Java.net Communications Community
JavaOne 2005

Mobicents
The First Certified Open Source Implementation of JAIN-SLEE 1.0

Ivelin Ivanov

JBoss, Inc.

Speaker Intro – Ivelin Ivanov

- Director of Product Development, JBoss
- Member of JBoss Core Team since 2003
- Java.net Communications Community co-lead
- Member of JSR 240 E.G.
- Member of Eclipse WTP Requirements Committee
- Contributor to GNU QeXO, Apache Cocoon, jXPath, XMLForm, FreeBuilder

Talk Overview

- IP Telephony: more than telephony over IP
  - ✓ Services – the key differentiator.
  - ✓ Concrete Example: SIP, commodity infrastructure, open standards.
  - ✓ Converged Services: separating carrier from operator, e.g. online gaming
- The requirements of IP telephony services motivate a new container architecture:
  - ✓ What are the requirements of such services?
  - ✓ Why does EJB + Signaling Stack not adequately address these requirements?
  - ✓ What motivates the need for a new service architecture?

Talk Overview

- Implementing the JAIN-SLEE spec on JBoss:
  - ✓ Quick SLEE Demonstration : A SIP Proxy Server
  - ✓ Key JBoss AS components used in the implementation.

IP Telephony In the Large

- There’s two parts to IP Telephony:
  - ✓ Call setup (signaling) and media.
  - ✓ Signaling is where the Network Intelligence (services) reside.
- This talk will focus on Signaling and Services

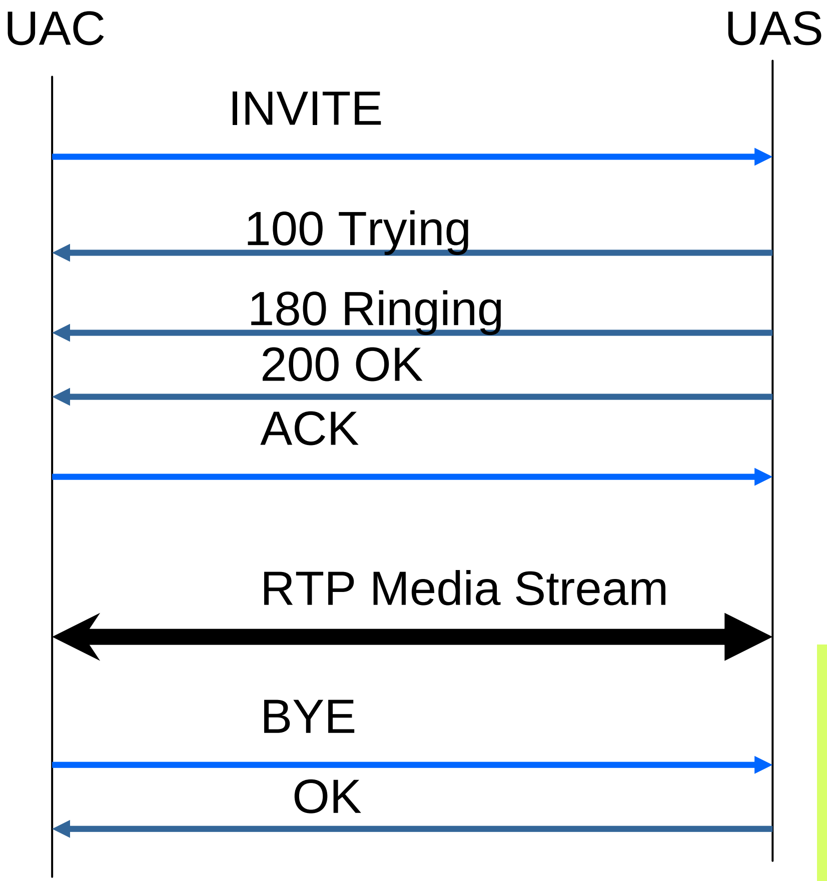
IP Telephony In the Large

- VOIP is everywhere!
  - ✓ Free or cheap voice is a commodity
  - ✓ Lower cost is not enough
  - ✓ Services is the differentiator – the way to make revenue
  - ✓ Innovation required
- VOIP enables innovation
  - ✓ New classes of services become possible
  - ✓ Converged services which combine VOIP and web services.

