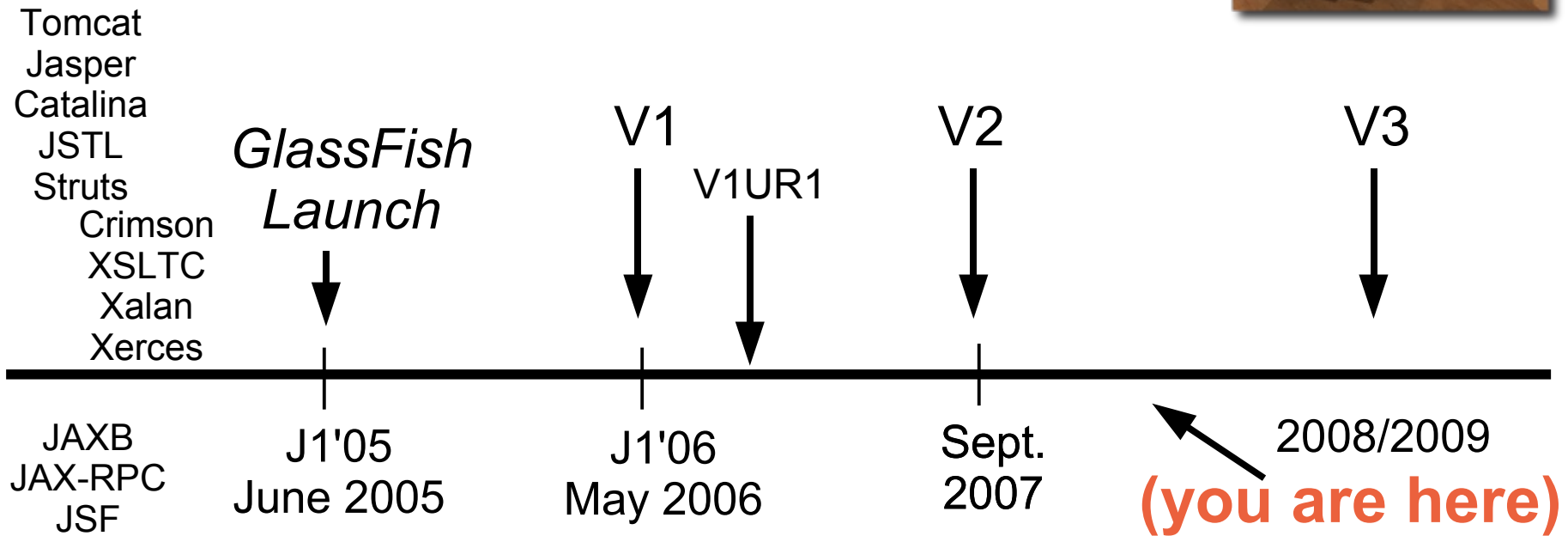
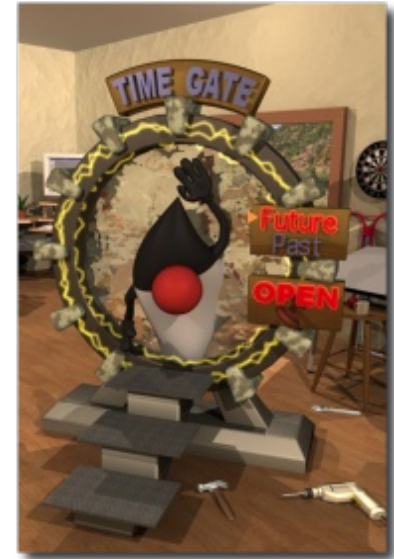
GlassFish



What Is Project GlassFish?

- A Java EE 5 compliant Application Server
- Enterprise Quality
 - > GlassFish Application Server 9.x
 - > Use it in production!
- Open Source
 - > CDDL (like OpenSolaris, NetBeans)
 - > GPLv2 (like Java and NetBeans)
 - > Use it in production!
- Community at <http://glassfish.java.net>
 - > Sources, bug DBs, discussions at Java.Net
 - > Roadmaps, Architecture Documents

Timeline of Project GlassFish



Releases in Project GlassFish

- GlassFish v1
 - > Victory! Java EE 5 Compliance!
 - > December 2006 : UR1 - bug fixes
 - > Growing # of Deployments
- GlassFish v2
 - > New WS stack, performance, startup time
 - > Cluster management, load balancing, failover
 - > Some scripting support
 - > Community, Transparency, Adoption
- GlassFish v3
 - > Modularized kernel (HK2)
 - > The Web 2.0 engine



GlassFish v2

SJS Application Server 9.1

- Metro Web Services Stack
 - > Performance, Microsoft interoperability
- Clustering, Load-Balancing, HA
 - > Advanced Management
- JBI support (OpenESB 2.0)
- Better user experience
 - > Single, smaller, download
 - > Multiple User Profiles
 - > Better startup time
 - > Update Center
 - > New admin console: JSF, AJAX, Charts

Available!



GlassFish v2: Grizzly & JSP Containers

- JSP Container
 - > Can use JSR-199 (Javac APIs in Java 6)
 - > 10x performance improvement
- Grizzly
 - > Improved over GlassFish v1
 - > Very Flexible and Customizable
 - > Non-blocking SSL
 - > Support Quality of Service constraints
 - > Scalable Async Req Processing (ARP)
 - > Supports *Comet* (long-term HTTP connections)
 - > Used in Jetty, etc...
- Hosting features
 - > Alternate docroots, webcontainer dynamically reconfigurable, ..

Management Features

- Centralized, secure, remote access
 - > Accessible as GUI, CLI, IDEs, Java-based programs
 - > Also available via provided ANT tasks
 - > JMX & Application Server Management eXtensions, AMX
 - > Can be monitored through *jConsole* and others
- Per-service monitoring levels
- Call Flow
- Self Management
- Resource consumption management

Metro

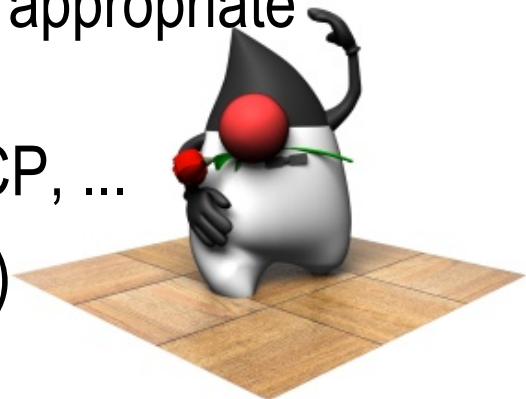
JAX-WS, WSIT, and JAXB implementations

- Dynamic JAX-WS Runtimes
- JAXB 2.0 fully support XML Schema
 - > A lot of reuse of GlassFish's JAXB implementation
- GlassFish JAX-WS 2.x
 - > WS Separation of Transport and Encoding
 - > HTTP, JMS, SMTP, TCP/IP
 - > MTOM, Fast Infoset (binary), Textual, Others
- WSIT (Project Tango)
 - > Microsoft Interoperability and Quality of Service
 - > Same (old) JAX-WS programming model
- Overall **Great** Performance (see next slide)



WSIT (Project Tango)

- WSIT (Web Services Interoperability technology)
 - > Extension to JAX-WS 2.x, part of GlassFish v2
 - > Implements WS-Addressing, WS-Security, WS-Secure Conversation, WS-ReliableMessaging, WS-MetaData Exchange, MTOM/XOP, WS-Policy, and more
- POJO programming model from JAX-WS 2.x
 - > Annotations, Asynchronous Web Services, ...
 - > No API, platform is responsible for providing appropriate QoS
 - > Alternate Transports – JMS, SMTP, FTP, TCP, ...
- Common work with Microsoft (.Net 3's WCF)



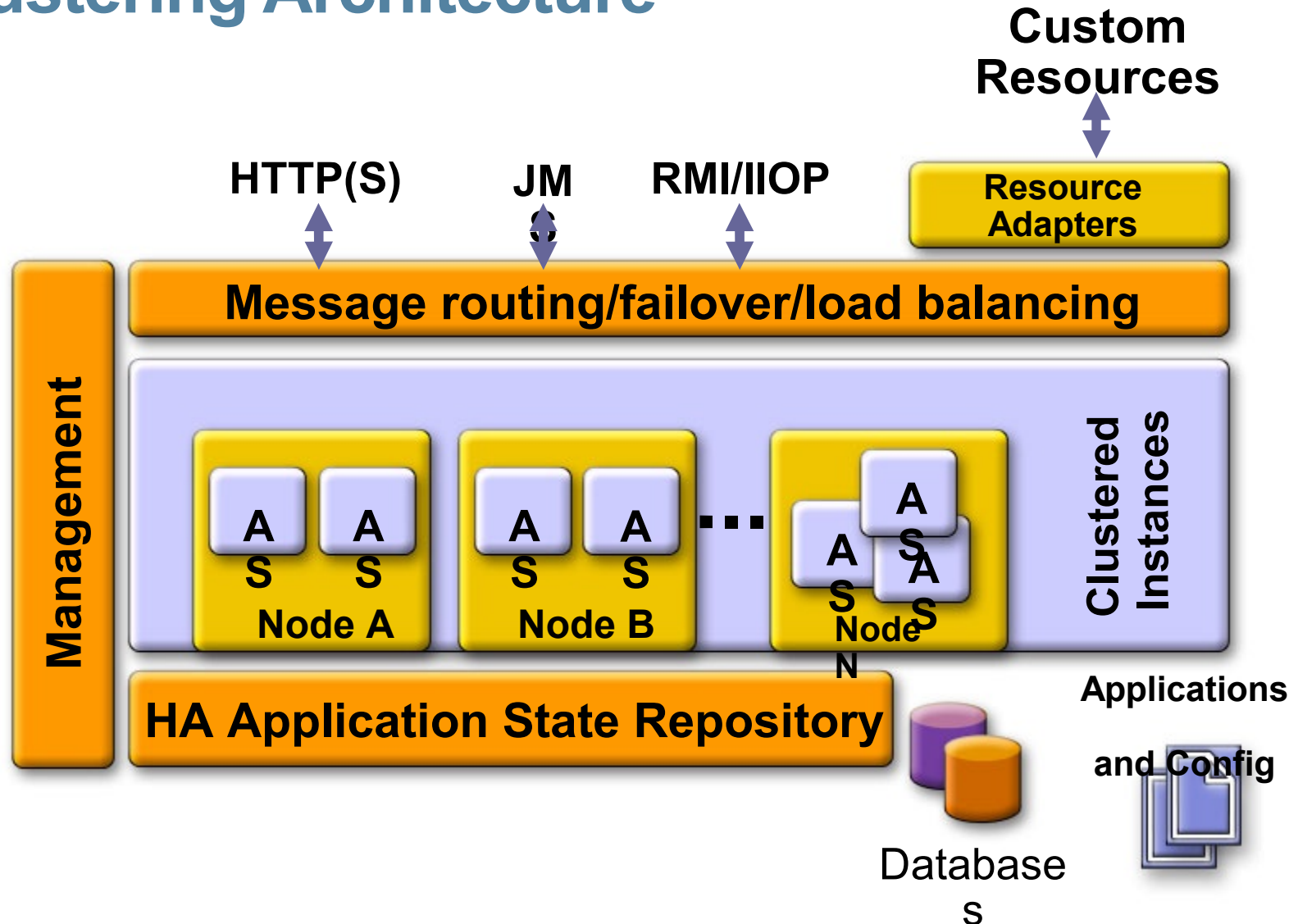
Top Link Essentials / JPA

- JPA now a separate JSR
 - > 1.0 very well accepted
 - > Does not require a container
- Oracle Contribution to GlassFish
 - > Fully JPA-compliant and open source
- Very Active Community
 - > Oracle, Sun, TmaxSoft, independents
 - > Mail: persistence@glassfish.dev.java.net
- Pluggable (per spec)
 - > In GlassFish, JEUS, JOnAS, Tomcat, Geronimo, JBoss, Oracle
 - > Converse is true also: Hibernate & OpenJPA run on GlassFish

