



# horizons

Sun's SOA & Composite  
Application Summit **2006**

**Kevin Schmidt, Director, Product Management**  
**Implementing a High  
Performance and High  
Availability Message Bus with  
the Java™ Message Service  
(JMS) Grid**

May 18, 2006



# Agenda

- Overview
- Details
- Integration with Sun Java™ Composite Application Platform Suite (Java™ CAPS)
- Comparison with Java™ Message Service (JMS) IQ Manager
- Beta and EAP Availability

# Overview

- New product offering
- Integration of the SpiritWave message server into Java CAPS
- Offered as a message server option with Java CAPS
  - Alternate message server that may be used with Java CAPS instead of JMS IQ Manager
  - Provides high availability and failover features not available with JMS IQ Manager

## Overview (cont'd)

- High performance single server JMS implementation
- Highly available multi-server clusters
  - Synchronized message stores (replication) providing:
    - fault tolerance
    - transparent failover & recovery
    - load balancing & scalability
  - Clustered / networked servers, filtering across WANs

# Details – History

- Originally SpiritWave from SpiritSoft
- Created in 1997
- First fully-featured all Java JMS API implementation

# Details – Features

- High Performance
  - Push based delivery
  - Tunable threads
  - Fast indexed message store
- Optimized flow control / throttling
  - Highly tunable environment
  - Slow consumers cause producers to be throttled

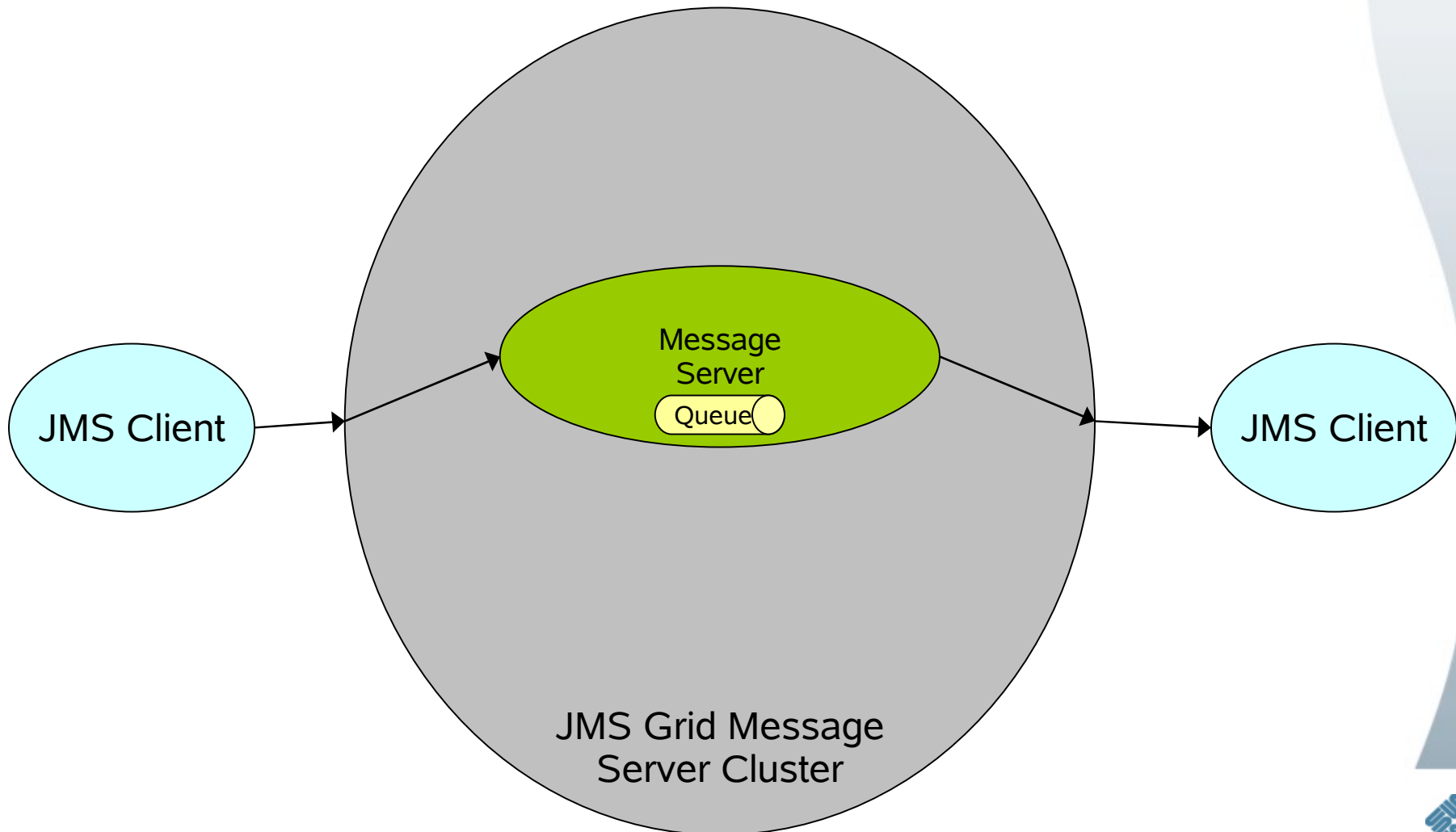
## Details – Features (cont'd)

