GigaSpaces Spring Support
Bringing POJOs to the Grid

GIGASPACES

WRITE ONCE.
SCALE ANYWHERE.

Dennis Reedy
GigaSpaces Technologies

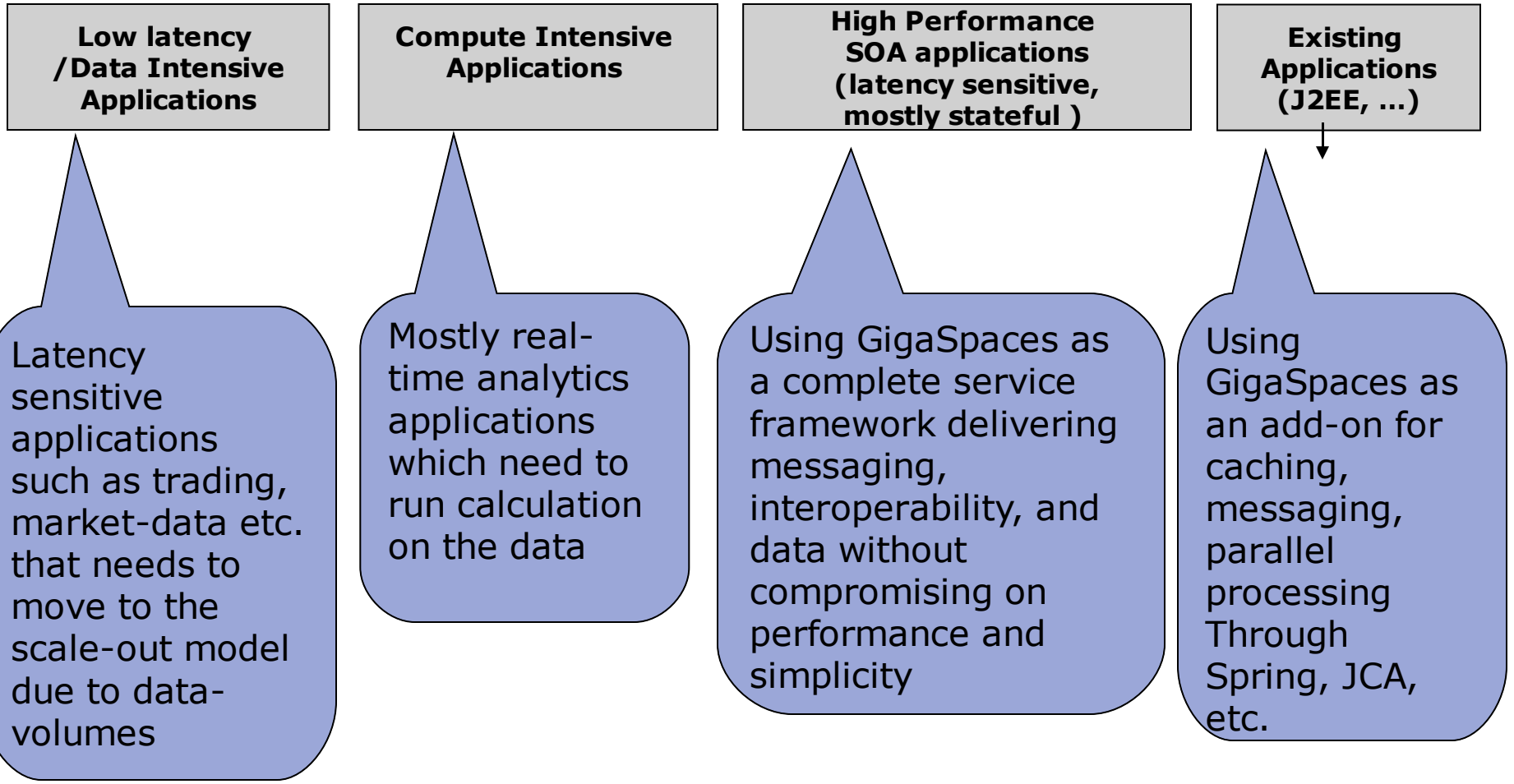
Presentation Takeaways

Scaling out challenges and how GigaSpaces addresses them

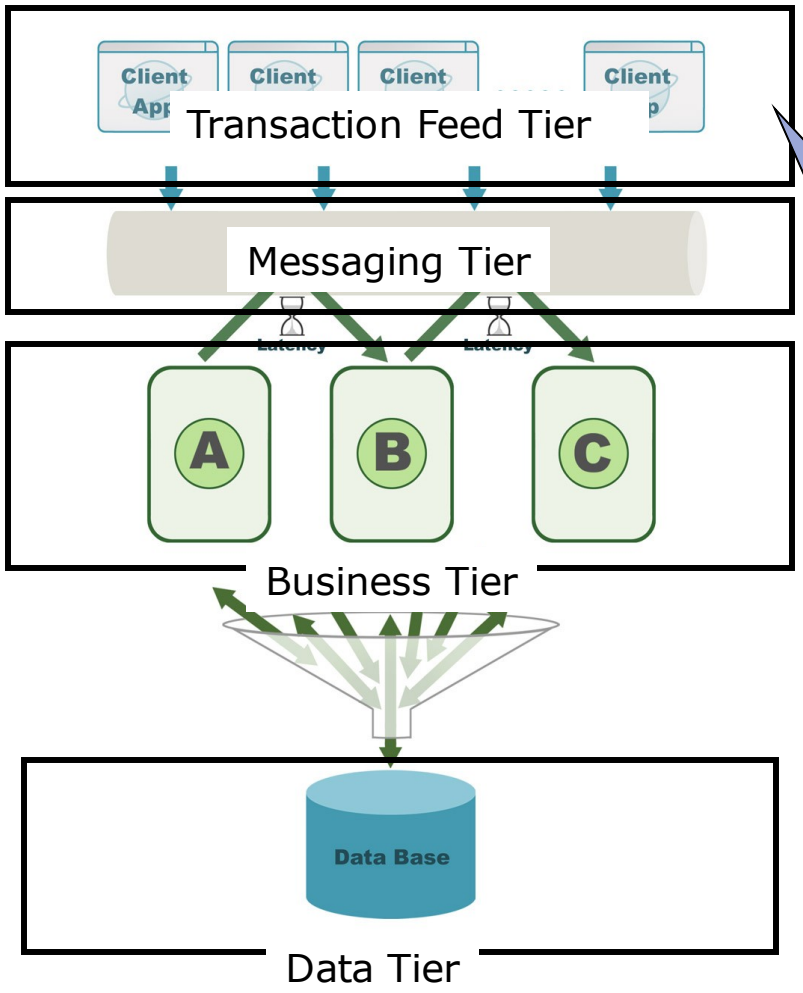
Combined benefit of GigaSpaces and Spring technologies

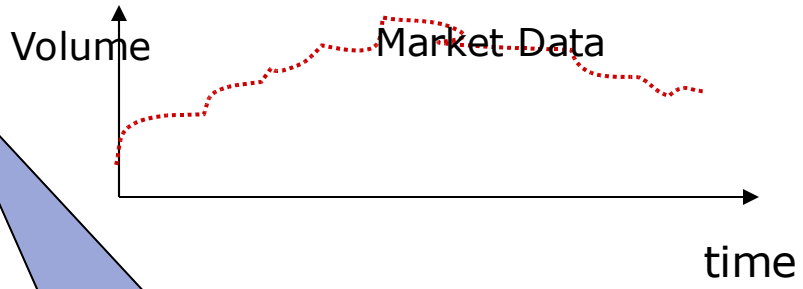
GigaSpaces Technologies Community Efforts

Who is using GigaSpaces?