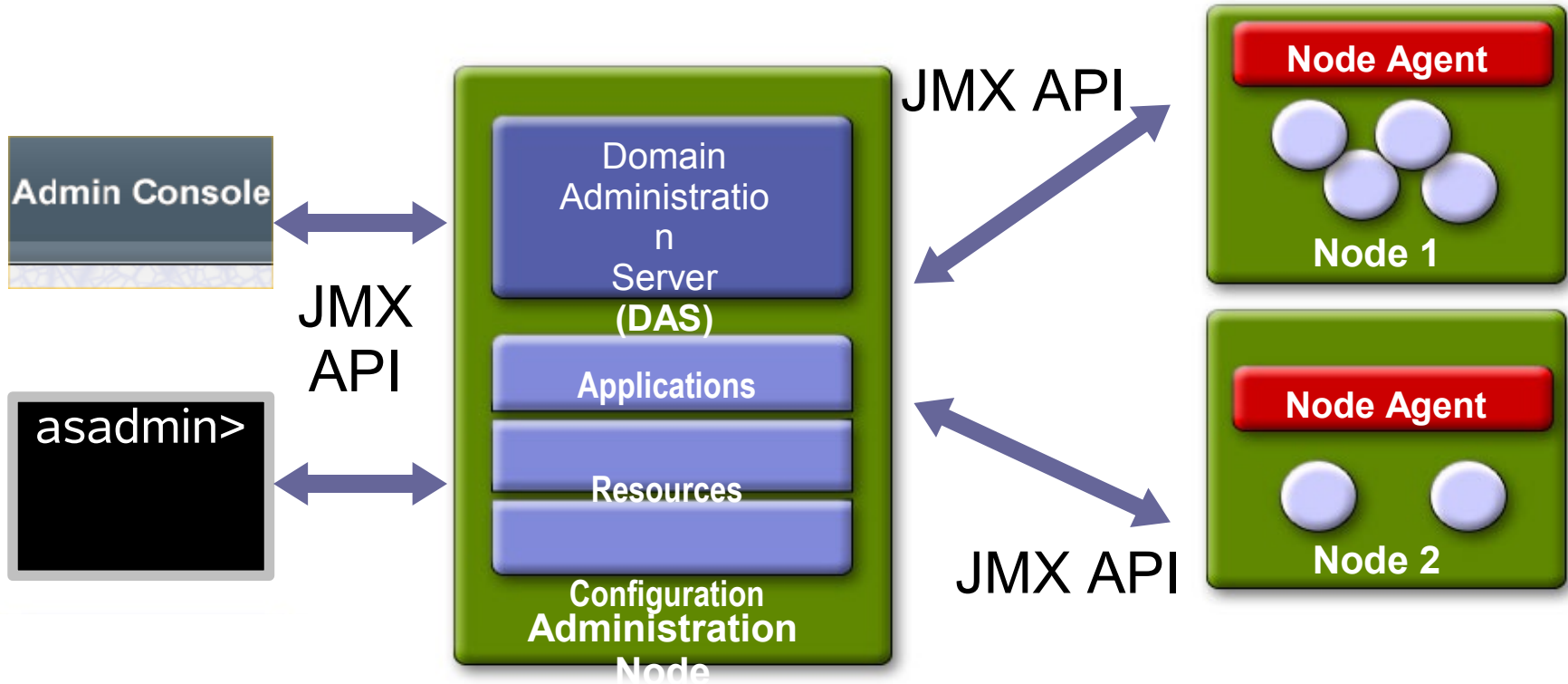
JBI – Java Business Integration

- OpenESB 2.0 implementation
 - > Included in GlassFish v2
 - > Integrated as a life-cycle module
 - > Integrated admin tools (Web and CLI)
- Many components
 - > Binding Components: HTTP, File, FTP, JMS, TCP, CICS, HL7, ...
 - > Service Engines: BPEL, XSLT, ETL, SQL, Scripting, Worklist, ...
 - > <https://open-esb.dev.java.net/Components.html>
- Tools support
 - > NetBeans SOA 6.0 Beta
- Possible to plug ServiceMix into GlassFish v2
 - > Support wider JBI story

Clustering Architecture



Clustering in GlassFish v2



Java EE
Server
Instance

JMX = Java Management Extensions



Dynamic Clustering and In-Memory Replication

- GMS with Project Shoal
 - > <http://shoal.dev.java.net>
 - > Dynamic clusters implemented with JXTA by default
 - > Extreme ease of use in cluster setup
- Replication
 - > What?
 - > HTTP session state
 - > Stateful EJB session state
 - > Single Sign-On state
 - > Container state (timers, ...)
 - > How?
 - > Default is In-Memory replication with JXTA
 - > Can still use HADB for 99.999% uptime (higher perf degradation)

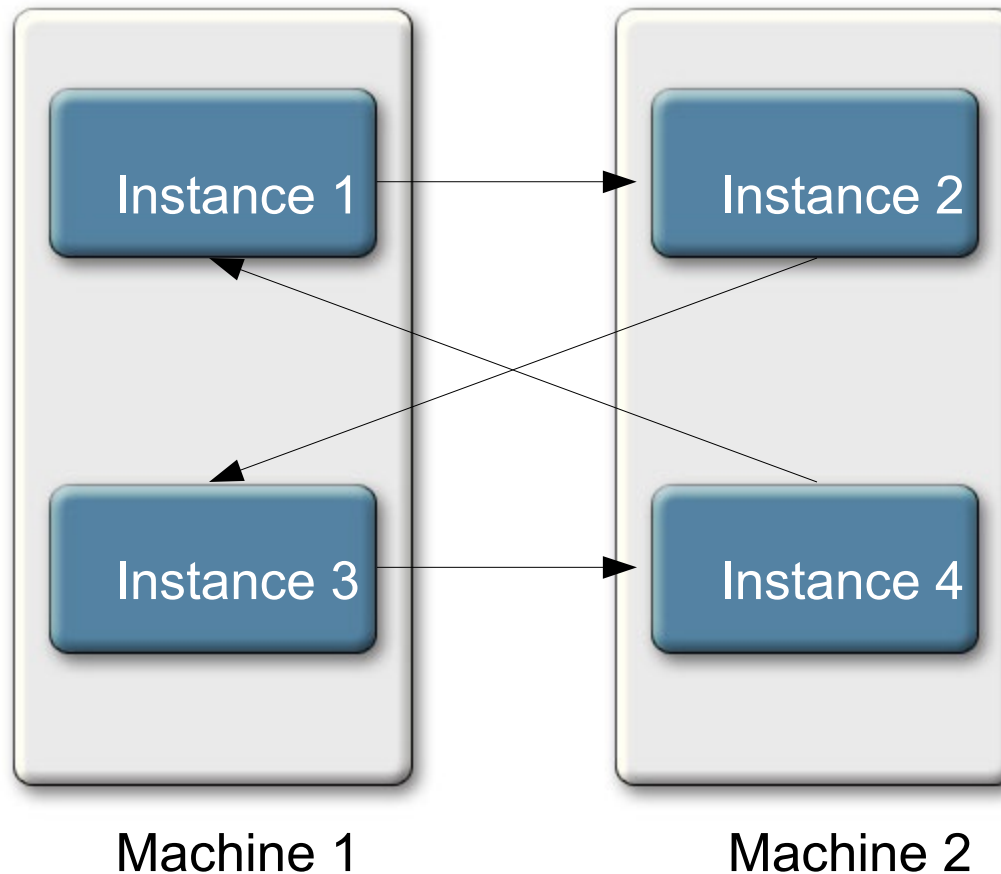


Shoal

Memory Replication

Typical cluster topology

Example: Maximize Availability on 4 node cluster on 2 machines



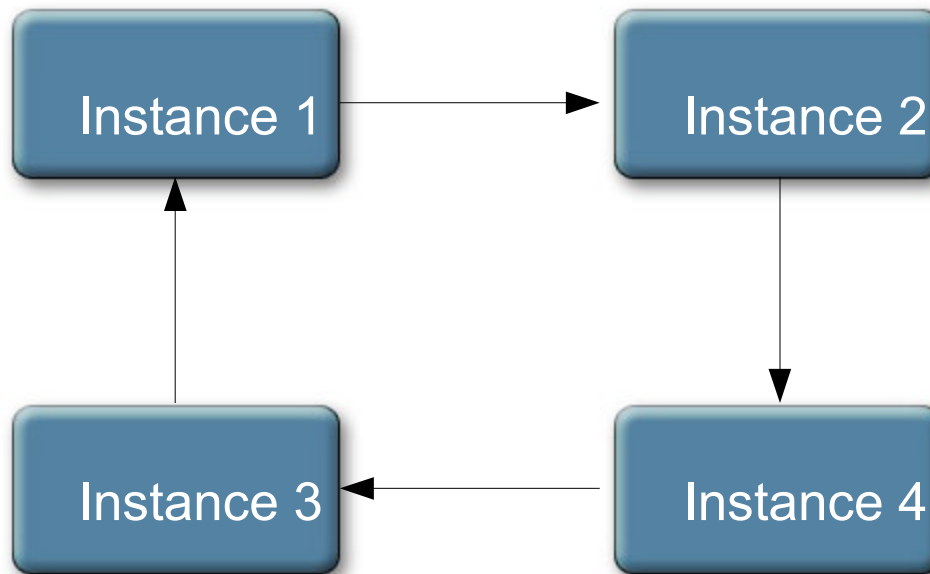
Memory Replication

Problem domain and scope

- Provide Application High Availability for :
 - > HTTP session state
 - > Single Sign On state
 - > Stateful EJB Session bean state
- Application server instances need :
 - > Automatic Transaction recovery
 - > EJB Timer migrations
 - > Cluster health
 - > In-memory replication module
 - > IIOP failover load-balancer