- Content Based Filtering
  - JMS API message selectors
    - Standard JMS API properties
    - XPath of XML messages
  - Wildcard subscriptions
- Error Handling
  - Undeliverable messages may be routed to a dead-letter queue from which they can be replayed

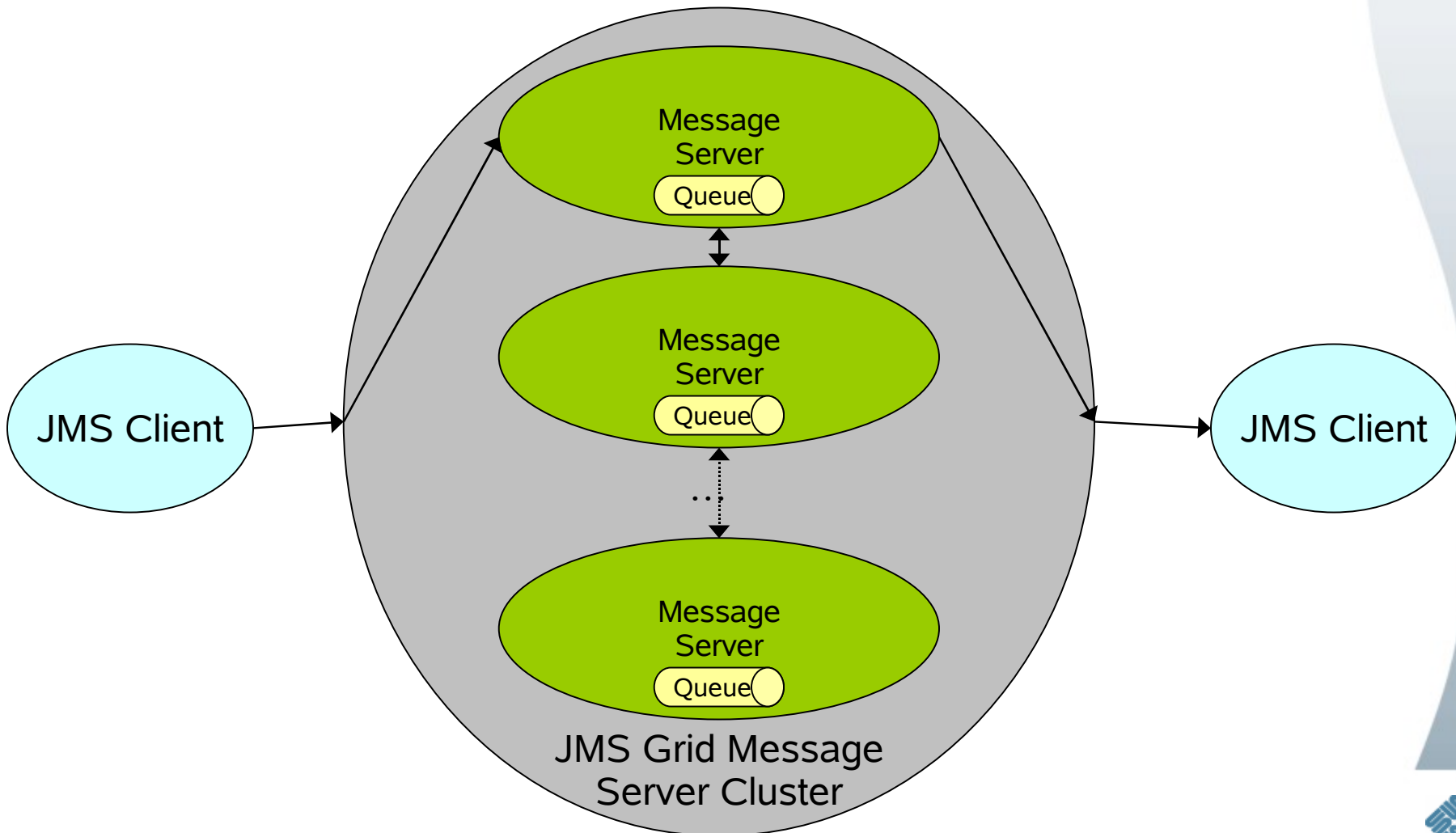
## Details – Features (cont'd)

