



## GemFire Enterprise Data Fabric for Risk Management

## EXECUTIVE SUMMARY

Enterprise Risk Management relies upon fast, coordinated access to data from multiple applications across the organization. Data must be instantly available to any requesting user or application at all times, with technical strategies in place to deal with offline data sources, unavailable servers, or excessive load volumes. This paper presents the GemFire Enterprise Data Fabric (EDF) from GemStone Systems. It details approaches to implementing GemFire in a variety of technical architectures and discusses caching and communication strategies that ensure high availability, performance and scalability. This paper is appropriate for a technical audience charged with supporting risk management in enterprise information systems. Readers may wish to review a business examination of this subject in "Enterprise Data Fabric: A Critical Foundation for Risk Management," also available from GemStone Systems.

## DATA SERVICES IN SUPPORT OF RISK MANAGEMENT

Financial service providers implementing risk management solutions are faced with the necessity to share and distribute large amounts of information across multiple systems. A comprehensive risk management platform must satisfy several key requirements:

- Make risk-related information instantaneously available to risk computation applications and consumers of risk information.
- Aggregate information from several different sources to integrate factors affecting credit, market, and operational risk.
- Move from operational silos to more fluid information sharing and flow between applications and departments.
- Deploy a system that can handle large data volumes and computations to support sophisticated financial engineering tools.

All of these risk management concerns focus on how data is handled within the enterprise architecture. An Enterprise Data Fabric (EDF) is an integral part of planning an application infrastructure to ensure that data is rapidly and consistently made available in a variety of formats on multiple hardware and software platforms, and in different business applications. Such an infrastructure also facilitates instant correlation of risk levels across the organization based on external events.

## GEMFIRE EDF

A data fabric in support of enterprise risk management must support a number of fundamental requirements:

- Ensure high performance through rapid data delivery data on demand to risk applications across a distributed network
- Maintain high availability of data in the face of heavy loads or network interruptions
- Scale to handle large data volumes and high numbers of requesting applications or clients without data latency
- Integrate data across different trading and risk management applications by presenting a universal virtual source and format for access that hides data source differences
- Native support for both Java and C/C++/C# environments
- Flexibly interact with distributed data sources, interfaces, and requesting applications as network architectures change

GemFire EDF offered by GemStone Systems acts as a corporate data fabric, binding enterprise sources of data and consumers of information. In a risk management architecture, GemFire delivers high performance, high availability, instant responsiveness and scalable data distribution. Risk/IT architects can use GemFire to build, enhance or simplify access, distribution, integration, and management of risk-related information. A GemFire-based solution offers a broad range of functions that include high-performance data caching, data distribution, data virtualization, high availability, data integration, and continuous data analysis. These functions can be effectively utilized in an architecture that caters to enterprise risk management.

## GEMFIRE EDF- DESIGN APPROACHES FOR RISK ARCHITECTURES

Flexibility to operate on multiple hardware platforms and system topologies is one of the key requirements for a data fabric. GemFire EDF can be utilized in different design configurations to achieve architectural goals consistent with application requirements and available data while conforming to enterprise standards. For organizations seeking to improve existing architectures, GemFire works with third-party platforms such as application servers, integration servers, and database management systems. It provides improvements in performance, scalability, and data availability without requiring extensive recoding of the underlying platform. Similarly, GemFire can augment (or in some cases replace) application-specific approaches to caching and data management that have been designed with proprietary technologies. The goal in all cases is to provide a more robust, reusable data framework for better performance and scalability in risk management. Companies designing new risk management infrastructures can incorporate GemFire as a central component to simplify and speed development while ensuring peak performance as new applications are created and rolled out.

## HIGH-PERFORMANCE DATA DISTRIBUTION AND CACHING DESIGN PATTERNS

Using GemFire, different data distribution and caching topologies can be deployed to suit a variety of application hardware environments. This section presents three different scenarios and discusses relevant solution architectures that can be applied to each.