Memory Replication

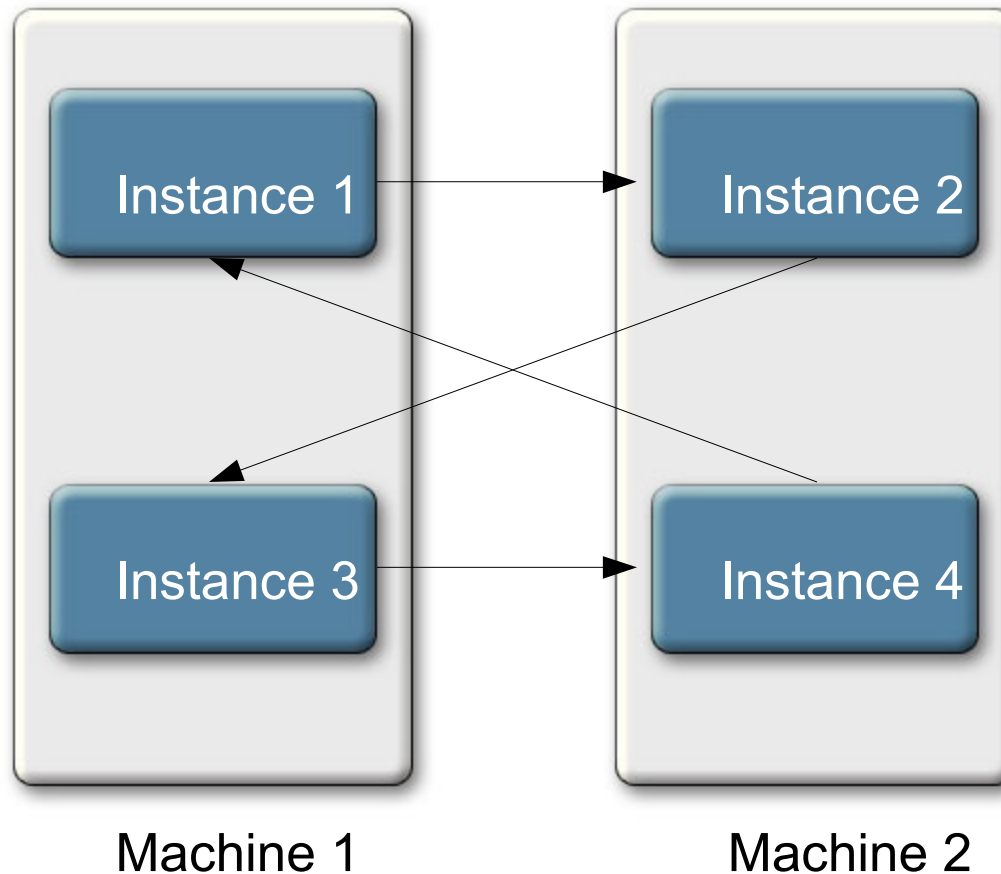
Typical cluster topology



Memory Replication

Typical cluster topology

Example: Maximize Availability on 4 node cluster on 2 machines

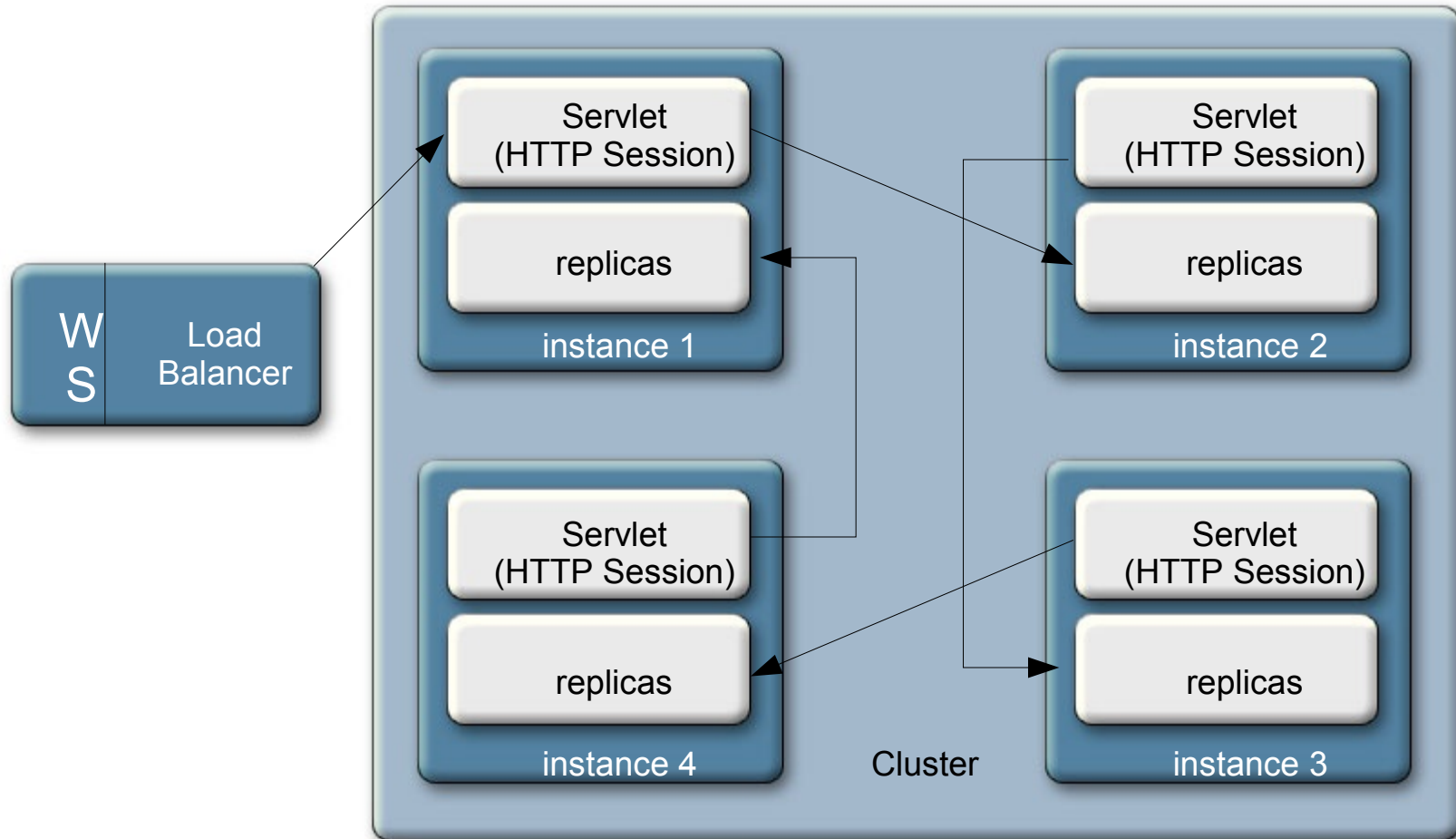


Memory Replication

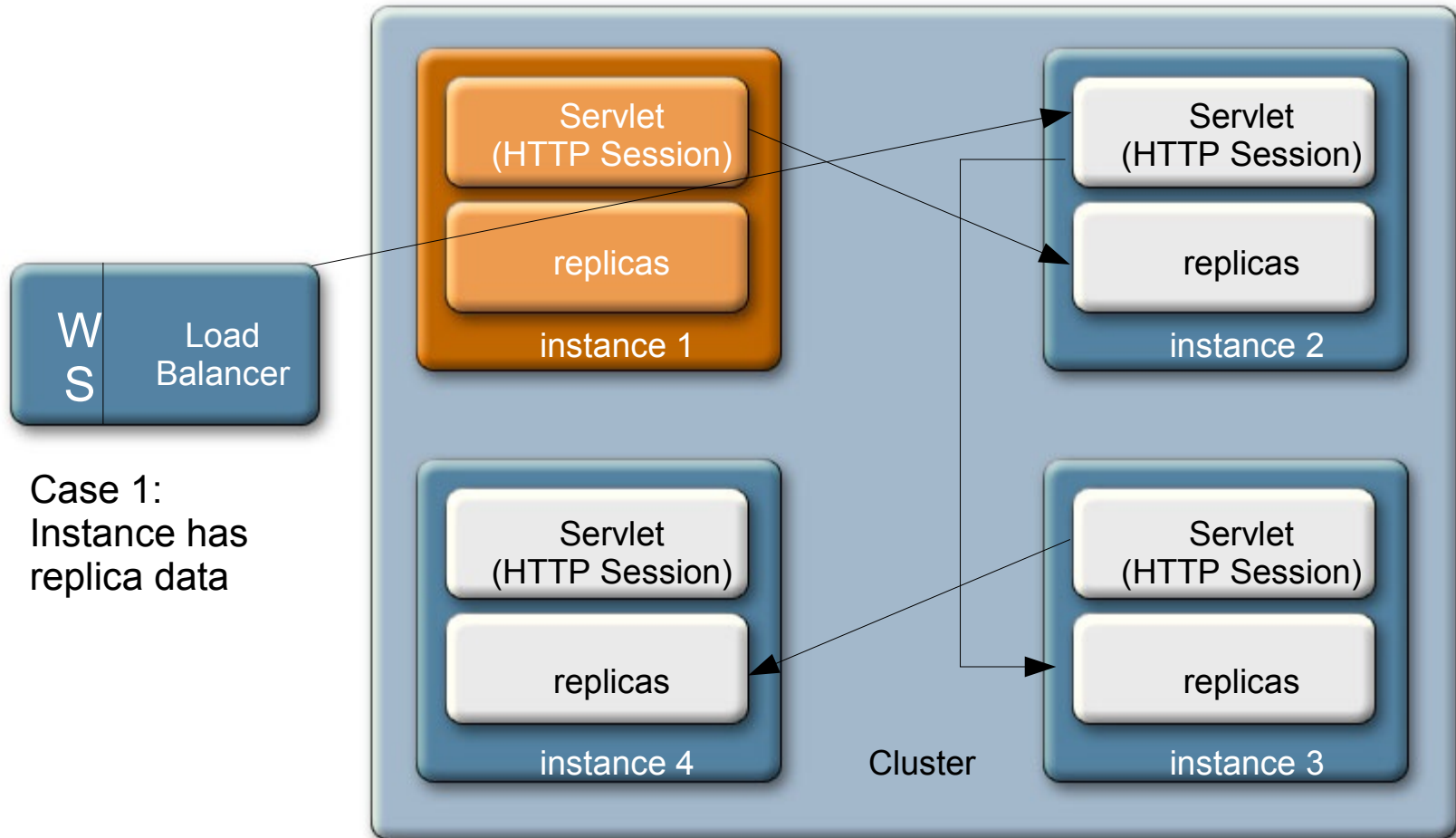
Typical failover scenario

- Location transparency
 - > fail over request can go to any instance in the cluster
- 2 Cases: Failover request lands on
 - > Case 1: instance with replica data: ownership taken, processing continues
 - > Case 2: instance without replica data
 - > instance sends broadcast request
 - > instance with replica data transfers data back to requester and deletes its copy after an acknowledge
 - > JXTA makes this easy (propagation communication channels are scoped within the “group” (i.e. the cluster members))