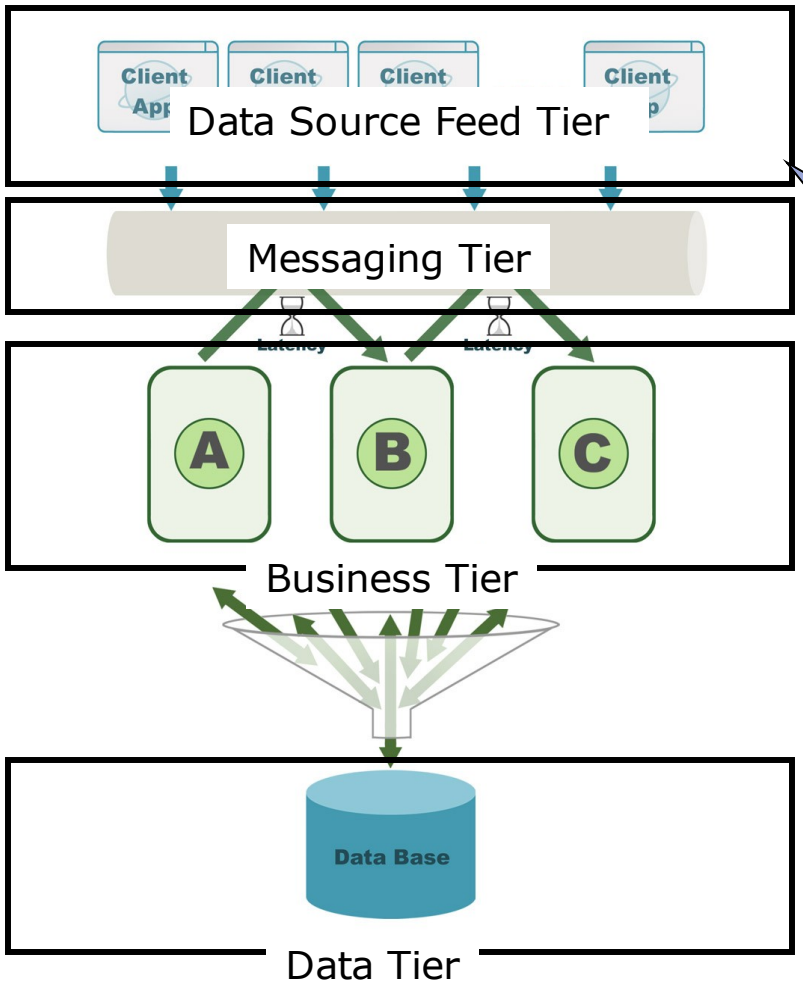
Transactional Applications (Trading, FX, ...)

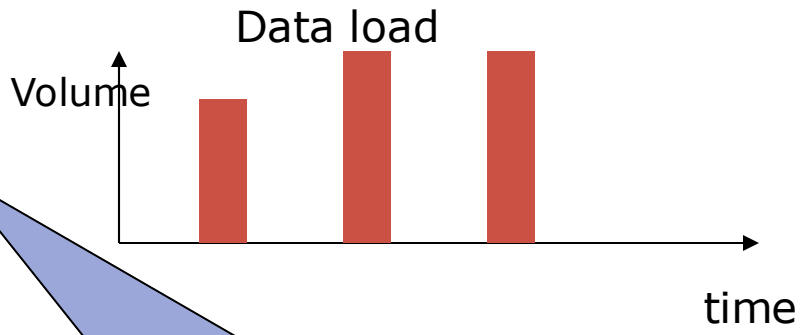




Transactional Applications
-Transaction comes from messaging
-Continuous high rate
-Low latency sensitive
-Processing normally includes, validation, matching, routing
-Reliability requirements are very strict

Analytic (Batch) Applications:





Analytic (Batch) Applications
-Feed - files, messaging
-Bursts of feeds
-Not sensitive to latency
-Processing normally includes, validation, calculation, workflow
-Reliability requirements are less strict (data can re-calculated, operations can be batched)..`

Common bottlenecks with existing architecture

- **Data** – I/O overhead,
- **Messaging** – I/O Serialization, Workflow increase that overhead per hop.
- **Processing** – CPU, Parallelization
- **Architecture** – Multiple Clustering models, Multiple tiers, Silos, Distributed transactions