### *Scenario 1: Hierarchical Caching Architecture For Grid-like Deployments*

When risk applications are deployed over a distributed network like a grid, multiple engines may be used to perform different kinds of operations. Some applications deal with fast-changing data as they perform theoretical computations and other forms of pre-processing, with the results fed to applications that compute metrics used by risk information clients. In this scenario, GemFire can be used to deploy a data grid (Figure 3a), with a set of client and server caches. The client caches ("edge caches") are responsible for positioning data close to the risk applications directly serving the risk information consumers. These client caches connect to one or more server caches that interact directly with the data sources and are positioned near the applications that deal with fast-changing data. A miss of the client cache results in the request being delegated to the server cache. A miss on the server typically results in the data being fetched from the origin data source. In this configuration each client cache holds a facet of the risk data originating from one or more server caches. With directed inter-cache communications, traffic to the client caches is dramatically reduced, promoting scalability. The server cache is typically deployed on a more powerful machine and reduces unnecessary traffic to the data source. The hierarchical cache configuration can be loosely coupled, where the client cache and the server cache do not participate as part of a single distributed system. In fact, client application deployments themselves could be in cluster and deployed using a distributed system of their own. The servers likewise can be connected to one another in a single distributed system, mirroring data with each other for high availability and load balancing requests from clients.

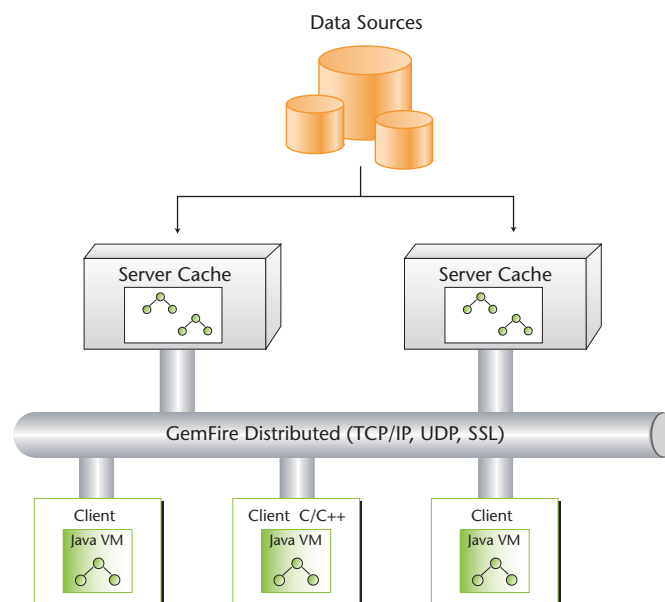


Figure 3a: GemFire Client-Server Cache Architecture for Distributed Risk Applications

**Scenario 2: Pee-to-Peer Distributed Architecture**

For financial institutions using a distributed infrastructure for their information-processing and risk management applications, GemFire can be deployed in a peer-to-peer configuration (Figure 3b) enabling a truly distributed data-grid that supports risk applications. Each application node in the grid can hold information relevant to it in a local cache. GemFire is based on the JCACHE standard for distributed caching[1] for Java and has a native C++ implementation as well for C/C++/C# environments. It enables applications to retrieve information not found in their local cache from other caches in the distributed network, avoiding data latency issues. Application nodes that cache similar information can communicate synchronously with each other for sharing as well as updating cache information. Alternatively, two caches with data regions not related to each other can be deployed in a loosely-coupled fashion, thereby reducing network traffic when cache updates are published. Caching services can be instantiated on grid nodes to provide on-demand data provisioning and accommodate load-balancing and work scheduling by the grid job scheduler. GemFire also complements RDMA (Remote Direct Memory Access) topologies like InfiniBand, and virtualizes data across interconnected nodes, creating a single system image of shared data that elegantly spans the system area network.

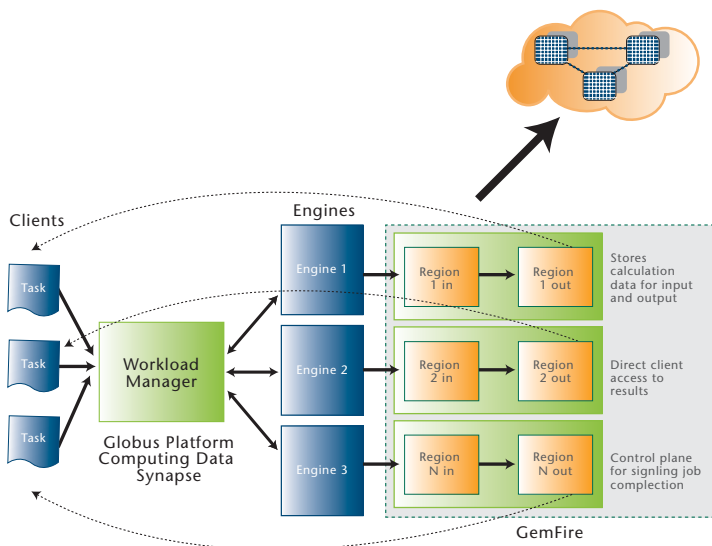


Figure 3b: GemFire based Peer-to-Peer Cache Topology for Grid-based Risk Applications