HTTP Session State Failover

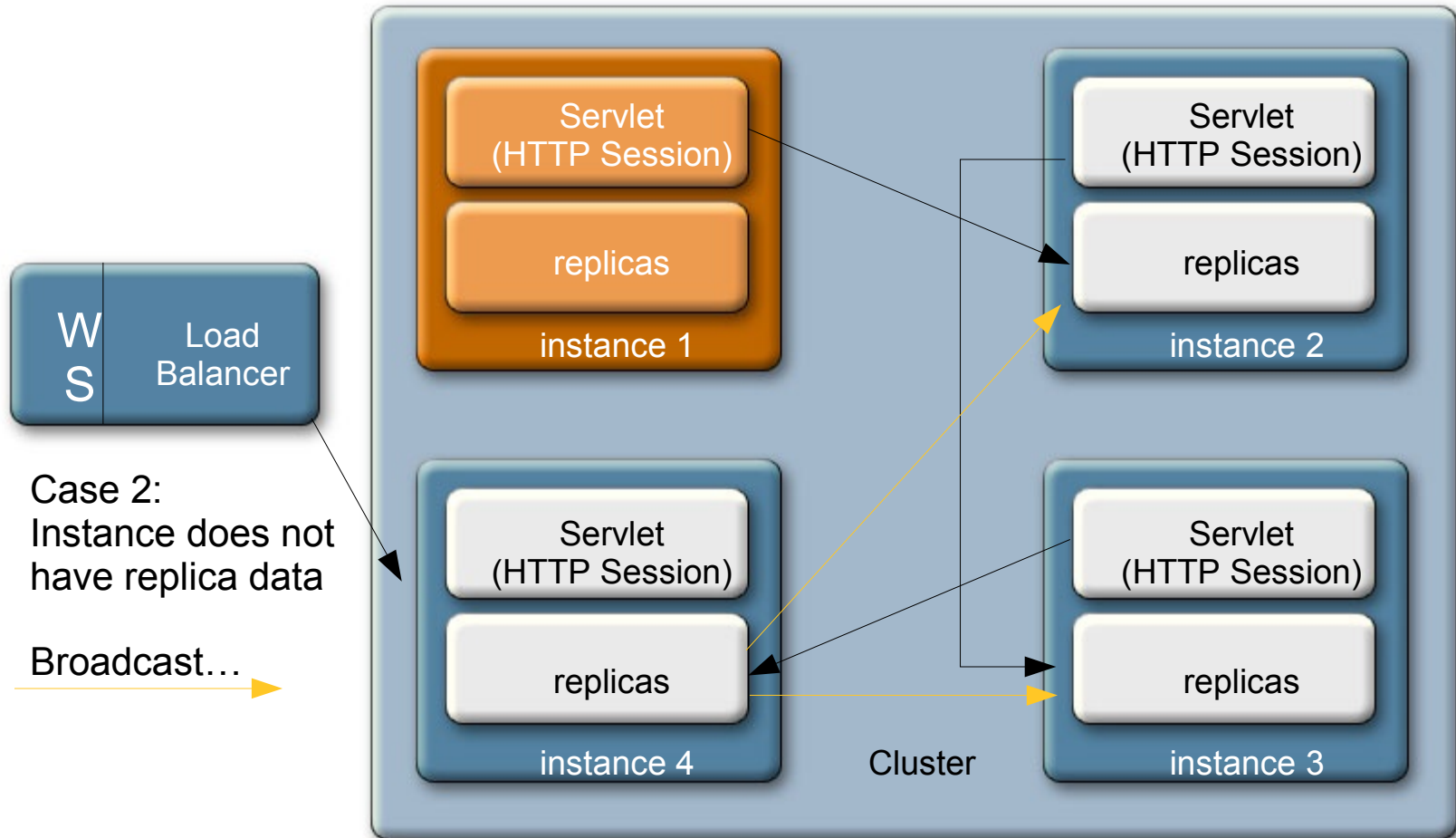


HTTP Session State Failover

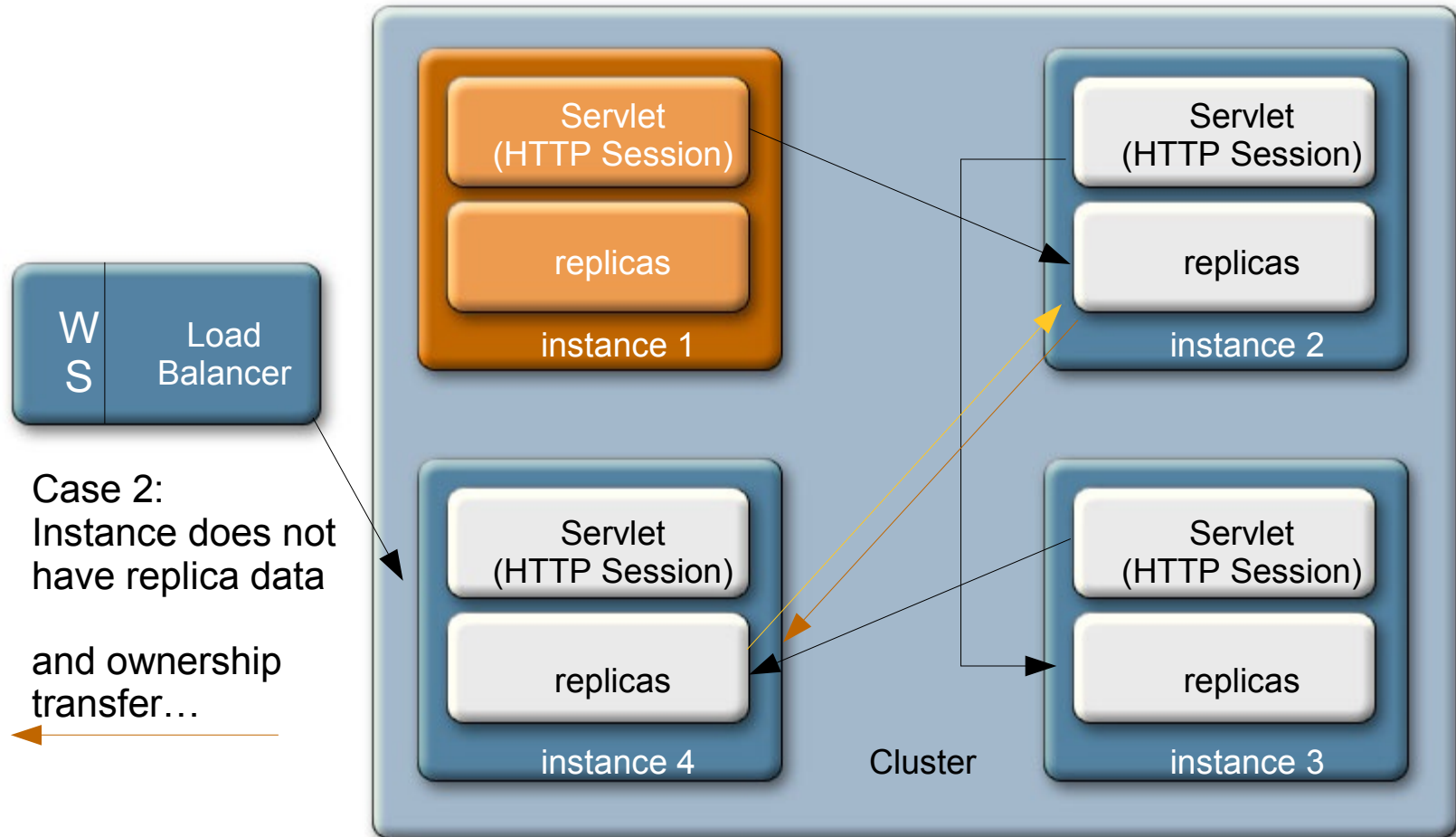


Case 1:
Instance has
replica data

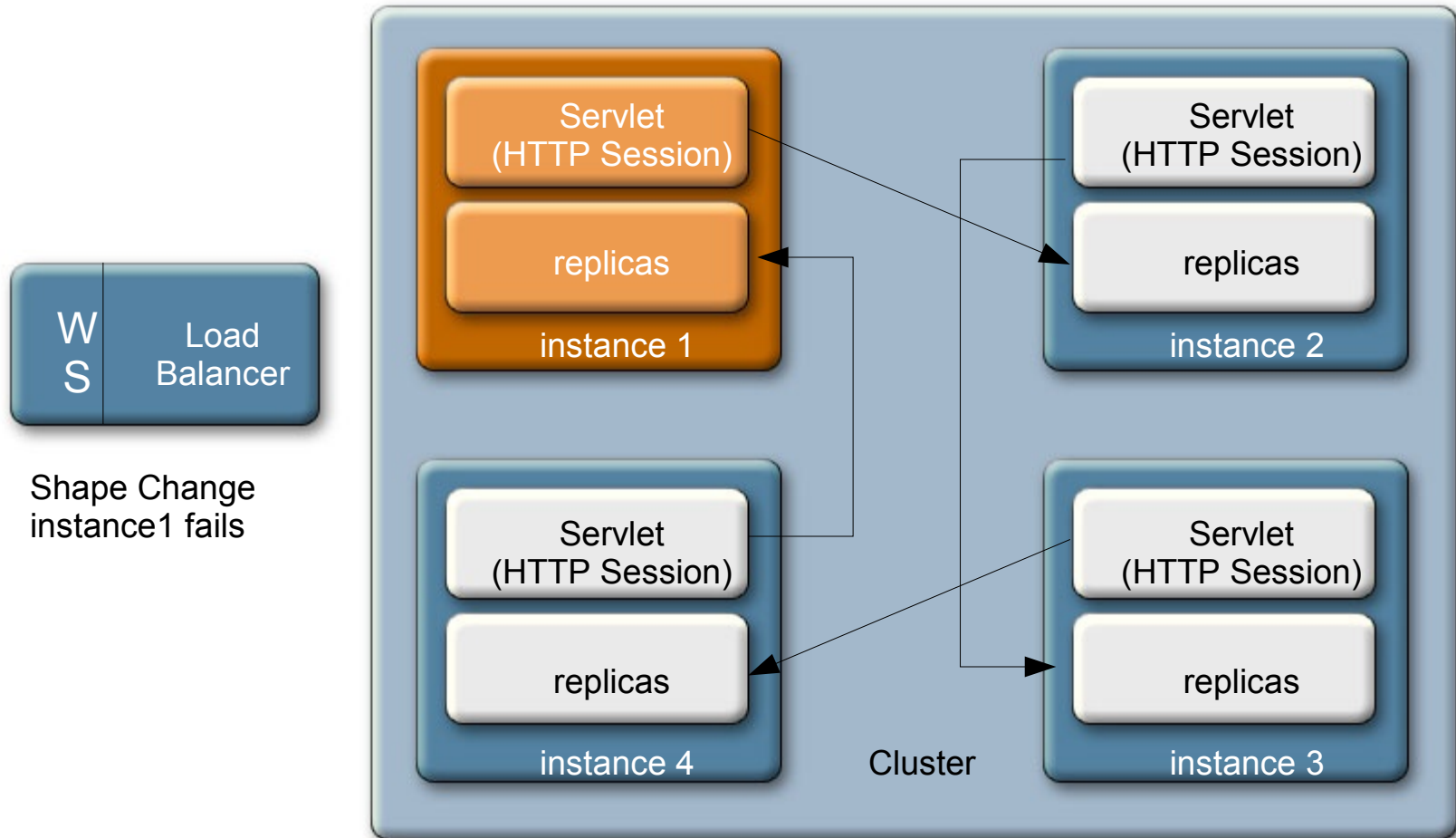
HTTP Session State Failover



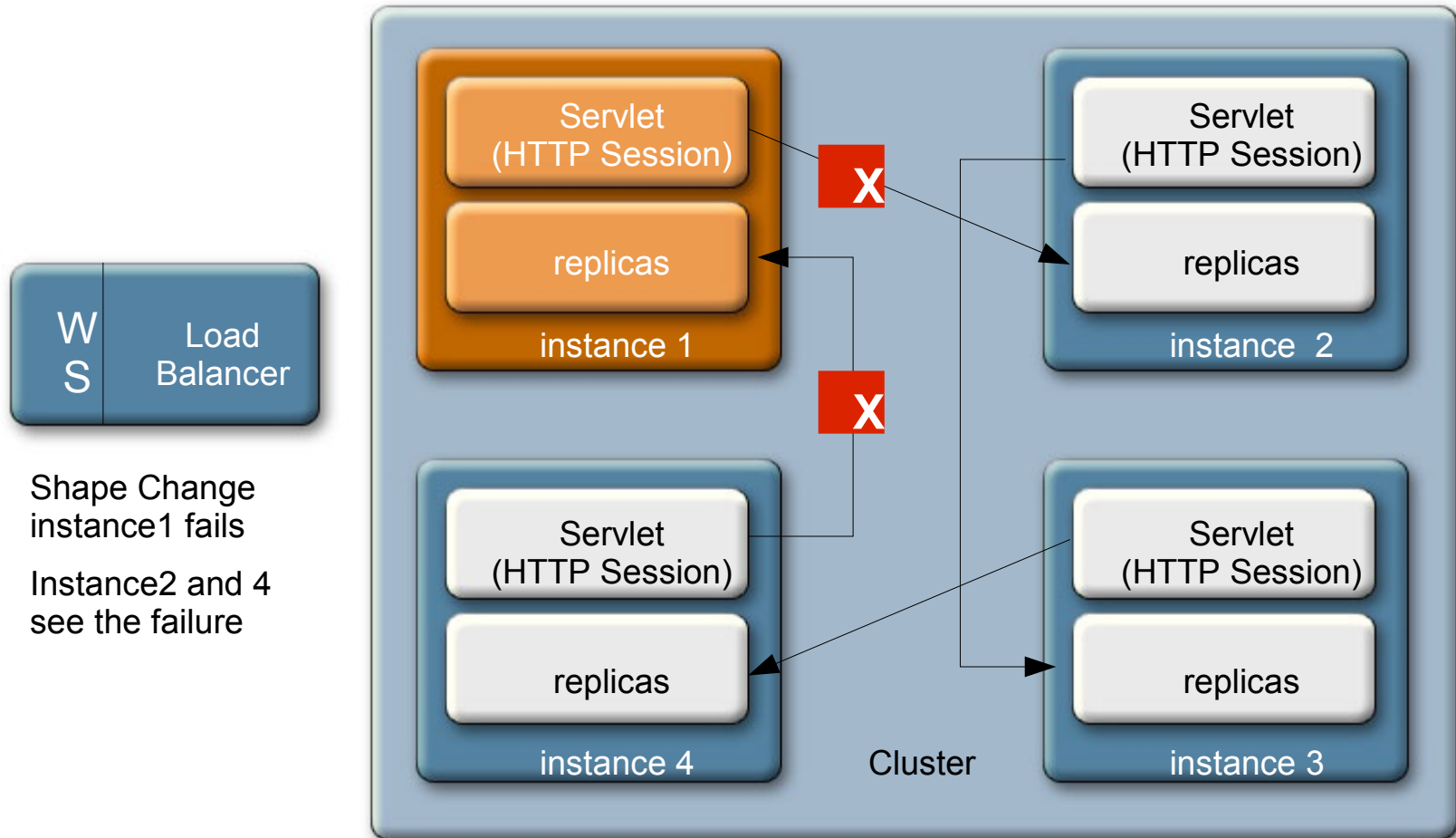
HTTP Session State Failover



Cluster Dynamic Shape Change



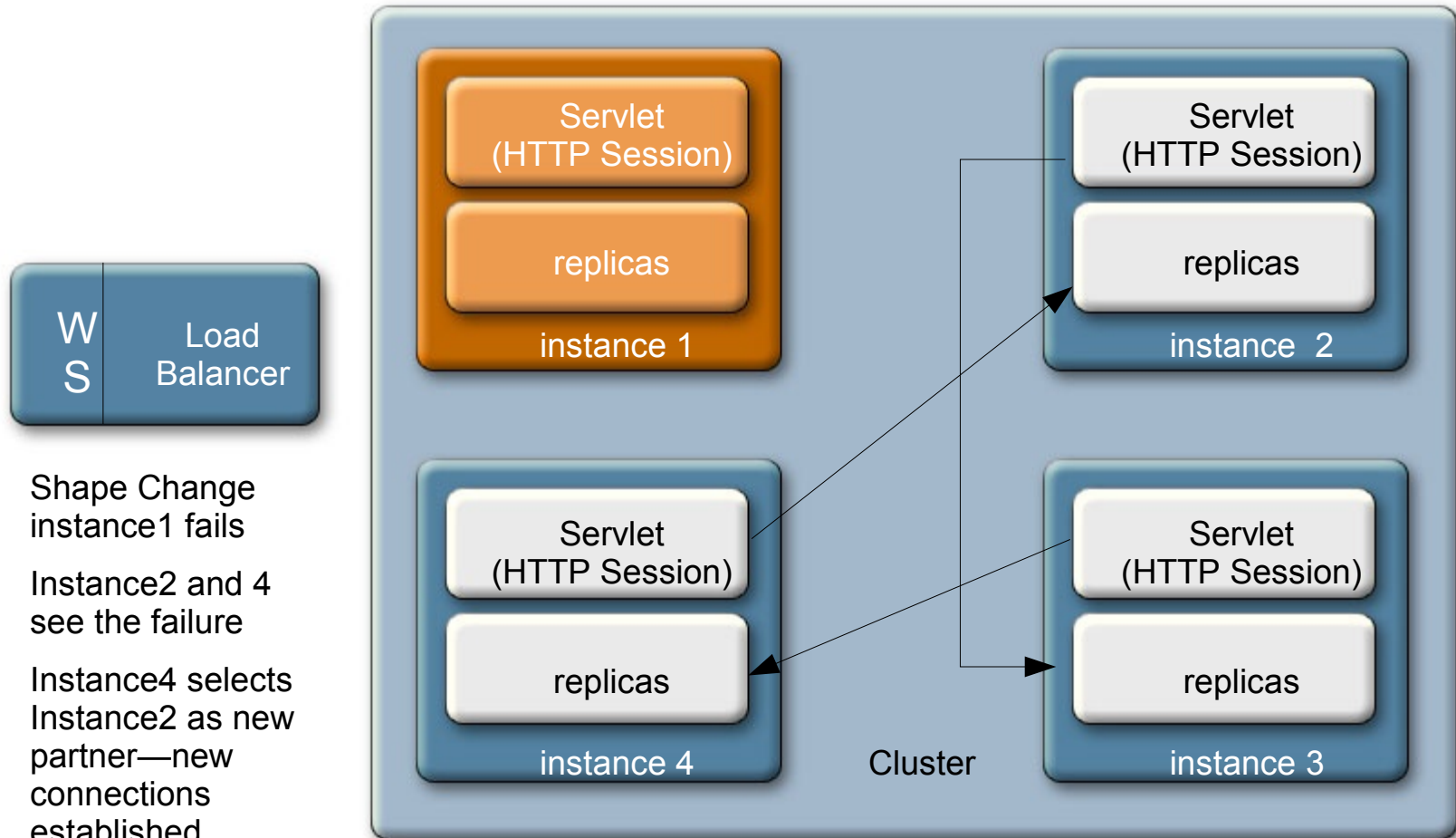
Cluster Dynamic Shape Change



Shape Change
instance1 fails

Instance2 and 4
see the failure

Cluster Dynamic Shape Change



Shape Change
instance1 fails

Instance2 and 4
see the failure

Instance4 selects
Instance2 as new
partner—new
connections
established

the reverse happens when an instance joins or re-joins the cluster

Memory Replication Configuration

Our hope was to say...

- “This page left intentionally blank” ;-)