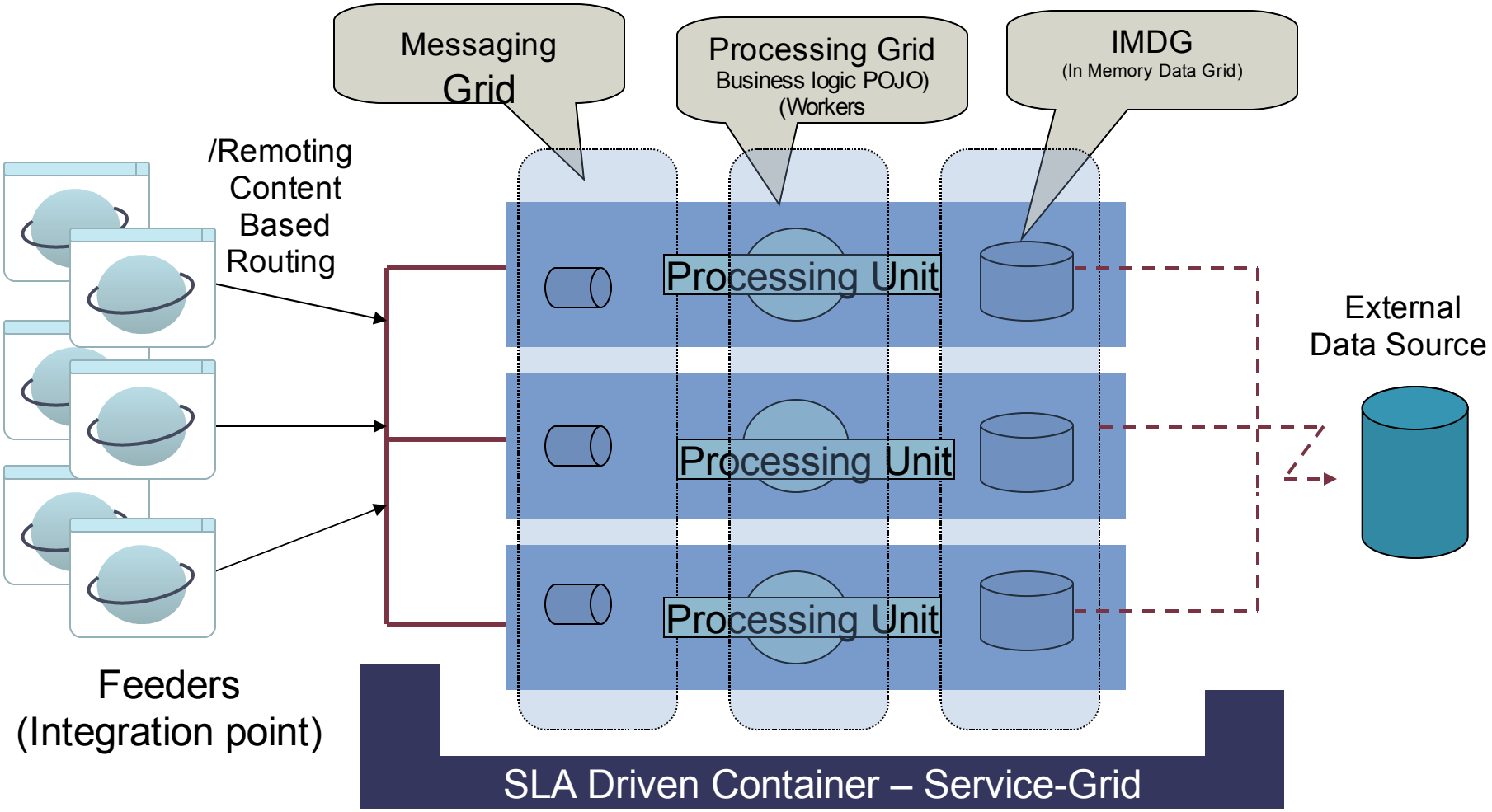
- **Result**
  - Limited Performance, latency
  - Limited scalability
  - Limited reliability – most failure happens between tiers

Space Based Architecture in a Nutshell

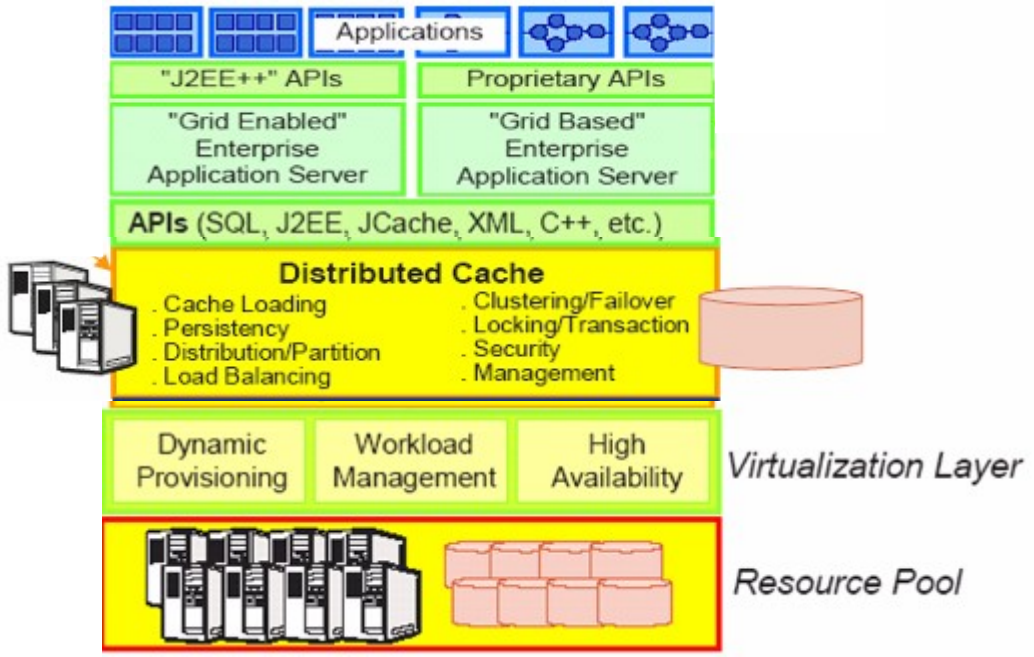
- Defines architecture for scaling-out stateful applications
- Utilize a common middleware for handling messaging, data and processing a.k.a “virtual middleware”
- Based on JavaSpaces