- JMS API 1.1 compliant
- Pure Java Technology implementation
- C/C++ APIs

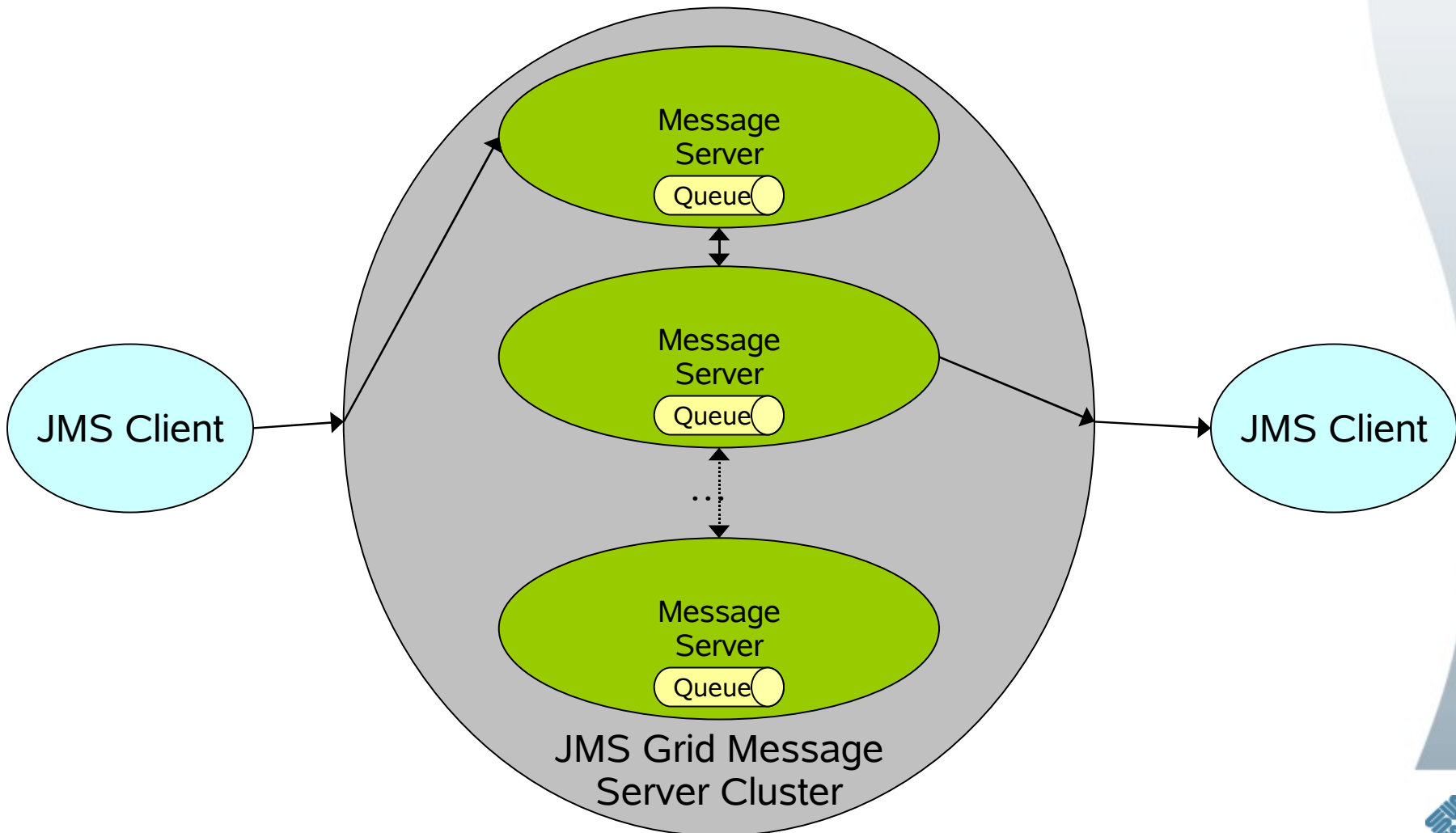
# Details – Single Message Server



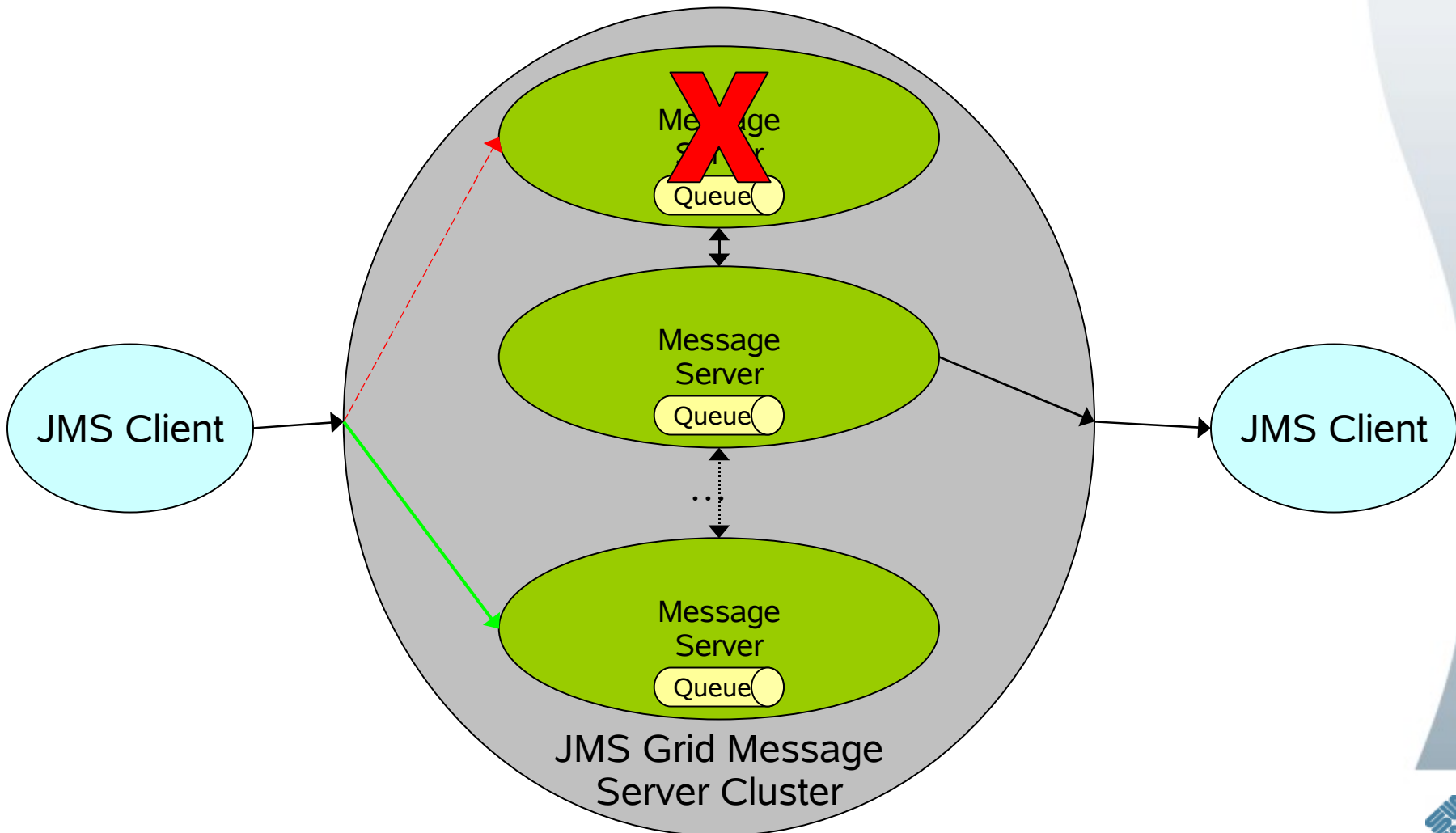