 - > Meaning “zero configuration required”
- We came close to that goal...

Memory Replication Configuration

Out of the box...

- Create a domain
 - > Use the 'cluster' profile – defaults for replication are handled
 - > Enables GMS—heartbeat enabled
 - > **persistence-type = "replicated"**
- Create a cluster and instances
- Deploy your application with **availability-enabled=true**
- That's it

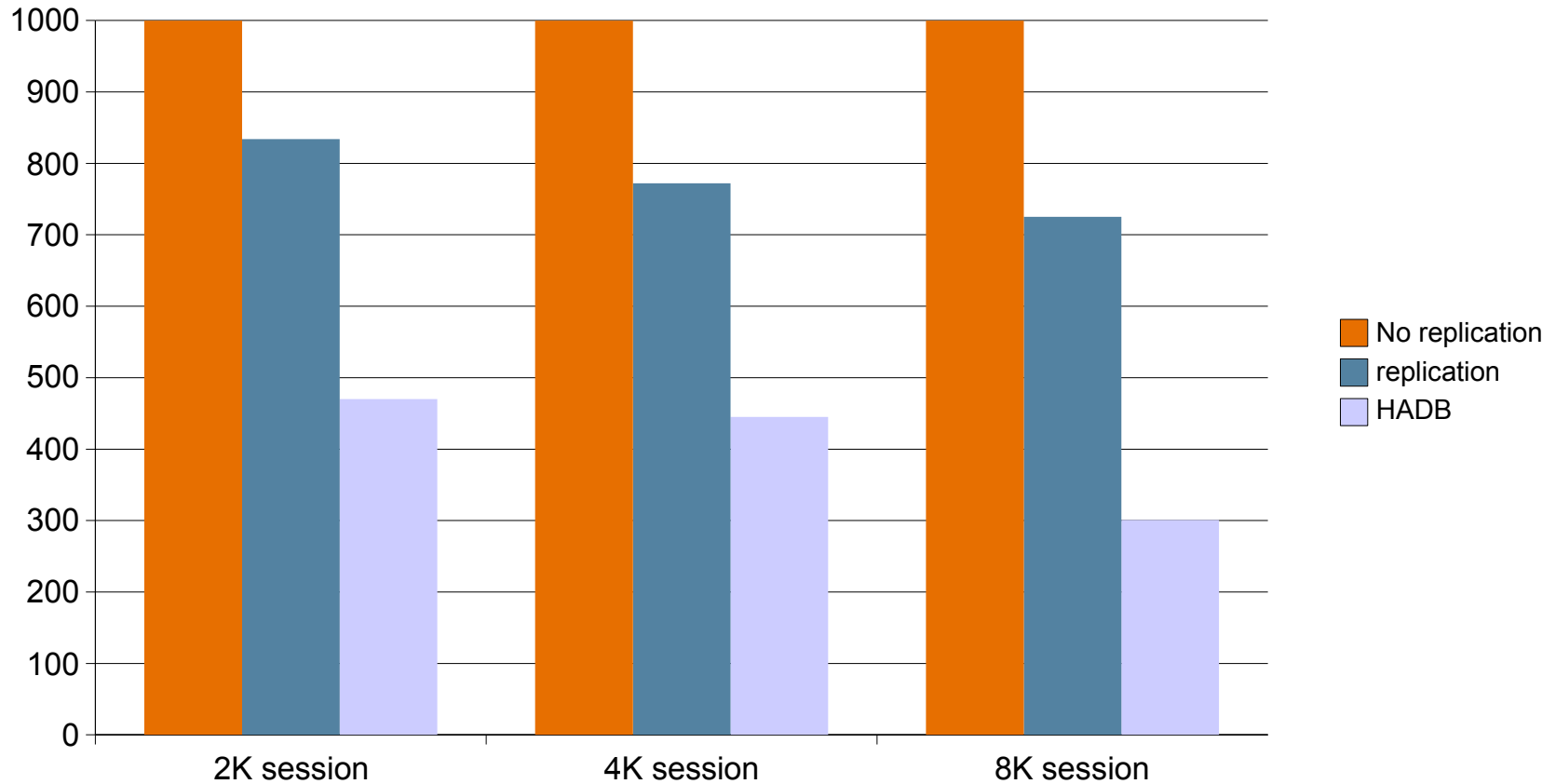
Memory Replication Configuration

Making your app distributable

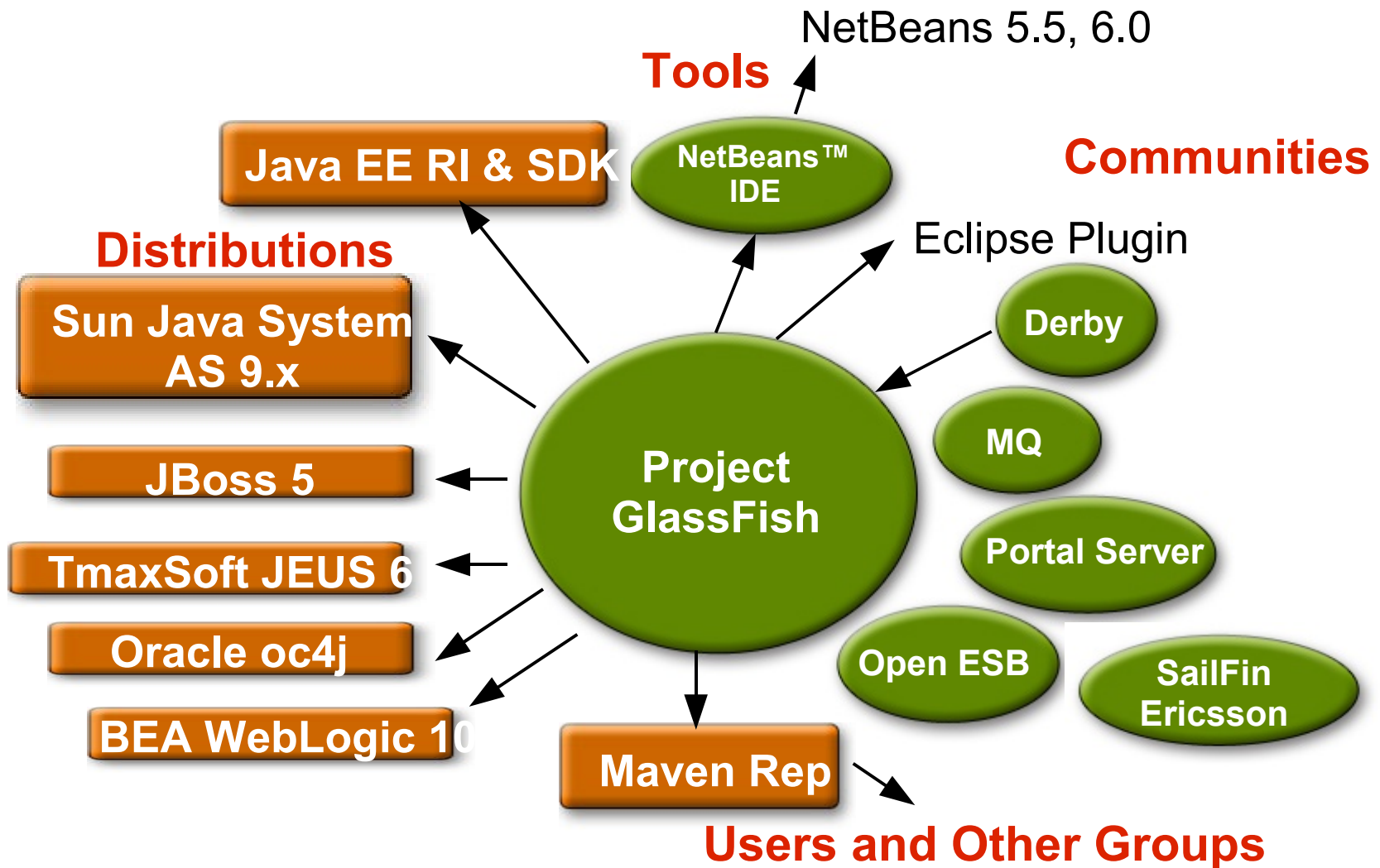
- **<distributable/>** element
 - > Required in **web.xml**
 - > Indicates you believe your application is ready to run in a cluster
- Serializable objects required
 - > HTTP Session state
 - > EJB technology Stateful Session Bean state