- Utilize the ServiceGrid to enable
  - Wiring of the business logic with the middleware through distributed dependency injection
  - Policy based infrastructure
  - Single point of access for managing distributed application

SBA End to End Solution for Scaling out Stateful Applications



The Gartner View



*"…new-style application server products will compete against J2EE vendors and Microsoft for the **high-end OLTP** and the mass-market business application platform projects, gaining at least **20 percent** share of the total market." **

Source Gartner

Why Spring

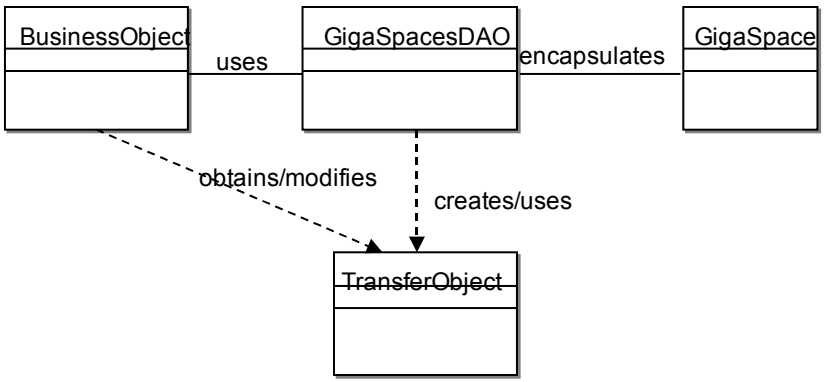
- Spring is about application programming model
- Provides a POJO abstraction that decouples your code from platform code
- More general than Java EE programming models
  - No assumptions about underlying environment
  - Can accommodate radically different platforms underneath
- Goal
  - Decouple programming from deployment infrastructure
  - Allow choice of most appropriate runtime without re-architecture of business logic
  - One of Spring's main purposes is to facilitate choice

Relevant Spring Services

- Injection
  - Container can inject proxies interacting with space
    - Application code depends on business interface
    - Space based API is concealed behind POJO invocation
  - Uses familiar Spring remoting concepts
- Transaction management
  - Declarative and programmatic transaction management using Spring APIs
    - Propagation.MANDATORY, Propagation.NEVER, Propagation.NOT_SUPPORTED, Propagation.REQUIRED, Propagation.REQUIRES_NEW, Propagation.SUPPORTS
  - Choice of space-aware transaction managers underneath
    - XA/JTA transaction
    - Jini Transaction Manager (Mahalo)
    - Local Transaction Manager

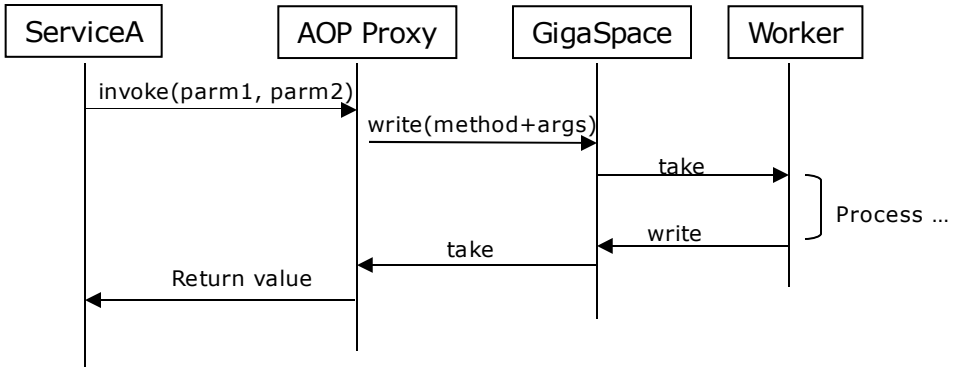
Using the Space through a DAO

- Familiar Spring abstraction with convenient superclass for GigaSpaces access
- Developers focus on business objects (pojos), not `Entry` objects
- Space matching semantics preserved through POJO<->Entry conversion
  - Through Annotations or conversion helpers