Signaling and Services

- In order to set up a call the two end-points (IP Phones) exchange messages.
- SIGNALING refers to the messages that are required to set up the call.
- SIGNALING is interesting because Services reside in the Signaling Plane.
- SIP is the dominant standard for call setup.
- We will motivate the requirements using SIP as an example.
  - ✓ SLEE is SIGNALING PROTOCOL AGNOSTIC.

Motivating the Requirements
Example Simple SIP Call Flow

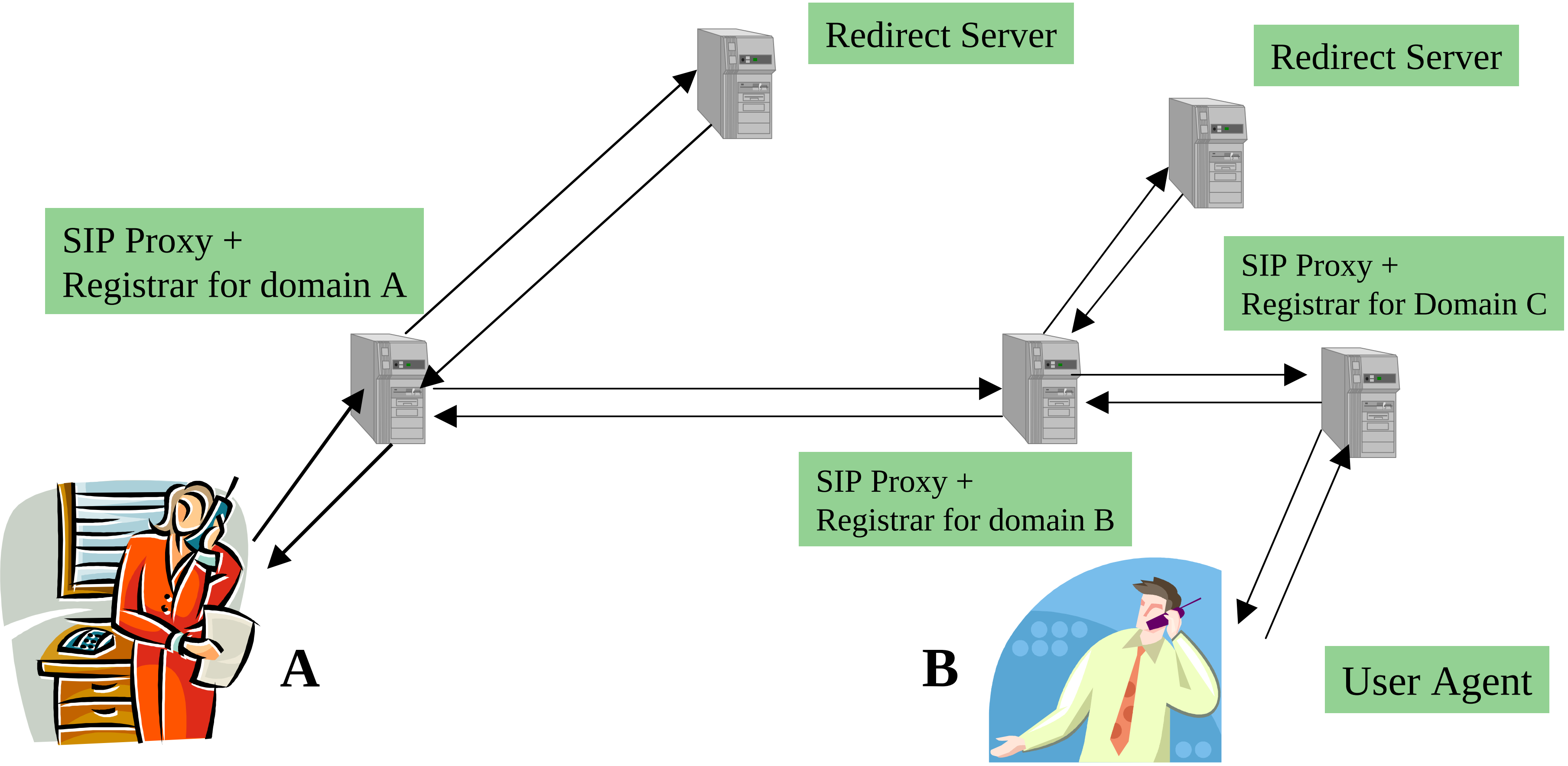


UAS – User Agent Server

UAC – User Agent Client

There can be intermediate Signaling nodes (Proxy Servers that keep call state and network services).