Memory Replication Performance

Hits/Sec ~17-25% throughput degradation



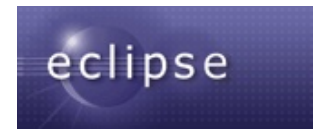
(Some) Distributions & Contributors



Tools Support

- NetBeans 5.5.1, 6.0
- Best integration with :
 - > full Java EE 5 support
 - > resource creation
 - > remote debug
 - > incremental deployment
 - > profiling
 - > wizards, etc...
- Additional features (SOA, UML, jRubyOnRails, ...)

- GlassFish (v1, v2, v3) plugin for Eclipse 3.3 (Europa)



- Genuitec's MyEclipse offers greater integration between IDE and GlassFish



GlassFish v3

- Small & Fast
 - > Less than 1.0 sec startup
- Totally Modular, kernel is <100K
 - > Can run in a phone or a desktop application
 - > Can be embedded in-process
- An ideal Container for Web 2.0
 - > Java Web Applications, PHP, jRuby/RoR, ...
 - > Support for upcoming Java EE 6 profiles
- A container that can do Java EE and more
- Screencast and preview code available today
 - > <http://hk2.dev.java.net>
 - > http://blogs.sun.com/dochez/entry/first_glassfish_v3_screencast
- Still early stages... working on schedule



AJAX and Scripting Activities

- AJAX
 - > jMaki - <http://ajax.dev.java.net>
 - > Encapsulates very easily AJAX widgets (JSP taglib)
 - > DynaFaces - <http://jsf-extensions.dev.java.net>
 - > AJAX and full-featured JSF components
 - > WoodStock - <http://woodstock.dev.java.net>
 - > Repository of AJAXyified JSF components
 - > JSF Templating <http://jsftemplating.dev.java.net/>
 - > Templating for pages and components
- Phobos - <http://phobos.dev.java.net>
 - > Scripting on the Server
- Comet and Grizzly - <http://grizzly.dev.java.net>
 - > Long-term HTTP connections for push content

The SailFin Project

- Ericsson SIP Servlet Contribution is available at:
 - > <http://sailfin.dev.java.net>
- Visit, Download, Try, Join
 - > Milestone 1 available
- Not just for telco operators!
 - > Bridging the HTTP and SIP protocols
- Built on GlassFish v2 and expected first half of 2008

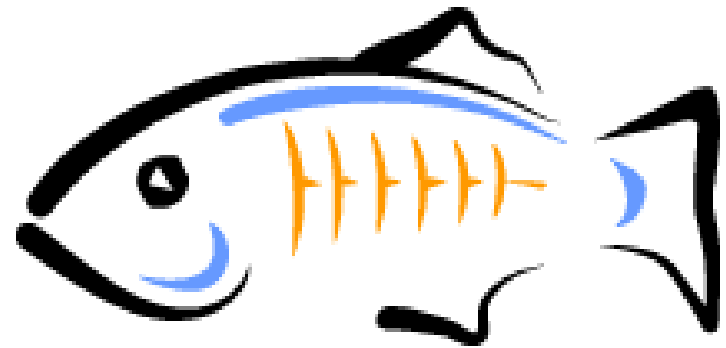


ERICSSON 
TAKING YOU FORWARD

<http://sailfin.dev.java.net>

Resources

- <http://glassfish.java.net>
- <http://wiki.glassfish.java.net>
- <http://blogs.sun.com/theaquarium>





Projekt OpenSSO

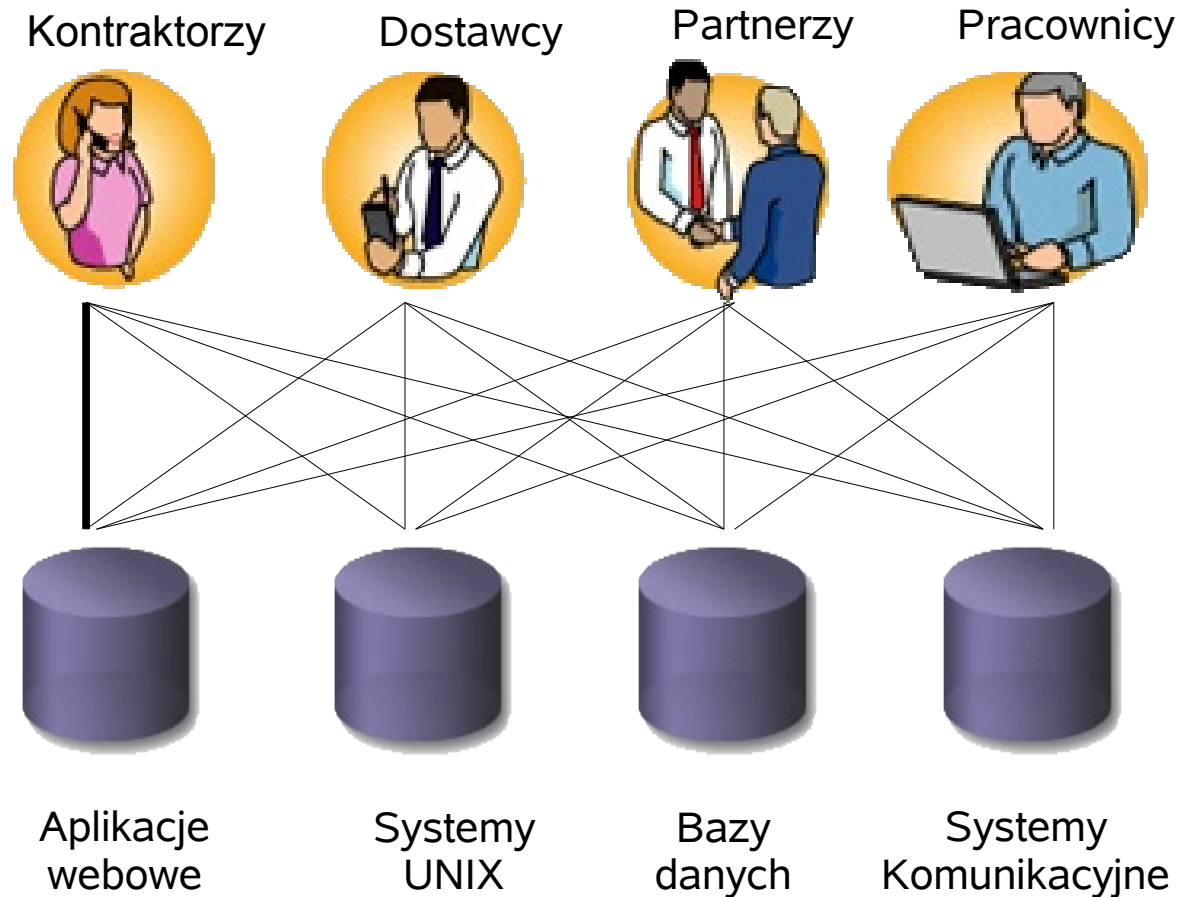
Marek Sokołowski
Sun Microsystems Poland



Kryzys tożsamości

Użytkownicy

- “Wyspy” tożsamości
- Mniejsza kontrola bezpieczeństwa i prywatności
- Powielone dane w wielu miejscach - “źródłach”
- Trudne utrzymanie spójności danych
- Wysokie koszty operacyjne
- Uregulowania prawne



Systemy/aplikacje

“Identity Grid”



Administratorzy



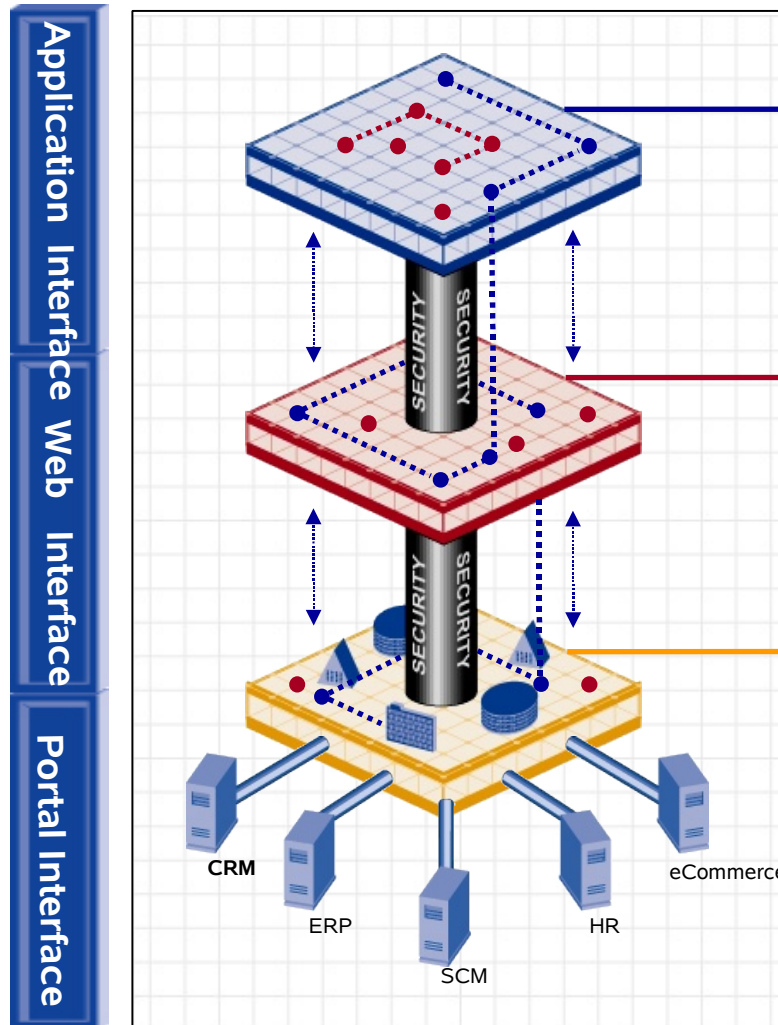
Pracownicy



Partnerzy



Klienci



Kategoria modułu

Usługi administracyjne

- Provisioning
- Zarządzanie hasłami
- Administracja użytkownikami
- Synchronizacja danych
- Policy Management

Usługi transakcyjne

- Transport “tokenów”
- Uwierzytelnienie
- Autoryzacja

Repozytoria danych

- Katalogi
- Relacyjne bazy danych
- “Płaskie” pliki

Open SSO

OpenSSO
Open Access . Open Federation

The Open Web SSO project (OpenSSO) provides core identity services to simplify the implementation of transparent single sign-on (SSO) as a security component in a network infrastructure. OpenSSO provides the foundation for integrating diverse web applications that might typically operate against a disparate set of identity repositories and are hosted on a variety of platforms such as web and application servers. This project is based on the code base of Sun Java™ System Access Manager, a core identity infrastructure product offered by Sun Microsystems.