# Details – Cluster of Servers



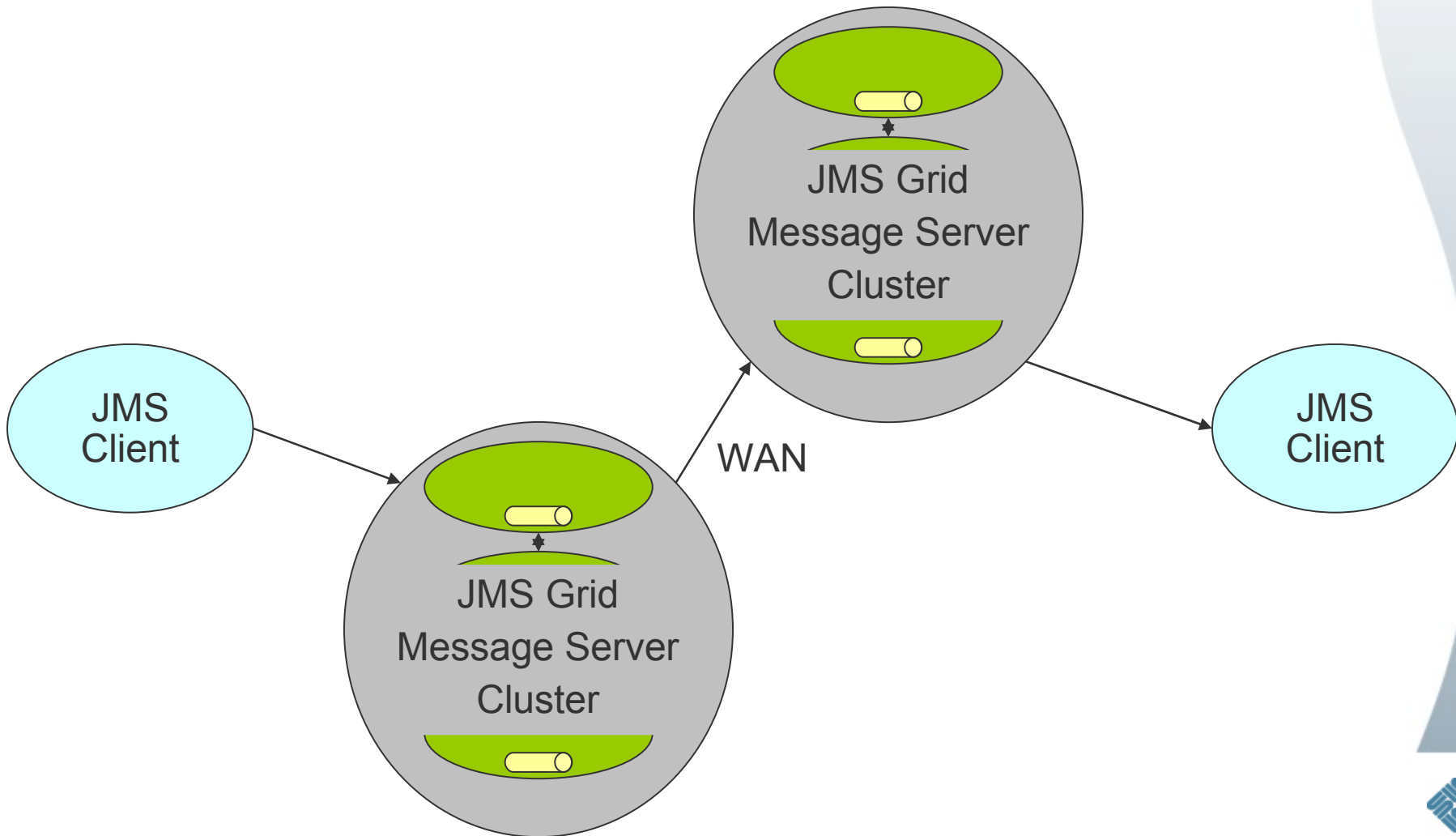
# Details – Queue Transparency



# Details – High Availability and Failover