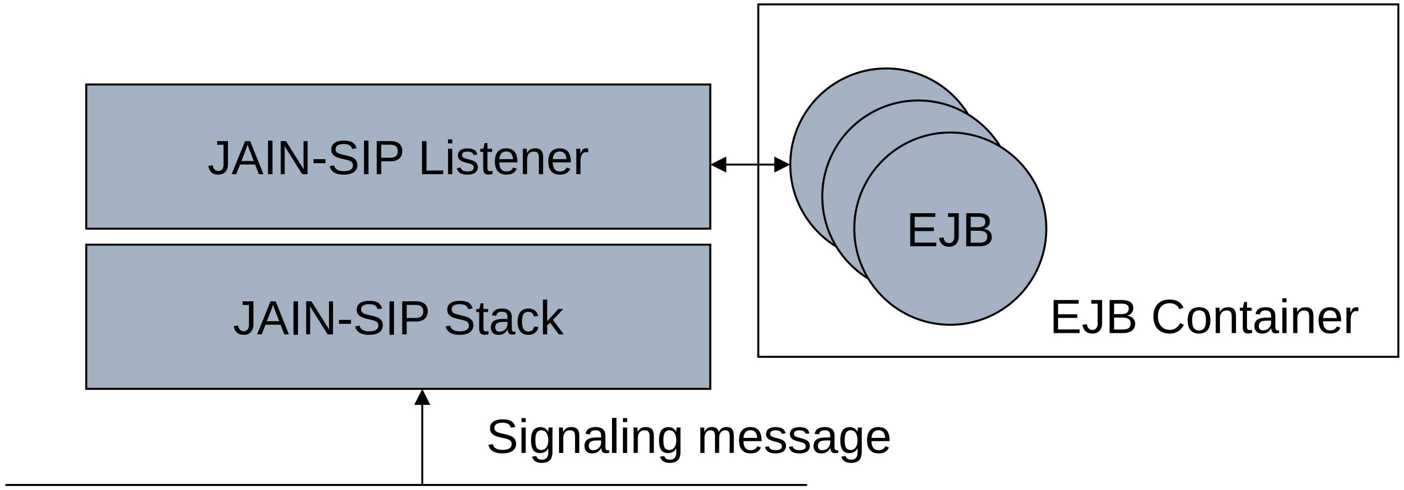
Motivating the Requirements
A Typical SIP Enabled Network



An Architecture for Building Services

- Components are good.
  - ✓ But I am preaching to the choir!
  - ✓ We need a component oriented event driven service platform
- Need high reliability and failure resilience
  - ✓ No downtime
  - ✓ 50ms response time
- Transactions are good
  - ✓ Simplifies the task of building resilient applications.
  - ✓ But I am preaching again!
- So we need a component oriented transaction supporting, event driven platform.

A Possible architecture for Building Signaling Services



Tightly Coupled Listener

Constrains distribution.