<https://opensso.dev.java.net>



Solution Objectives

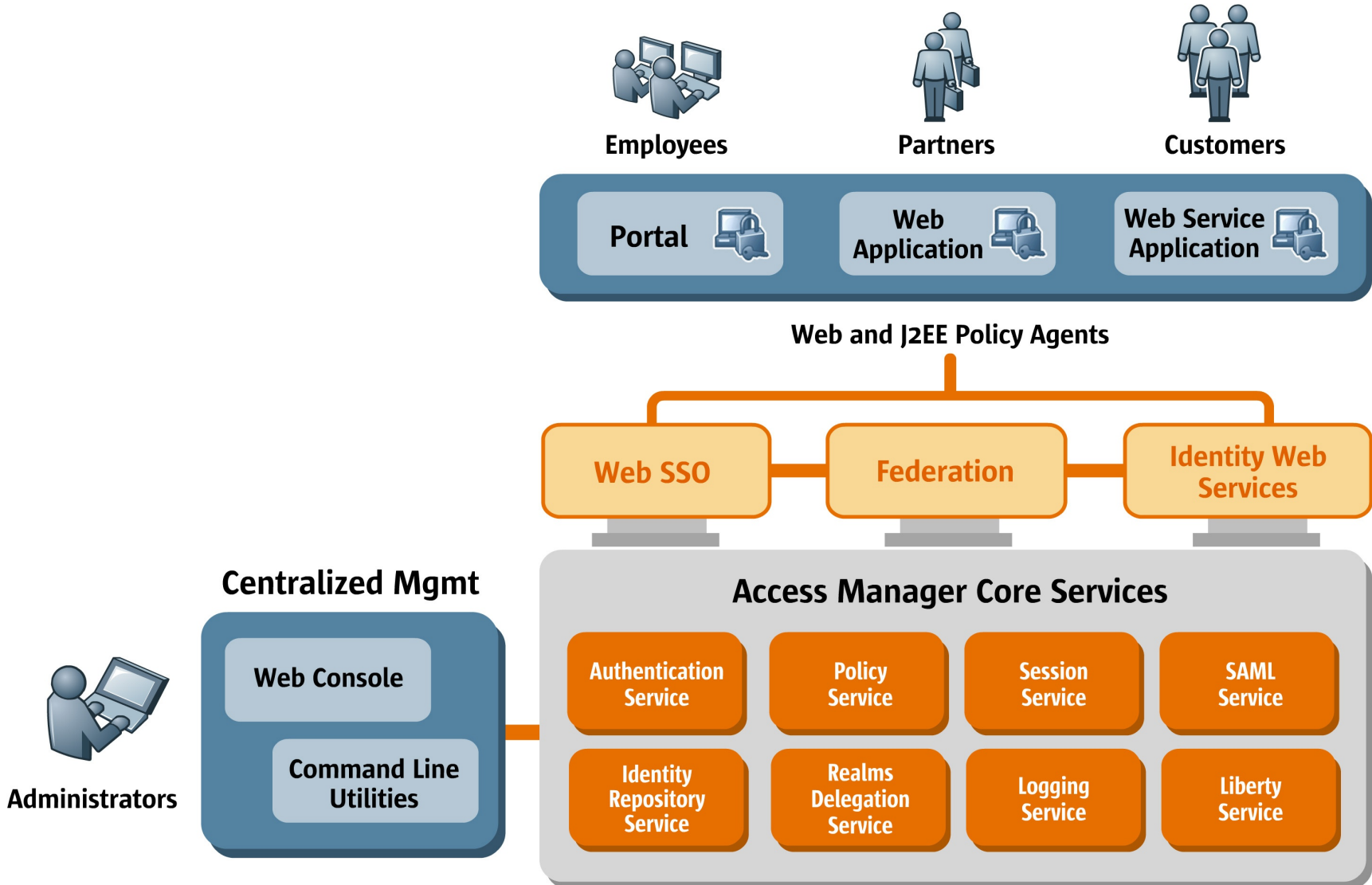
Access Management

- Consolidate security and access controls for employees, contractors, partners, and consumers
- Use familiar authentication mechanisms to simplify the user experience
- Create protection to match the sensitivity of applications
- Track and audit user access requests and resulting activities

Federated Identity

- Create seamless log-ins across internal and partner hosted applications
- Securely and privately share user information with partners
- Create a reusable framework based on open standards
- Apply consistent policy-based access across your partner ecosystem

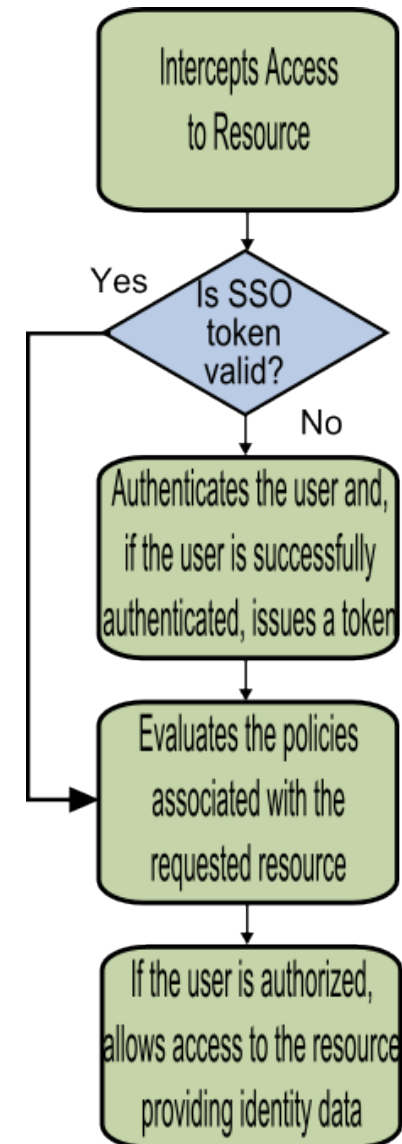
OpenSSO Architecture



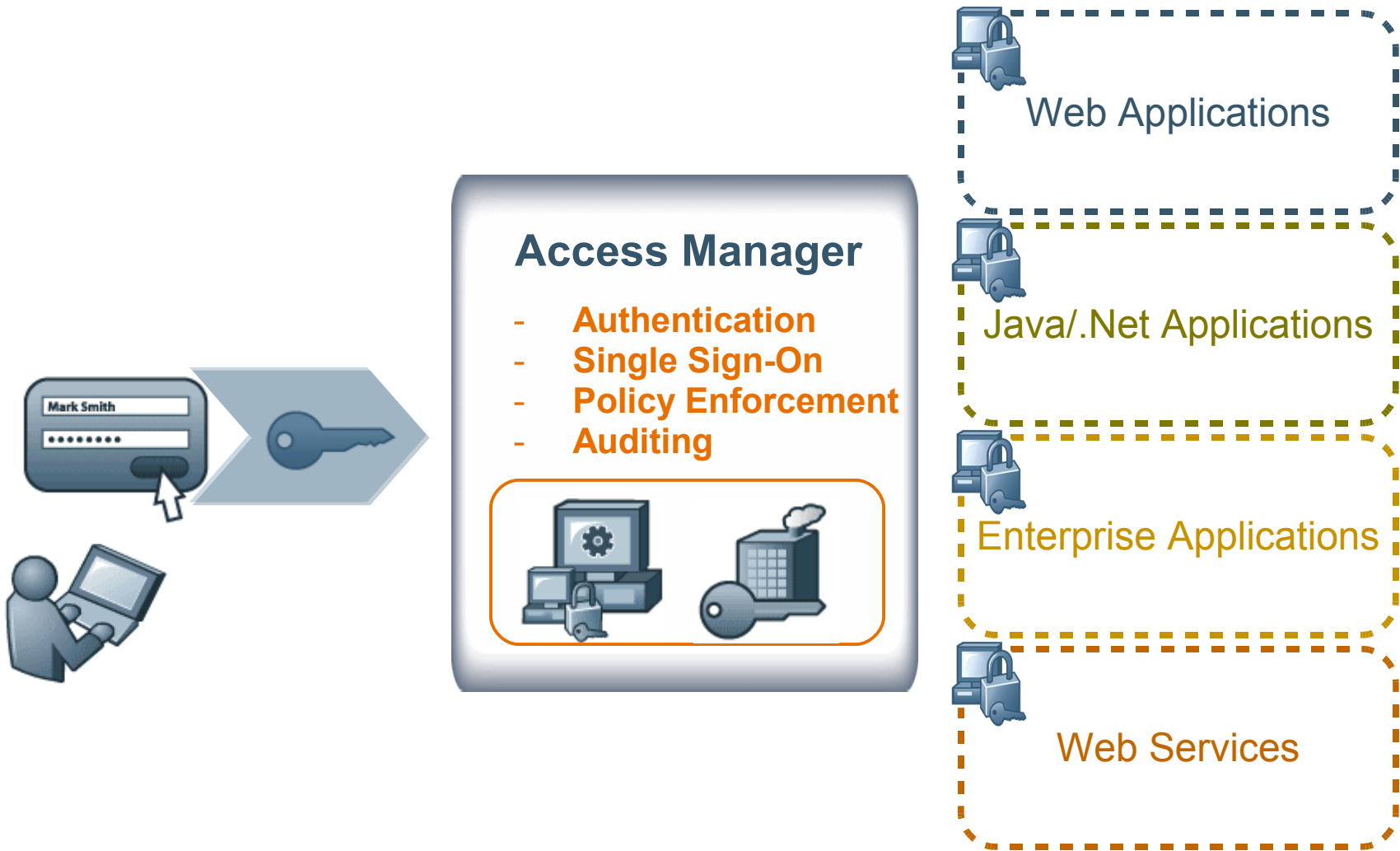
OpenSSO

How does it work?