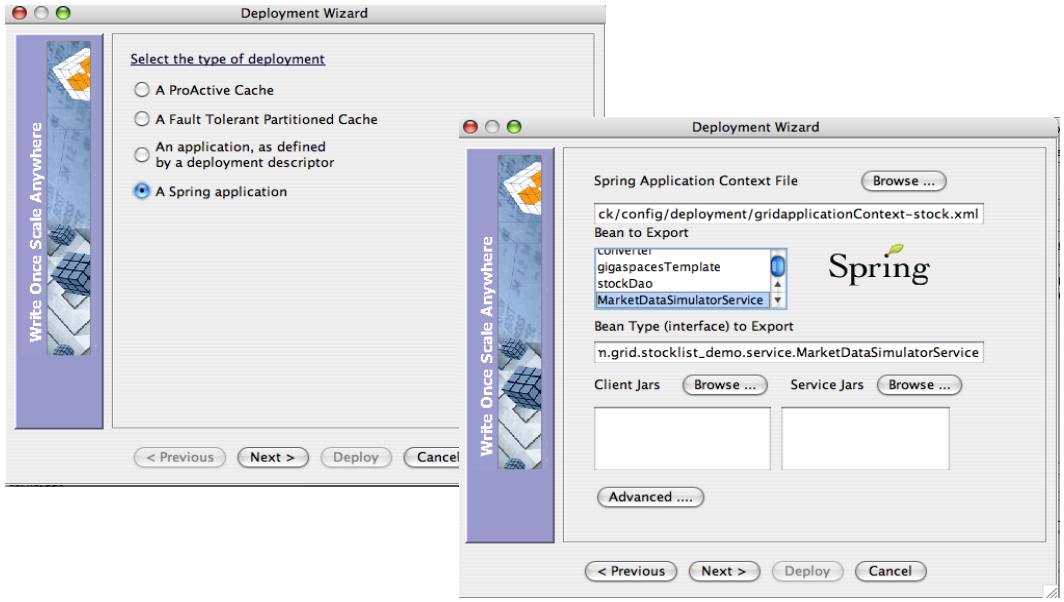
Spring Remoting using a Space

- Declarative remoting for a POJO
- Method invocations through the space
- Dynamic scalability
- Workers execute methods