Object management is under application control

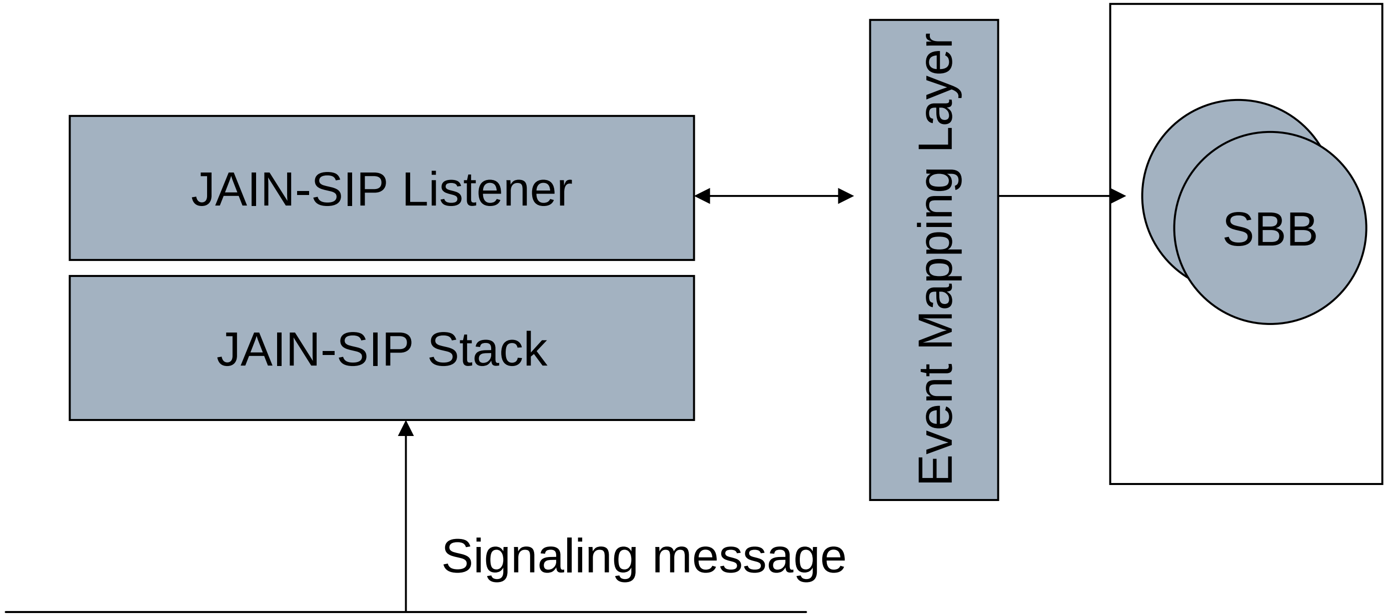
Application Complexity

High Latency

Persistent state is stored in an EJB.

Lets Replace the EJB

EJB offers a nice component model.
Lets keep the cool stuff about the EJB model and toss the rest out.



- Replace EJB with a lighter weight component - “SBB”
- Event driven (one way messages)

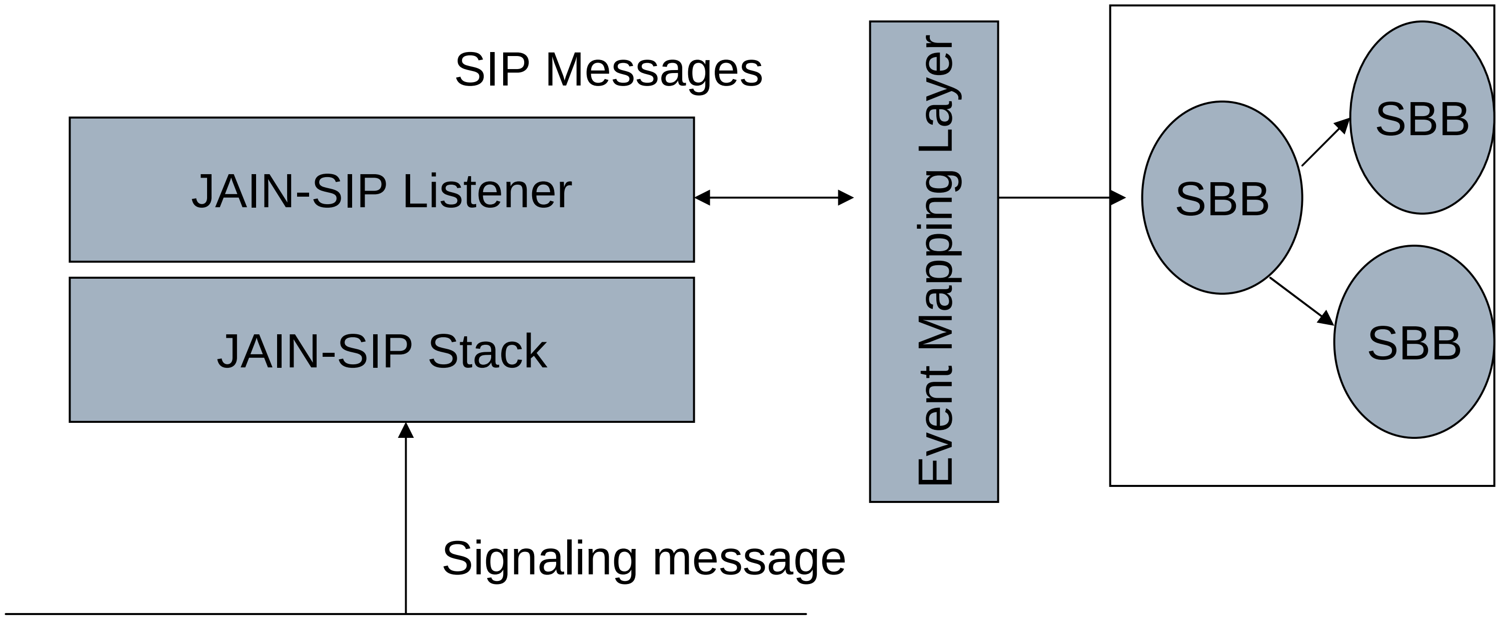
What about execution order of the SBBs?