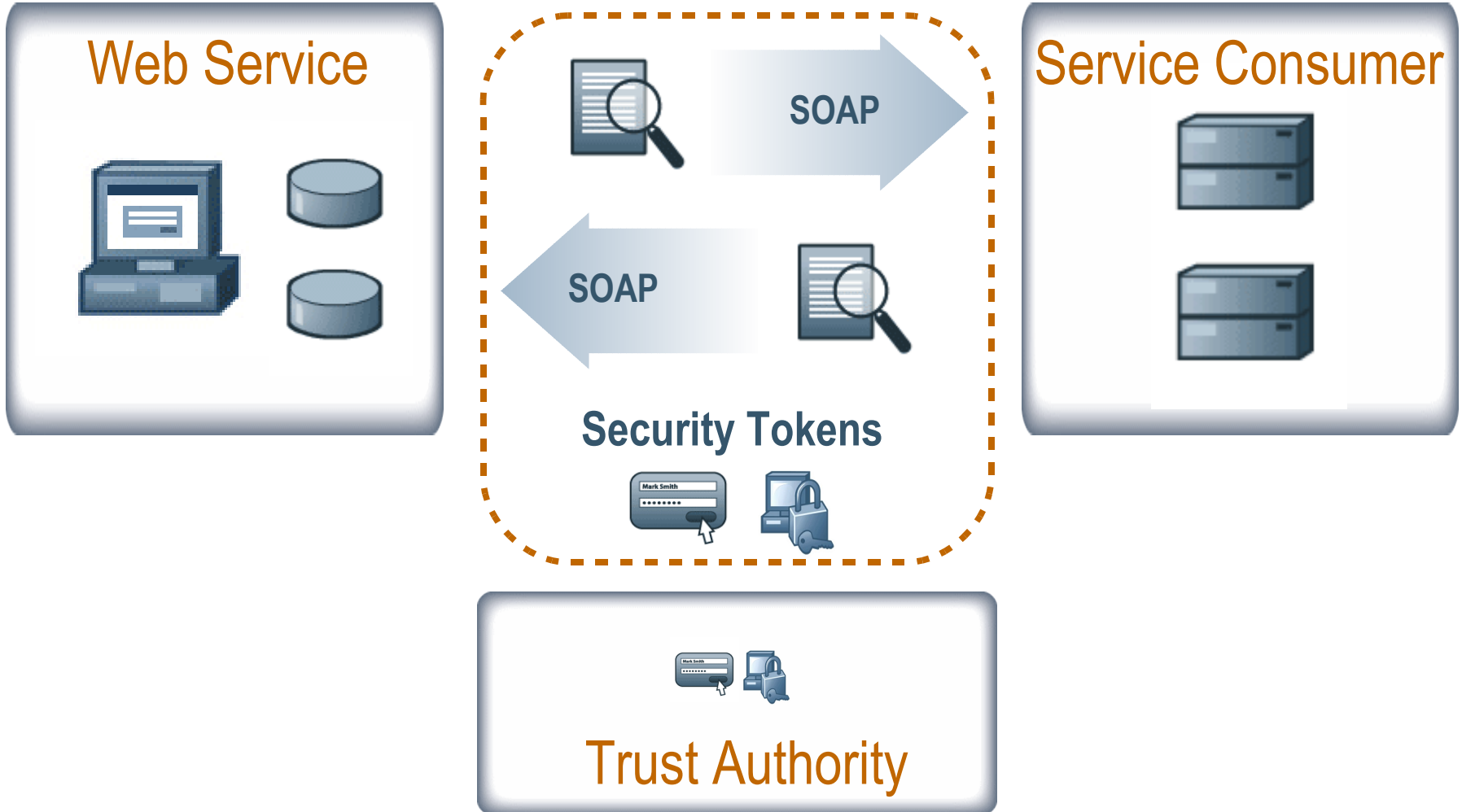
- Intercepts access to a resource
- Authenticates the user and, if the user is successfully authenticated, issues a token
- Evaluates the policies associated with the requested resource
- If the user is authorized, allows access to the resource, providing identity data
- Repeats the process
 - > Intercepts access to resource
 - > Uses token to authorize access depending on policy
 - > Provides identity data to resource
 - > Logs everything that happens
- Until session expires



Centralized Access Management

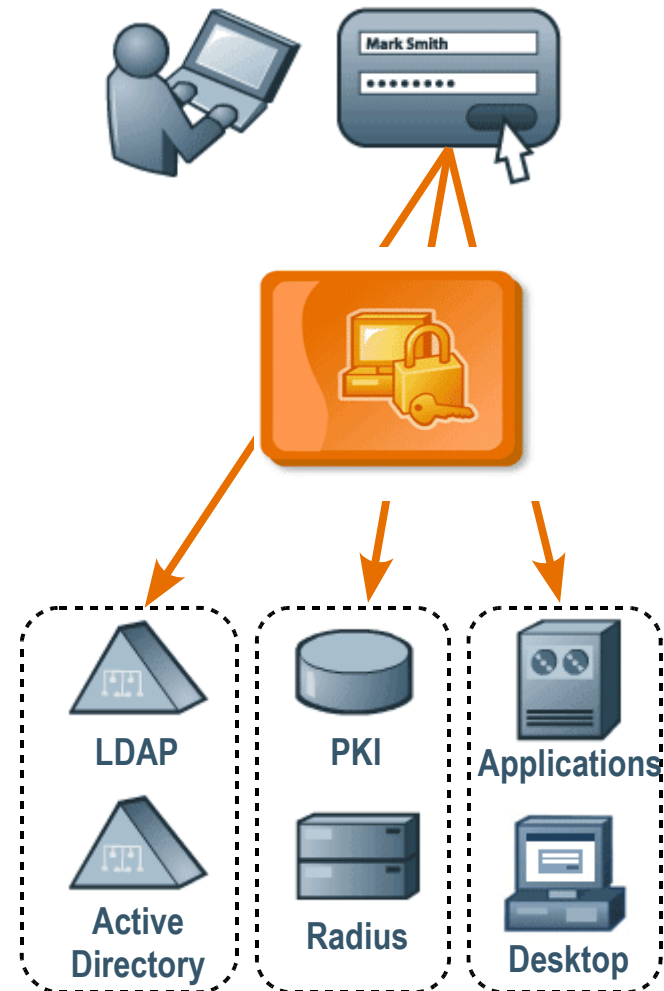


Secure Web Services



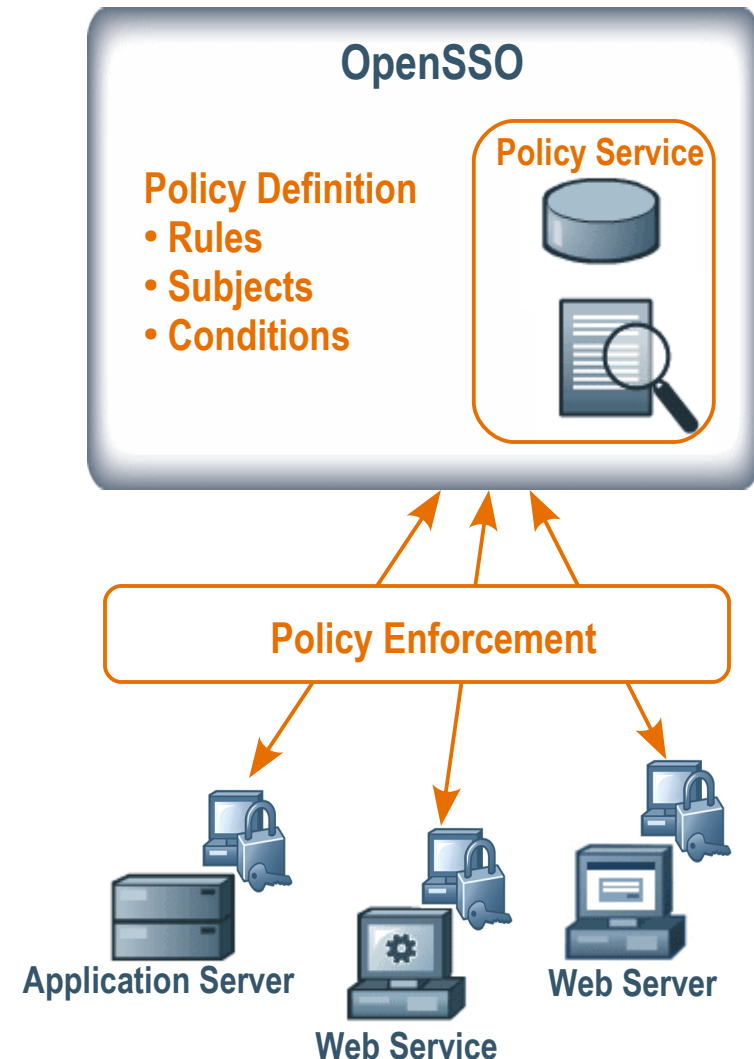
Centralized, Flexible Authentication

- Centralized Authentication Service, Authentication Modules and extensible framework
 - Reuse existing authentication mechanisms for any logon event – over 15 out of the box mechanisms
 - Link authentication mechanisms depending on the sensitivity of applications
 - Distribute authentication capabilities anywhere in the enterprise



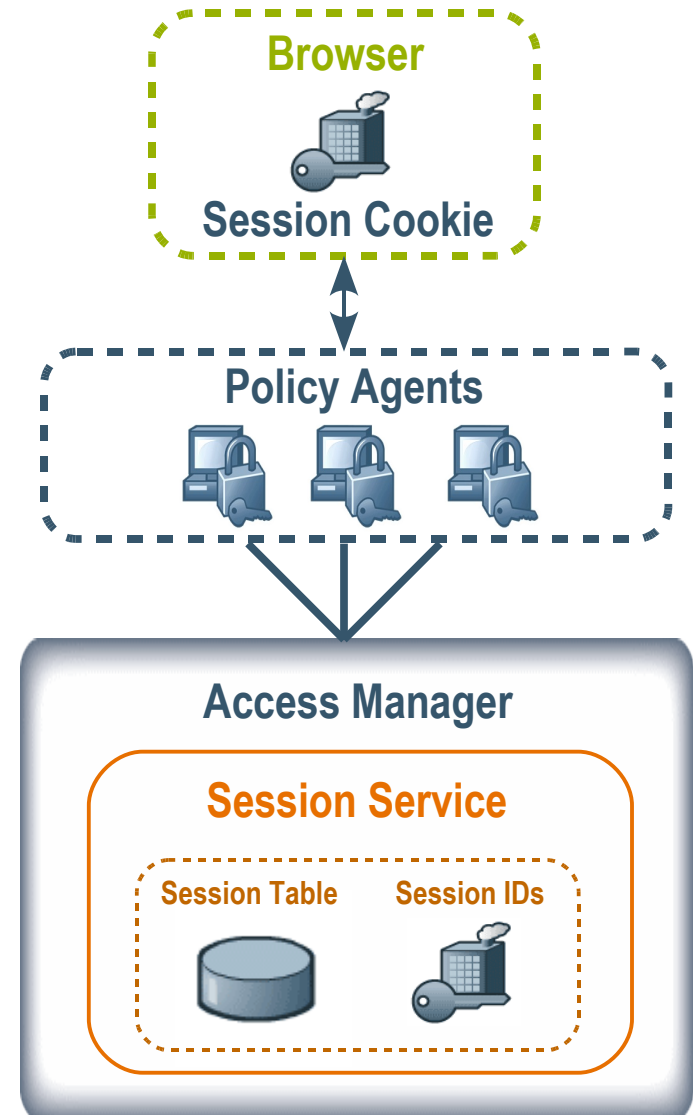
Authorized Access to Resources

- Centralized Policy Engine for real time access control decisions
 - Flexible policy configuration to define user relationships to protected resources
 - Plugin mechanisms to extend user profile information or make dynamic policy decisions
 - Distributed policy enforcement through Policy Agents, custom SDKs



Single Sign On to Any Resource

- Create and centrally manage user sessions for protected resources
 - Enable a sign on once, access anywhere experience for users
 - Provides a comprehensive view of activity on protected resources
 - Set session timeouts for users, groups, roles, or resources
- Provides for 3 main types of sessions
 - Basic user session: establish access on an application
 - Single sign-on session: share that session across multiple applications
 - Cross domain single sign-on: share that session with business partners



Centralized Auditing and Logging

- Centrally track all AuthN, AuthZ events
- Provide easy to manage proof points
 - > Who had access, who granted that access
 - > What systems did they access
 - > What functions did they perform
 - > When did they perform those functions
- Standards-based implementation
 - > Easy integration with existing auditing, reporting tools

Flexible Deployment

- Deploys as a Java web application to a standard web container
 - GlassFish Application Server, Sun Web Server
 - WebLogic
 - WebSphere
- Data store independent
 - Uses existing user profile repositories
 - Separates user profile store from policy and configuration
 - Works with LDAP, Active Directory, Flat files, DBs

Dziękuję

Marek Sokołowski
marek.sokolowski@sun.com