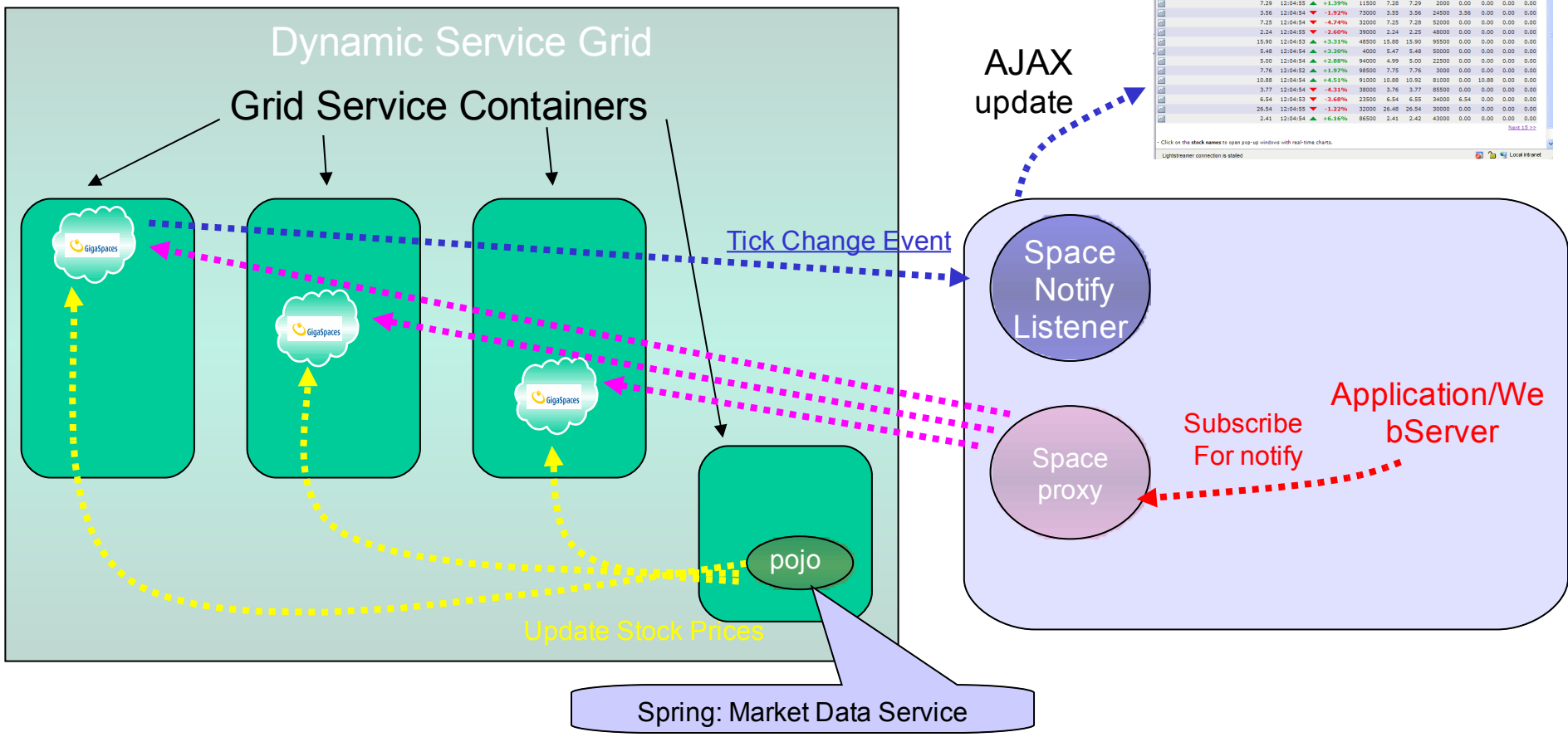


Integrate with Service Grid

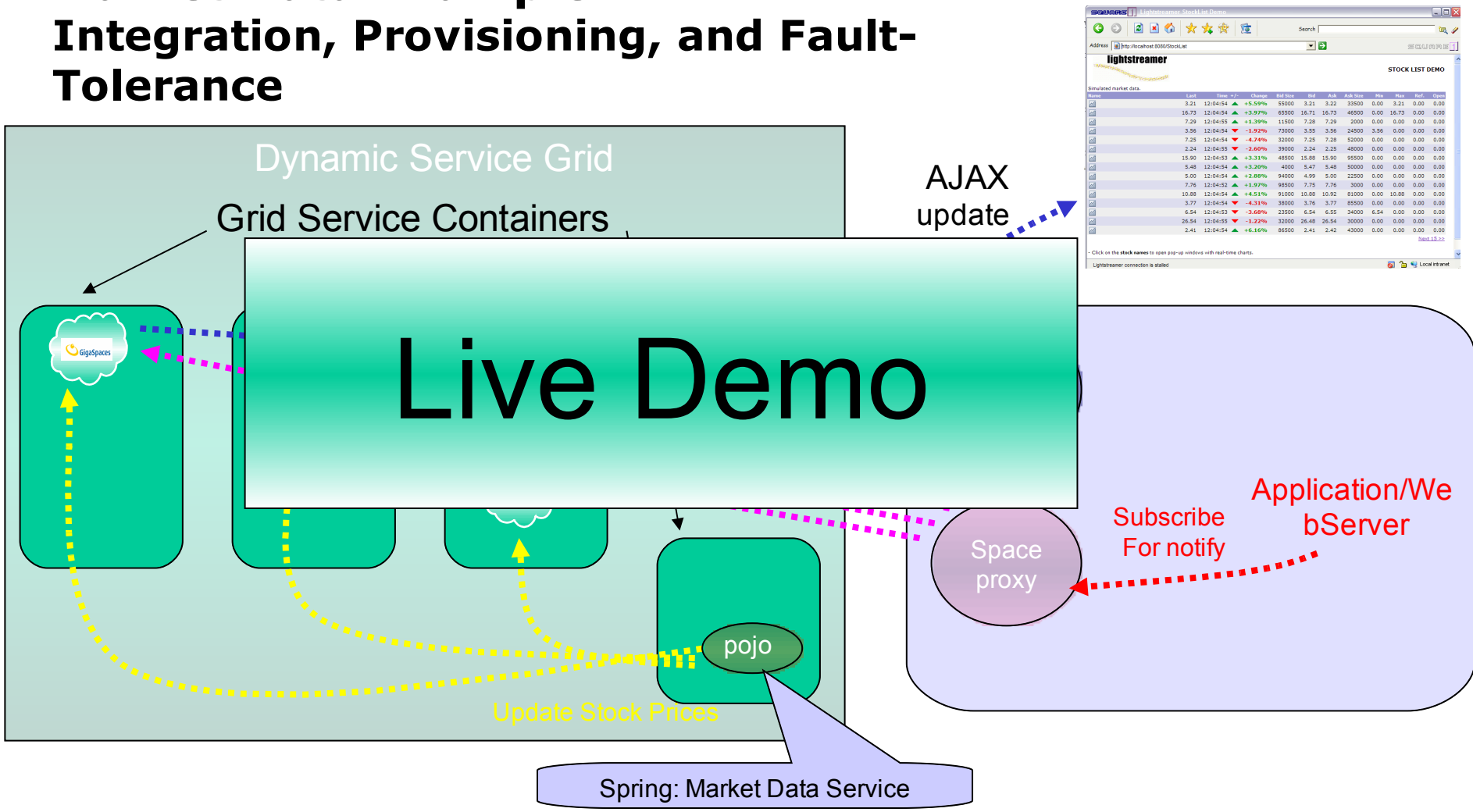
- Enable automated deployment and dynamic scalability
- Spring beans automatically exported as dynamic Jini service
- Spring technology infused through Platform policy interactions