Services and SBBs

- Services are compositional
- Each compositional block is an SBB.
  - ✓ SBB: Event Driven Service Building Block
- SBBs fire in response to events
- SBBs send each other events.
- Order of SBB execution is important
  - ✓ Otherwise outcome of composition is non-deterministic.

Lets group and order the SBBs

Lets group SBBs and define a means for specifying execution order



SIP Messages

JAIN-SIP Listener

JAIN-SIP Stack

Event Mapping Layer

SBB

SBB

SBB

Signaling message

A Service is a group of related SBBs.
Deployment descriptor allows us to specify execution order of SBBs.

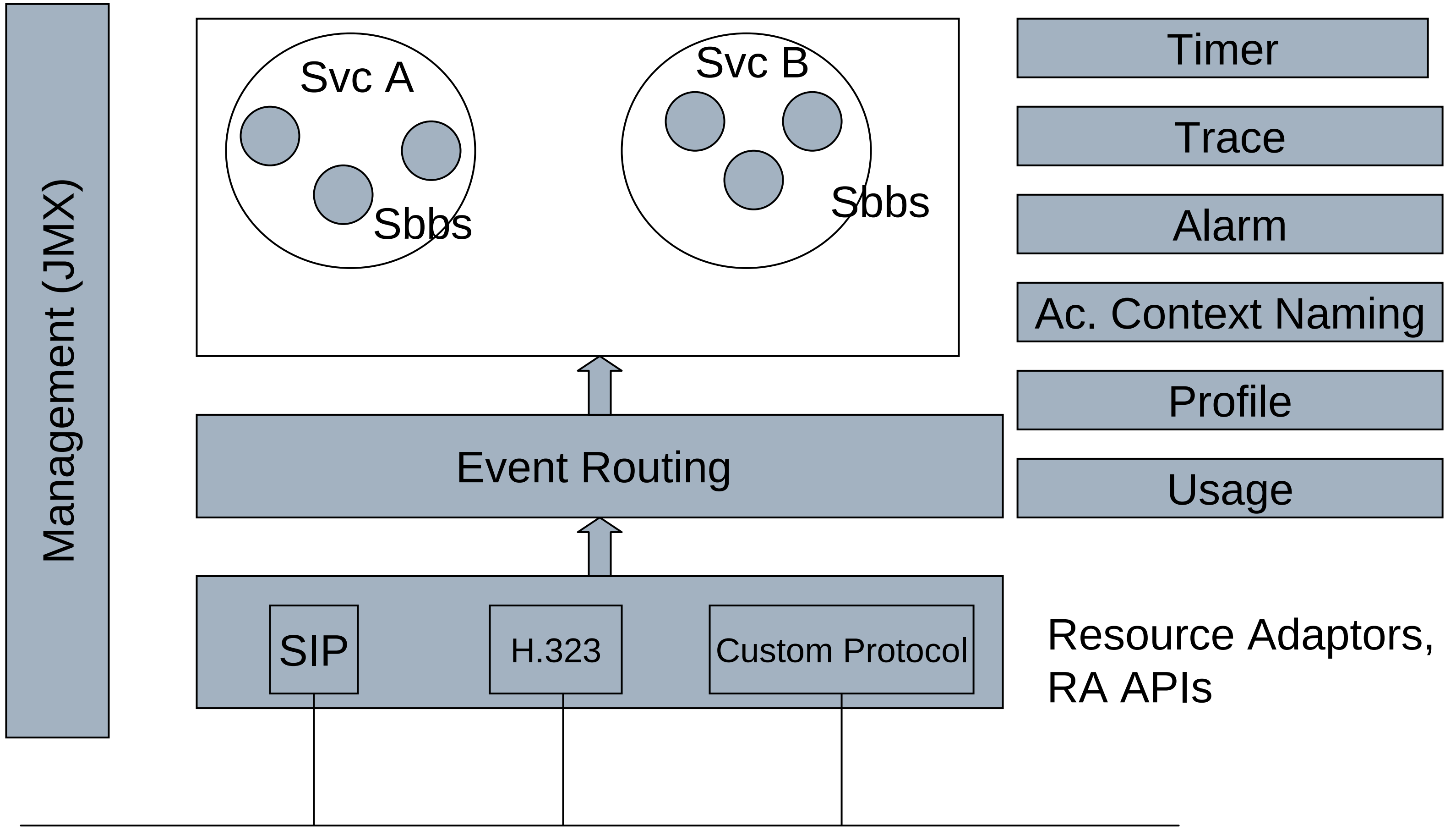
Summing it up: What is the SLEE ?