# Details – Networks of Clusters

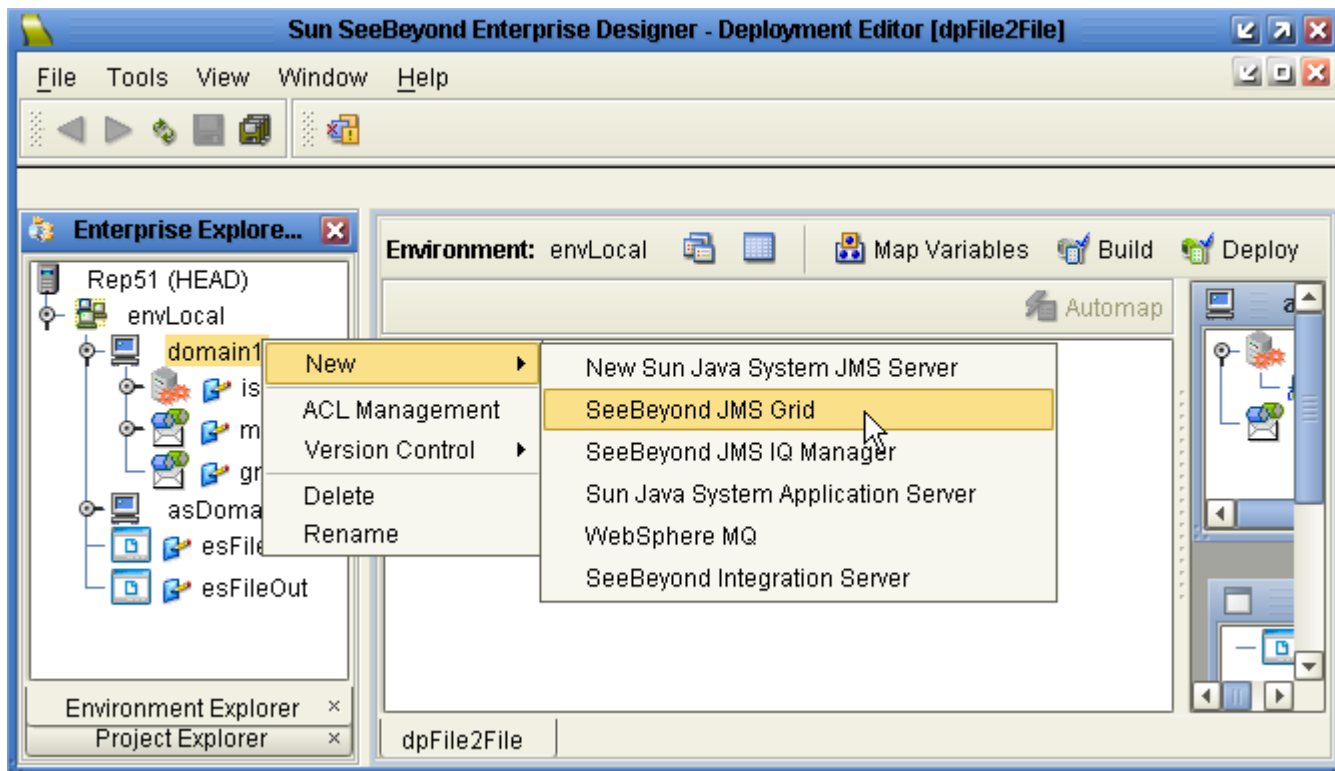


# Integration with Java CAPS

- Using JMS Grid instead of JMS IQ Manager requires **NO** changes to Project components
  - Create and configure JMS Grid in Environment
  - Map JMS API destinations to JMS Grid in Deployment Profile
  - Build and deploy normally