Market Data Example: Integration, Provisioning, and Fault-Tolerance



Market Data Example:
Integration, Provisioning, and Fault-Tolerance

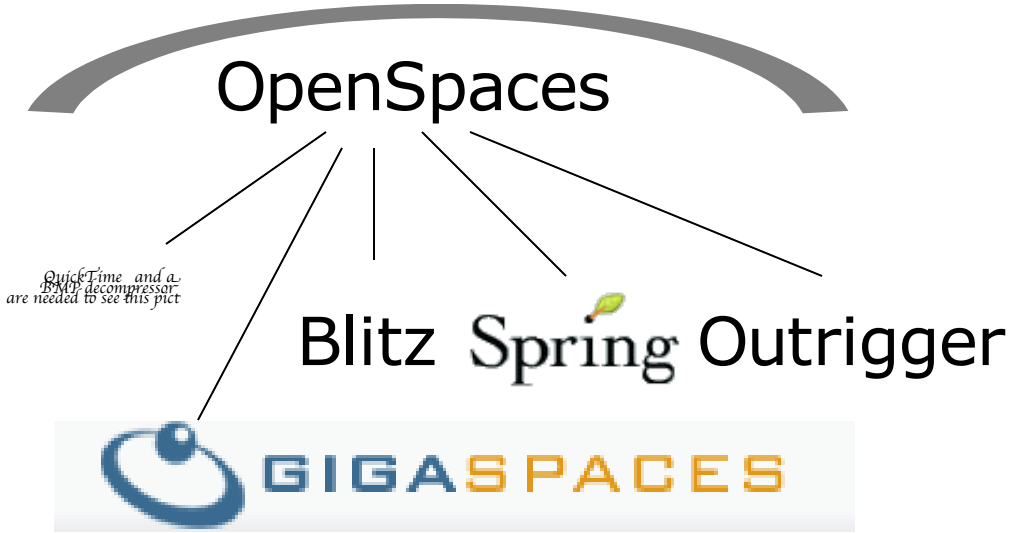


Who is using it?

- More than 60 customers world wide including major financial and Telco providers using GigaSpaces in production
- Spring support is already in One of the Largest Global Investment Banks

GigaSpaces Community Efforts

- OpenSpaces.org
  - Community to develop, inspire, nurture and move Space Based Architecture forward
  - Greater visibility - at an industry focus
  - Scope
  - When
    - End 3Q-> 4Q



Reference

- Spring Modules
  - A free open source project is now officially as part of the Spring modules framework

  http://springmodules.dev.java.net

- Release 5.1 of the GigaSpaces Enterprise Edition contains built in support for Spring

  http://www.gigaspaces.com

Q&A