- JSR 22 (spec leads Open Cloud and Sun).
- Crafted for the needs of Communications service platforms
  - ✓ Highly Available
  - ✓ Scalable
  - ✓ Distributed
- Supports standard JMX Management Interfaces
- Supports standard facilities (timer, trace, usage, alarm)
- Point of integration for multiple protocols and components:
  - ✓ Events and components are strongly typed using java interfaces.
  - ✓ A single container can support multiple protocols

Summing it up: Why Invent the SLEE?

- Need to support Asynchronous invocations.
  - ✓ EJBs are typically synchronous
- SLEE is designed for fine grained short lived objects that are typically replicated in memory.
  - ✓ SLEE objects are replicated in memory.
  - ✓ SLEE transactions are light weight.
  - ✓ SLEE manages transaction boundaries.

Simplified JAIN-SLEE Architecture



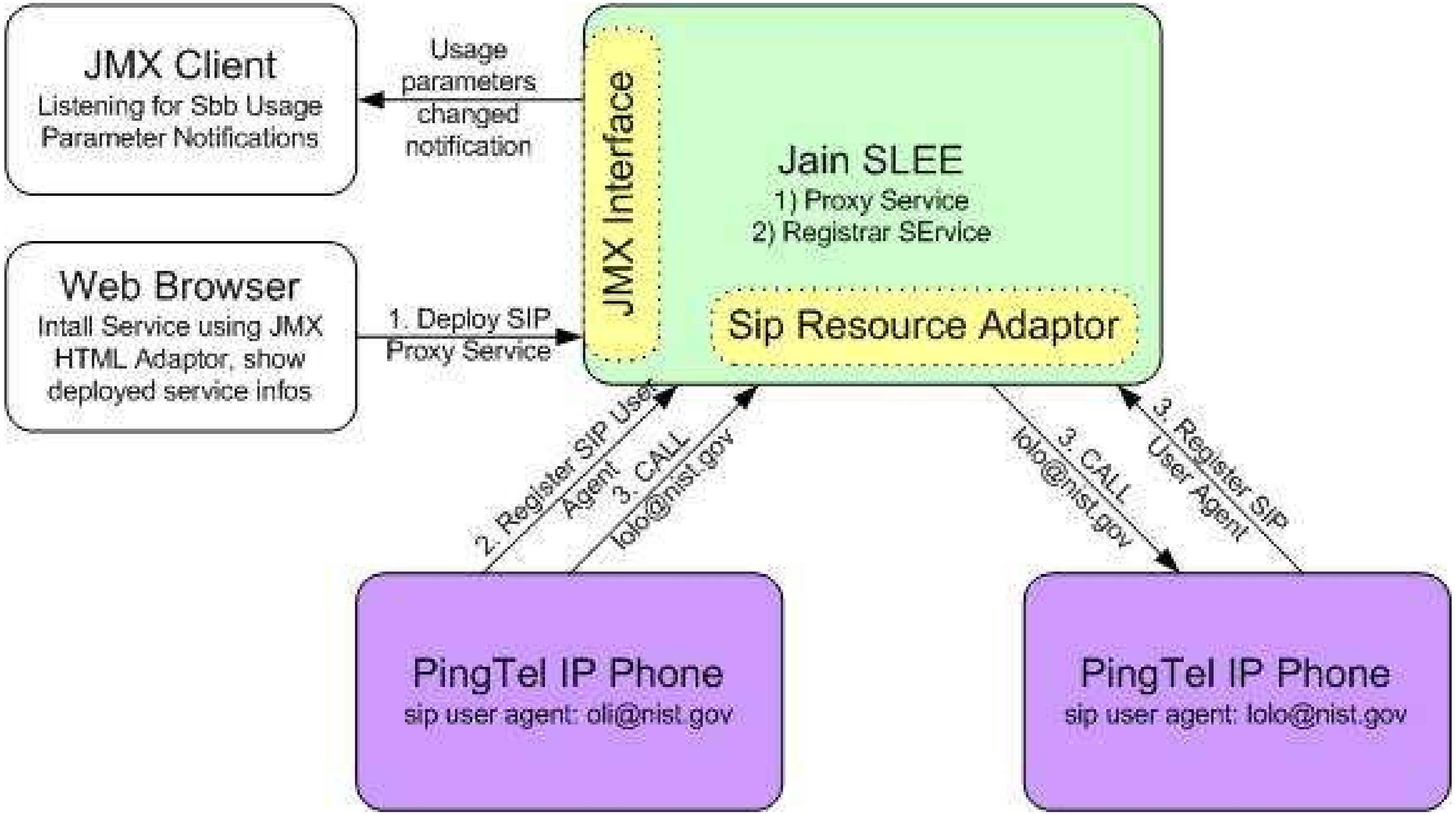
JMS vs. SLEE