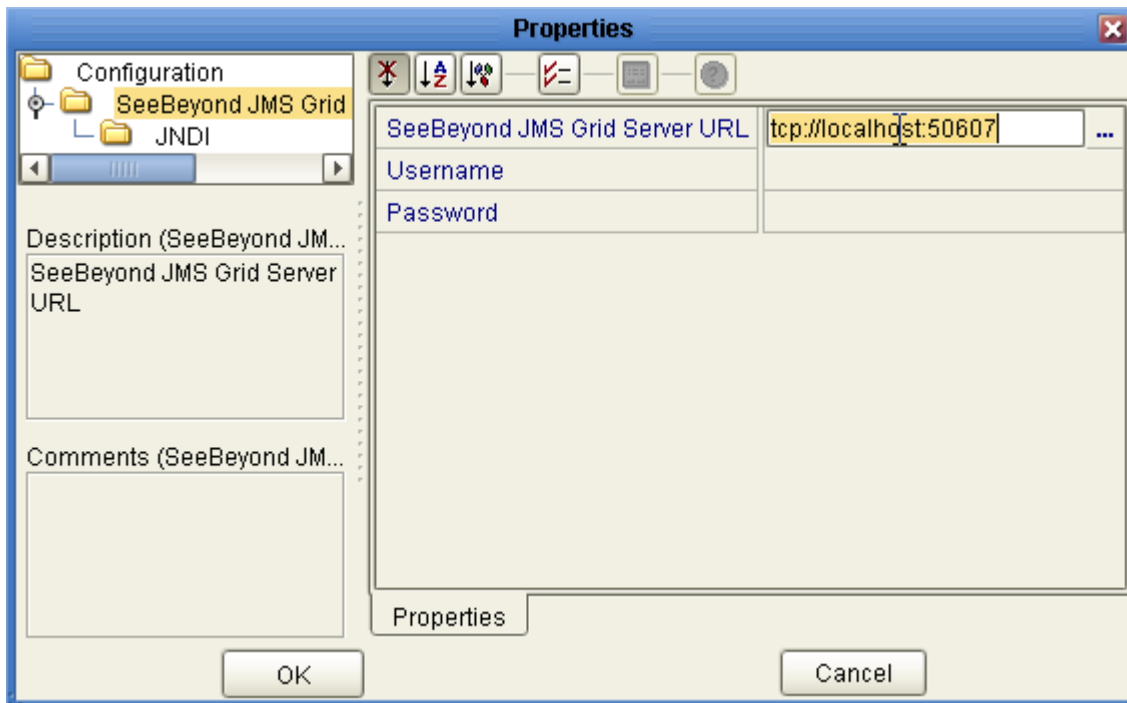
# Integration with Java CAPS (cont'd)

- Add a JMS Grid server to the Environment



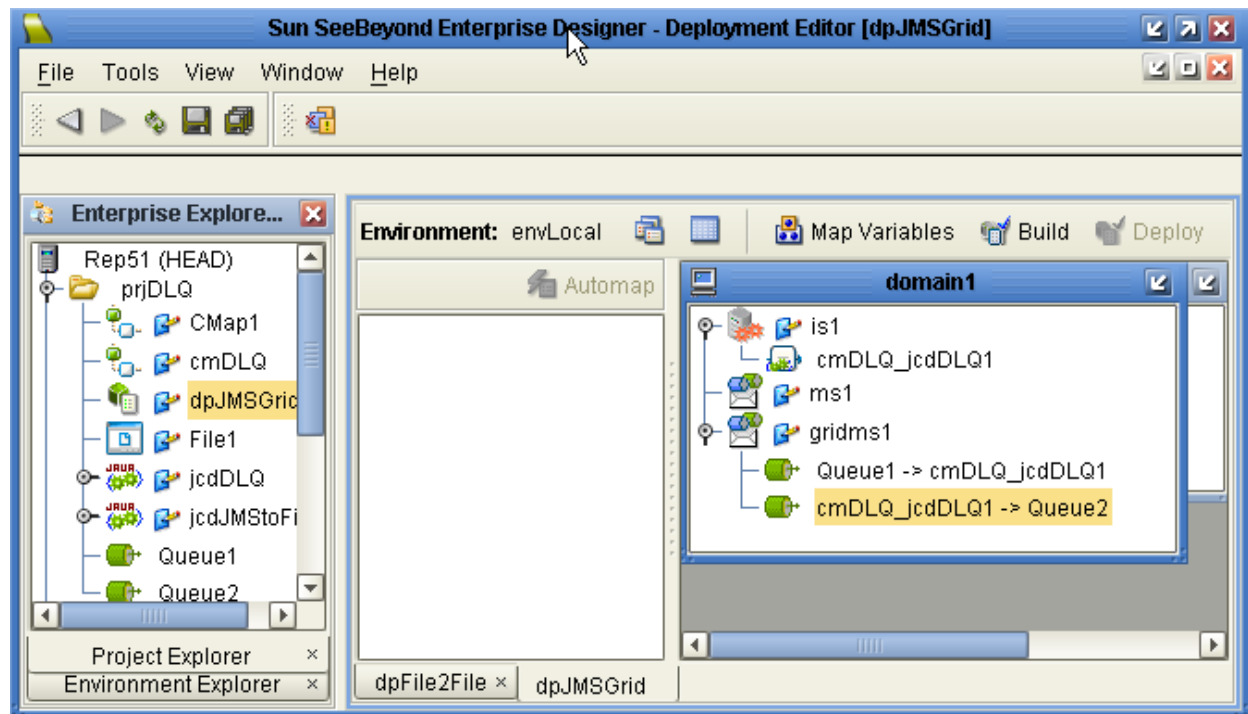
# Integration with Java CAPS (cont'd)