- SLEE uses publish-subscribe model like JMS so why not just use it?
  - ✓ Impedance mismatch.
  - ✓ SLEE messages are supposed to be processed in 10-100 ms. JMS messages could take anywhere from seconds to days. Results in different implementation strategies.
  - ✓ The “Topics” are not known a-priori here.
  - ✓ JMS drags in baggage that we don’t want.

The Mobicents Project

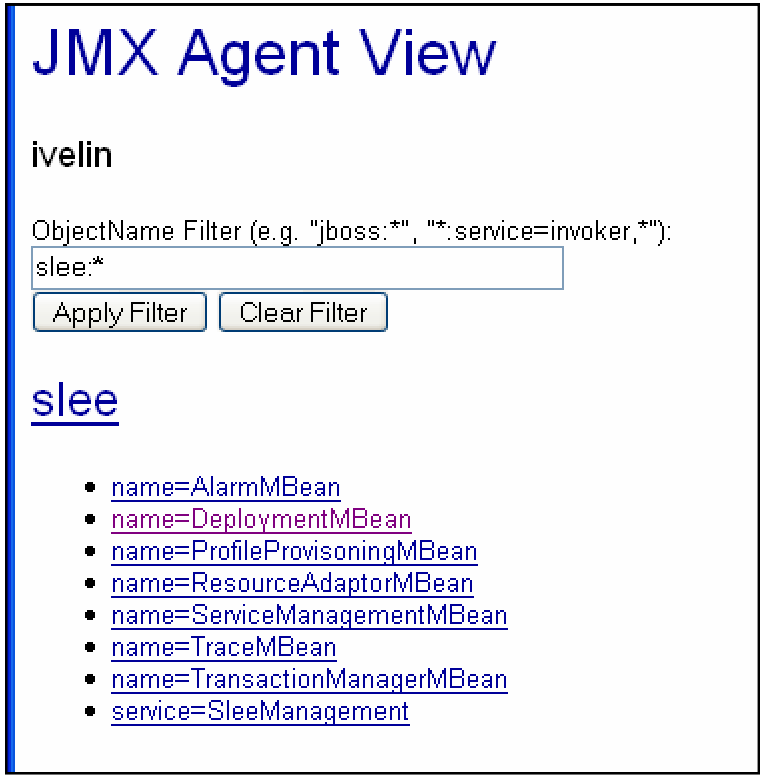
- Purpose – to build an open source SLEE implementation.
- Project is housed at http://www.mobicents.org
- Development Lead: M. Ranganathan (NIST)
- Founder, Administrator and Contributor: Ivelin Ivanov
- Core Contributors : Francesco Moggia, Tim Fox, Jean Deruelle, Buddy Bright, Ralf Siedow, Marco Montiero, Sancho Cesar Rego
- Significant contributions to date: Vodafone R&D, Lucent Technologies, PT Innovaco, TI Labs, Emil Ivov.
- An active project with a growing list of contributors!

Demo 1


JMX Client
Listening for Sbb Usage Parameter Notifications
Usage parameters changed notification
Web Browser
Intall Service using JMX HTML Adaptor, show deployed service infos
1. Deploy SIP Proxy Service
JMX Interface
Jain SLEE
1) Proxy Service
2) Registrar SErvice
Sip Resource Adaptor
2. Register SIP User Agent
3. CALL lolo@nist.gov
3. CALL lolo@nist.gov
3. Register SIP User Agent
PingTel IP Phone
sip user agent: oli@nist.gov
PingTel IP Phone
sip user agent: lolo@nist.gov

JMX and Management (cont')

- Deployment MBean
- All SLEE Services



JMX Agent View

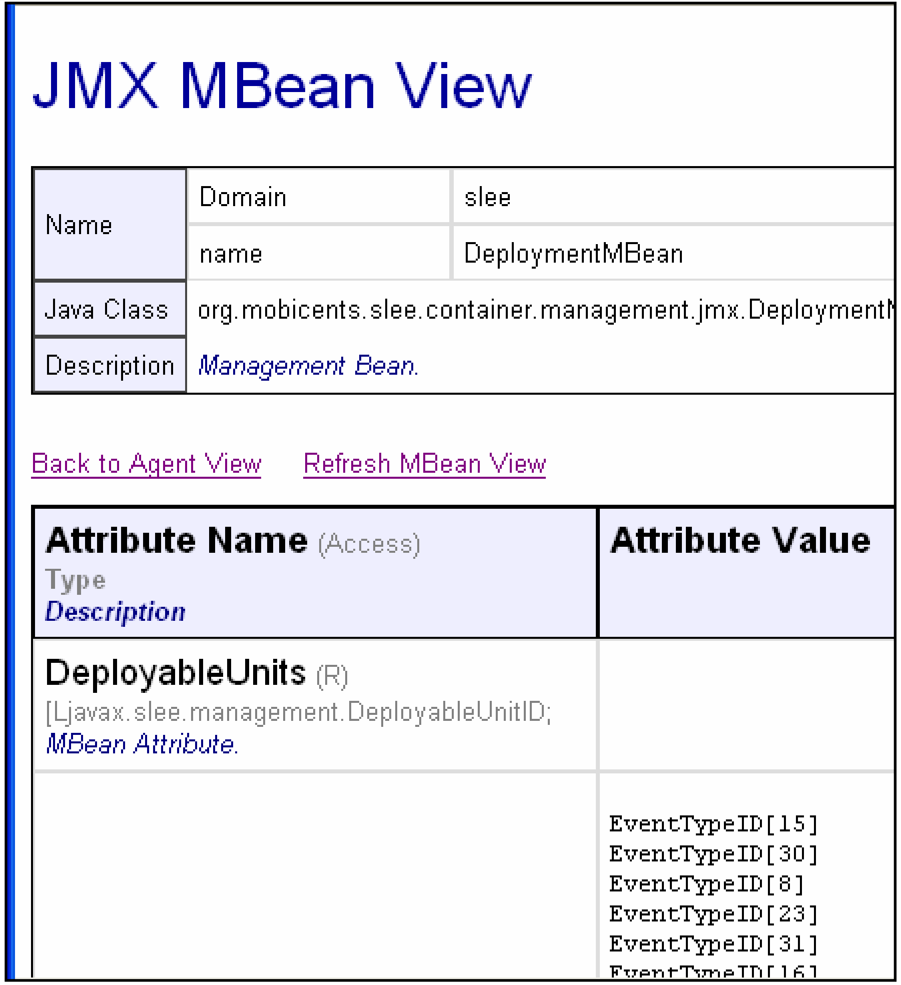
ivelin

ObjectName Filter (e.g. "jboss:*", "*:service=invoker,*"):

slee:*

slee

- name=AlarmMBean
- name=DeploymentMBean
- name=ProfileProvisoningMBean
- name=ResourceAdaptorMBean
- name=ServiceManagementMBean
- name=TraceMBean
- name=TransactionManagerMBean
- service=SleeManagement



JMX MBean View

Name	Domain	slee
	name	DeploymentMBean
Java Class	org.mobicents.slee.container.management.jmx.Deployment	
Description	*Management Bean.*	

Back to Agent View   Refresh MBean View

Attribute Name (Access) Type *Description*	**Attribute Value**
DeployableUnits (R) [Ljavax.slee.management.DeployableUnitID; *MBean Attribute.*	
	EventTypeID[15] EventTypeID[30] EventTypeID[8] EventTypeID[23] EventTypeID[31] EventTypeID[16]

Acknowledgement

- The JAIN-SLEE Specification is lead by Sun Microsystems and Open Cloud.
- Material from the JAIN-SLEE tutorial is reused in this presentation with permission.
- Mobicents is supported in part by the NIST Advanced Networking Technologies Division, PT, Vodafone, JBoss and others.