- Set the URL in the message server properties



# Integration with Java CAPS (cont'd)

- Project components are unaffected by decision to deploy to JMS Grid
- Deployment Profile maps components to JMS Grid



## Integration with Java CAPS (cont'd)

- Deployment Profiles are built normally
- EAR files are deployed normally
- Those JMS API destinations mapped to JMS Grid will use JMS Grid at run-time
- Enterprise Manager provides for viewing JMS API destinations

# Comparison with JMS IQ Manager

- Standard JMS API Features
  - Both support JMS API 1.1
  - Both support JMS API selectors
  - Both support throttling
  - Both support time-to-live

# Comparison with JMS IQ Manager

- Reliability Model
  - JMS IQ Manager provides single-machine reliability
    - Writes sync'd to disk before acknowledgement
    - Ensures messages aren't lost when using a single machine

# Comparison with JMS IQ Manager (cont'd)

- Reliability Model (cont'd)
  - JMS Grid provides multi-machine reliability
    - Messages are replicated to other servers in the cluster before acknowledgement, but not necessarily on disk
    - Provides improved performance but requires multiple machines
    - Option available for forcing syncs but decreases performance

# Comparison with JMS IQ Manager (cont'd)

- High Availability
  - JMS IQ Manager provides high availability through OS clustering
    - No built in clustering support
    - May be configured in active/passive configuration with OS clustering and shared disk
  - JMS Grid provides high availability features
    - Transparent failover and recovery
    - Automatic load balancing

# Comparison with JMS IQ Manager (cont'd)

- Performance
  - JMS Grid is up to 3 times faster than JMS IQ Manager depending on configuration
    - JMS Grid is faster when used in 2 server cluster or when working with non-persistent messages
    - JMS Grid is slower when forced sync is turned on

# Comparison with JMS IQ Manager (cont'd)

- APIs
  - JMS IQ Manager
    - Java (JMS 1.1) API available
    - C/C++ APIs available
    - COM API available
    - SSL support for encrypting the channel
  - JMS Grid
    - Java (JMS 1.1) API available
    - C/C++ APIs available
    - SSL support for encrypting the channel

# Comparison with JMS IQ Manager (cont'd)

- Special Features
  - JMS IQ Manager
    - Concurrency modes for controlling message order with individual JMS API destinations
    - Time dependency groups for controlling message order across multiple JMS API destinations

# Comparison with JMS IQ Manager (cont'd)

- Special Features (cont'd)
  - JMS Grid
    - Networks of clusters for automatic forwarding of select messages
    - Message selection using XPath on body of message
    - Sticky queues supporting a specific subscriber receiving all messages sent to a queue even when multiple subscribers

# Comparison with JMS IQ Manager (cont'd)

- Special Features (cont'd)
  - JMS Grid (cont'd)
    - Destination hierarchies and wildcard subscriptions for subscription to several destinations at once
    - Message compression option
    - HTTP tunneling option

# Beta and EAP Availability

- JMS Grid is available as part of the Java CAPS 5.1.1 Beta Program
- JMS Grid is part of our Early Adopter Program
- Contact Alex Andrianopoulos for details

## **Kevin Schmidt**

Director, Product Management  
Sun Microsystems, Inc.

626-471-6246

kevin.schmidt@sun.com

[www.sun.com](http://www.sun.com)