**Scenario 3: Risk Applications Deployed on Multiprocessor-Based Systems**

In environments where risk management applications are deployed on a multiprocessor machine such as a Sun 4-way or 8-way box, operating system shared memory space configuration can be used to provide caching and data sharing across multiple processes. GemFire is unique in its ability to accommodate both C and Java applications and offers support for C/C++ applications through native shared memory maps, linked-lists, queues, and arrays. The behav-

ior of these collections is implemented in C for interoperability, so that C applications use a function library to access them, and Java applications see them as direct-shared objects. GemFire also supports optimized garbage collection and optional disk paging for shared memory objects to avoid memory overflow conditions.

This high-speed shared object caching mechanism is highly valuable when risk applications are co-located with large amounts of data that need to be shared across multiple risk engines. The support for different application formats (C, C++, Java, and XML) also enables data to be distributed efficiently across the application processes, avoiding repeated fetches from the data sources.

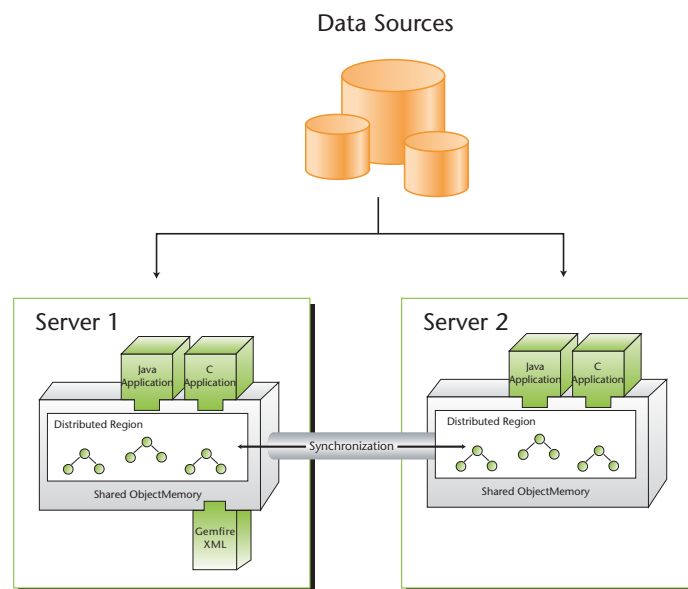


Figure 3c: GemFire Shared Memory Architecture for Risk Applications on Multiprocessors

## OTHER ARCHITECTURAL CONSIDERATIONS

In addition to the data caching and distribution approaches discussed above, there are other technical issues that merit consideration from an overall data infrastructure standpoint. These include:

**High Availability:** In a distributed network of applications, data must be available at all times to permit uninterrupted operations. Critical functions such as risk management must be able to withstand and adapt to a malfunctioning node or inaccessible data source. GemFire contains features that ensure high availability in the face of point failures. Cache regions can be mirrored to different nodes to ensure that risk application data is available even when one application node fails. Data can also be partitioned across several risk application machines to enable scalable deployments with large data volumes. GemFire also supports a hot-standby option, which allows seamless fail-over to a secondary server when the primary GemFire cache server is

down. In addition, in-memory data regions can be configured so that they are persisted to disk and can be recovered upon application restart. Data held in memory can be paged to disk to avoid out-of-memory errors. Data availability is a critical foundation of any real-time risk infrastructure.

**Data Consistency:** Effective enterprise risk management requires consistent data across all associated applications. Data must be simultaneously updated with respect to back end data sources to maintain a constant version among all distributed data caches. GemFire ensures consistency of data cached in the different nodes of an application network through flexible synchronization and distribution policies. System architects can choose an approach best suited to the nature of their risk application environment. In some instances it is sufficient to have asynchronous communication between cache data regions, wherein changes to data in one node need not be immediately transferred to another (for instance, when running a what-if scenario on an application node). For such scenarios, GemFire supports distribution with no acknowledgement, which guarantees the highest response time and throughput. However if caches require synchronization at all times, architects can use a synchronous communication model (distribution with acknowledgement and with or without global locking). A combination of both approaches (synchronous for certain critical data regions and asynchronous for others) is typically employed as a strategy that balances data consistency and performance. Another dimension of data consistency involves keeping cache data consistent with any changes in the back end data sources. For example, if there are changes to the market data that is being fed to the risk management applications, the in-memory data caches must be updated to ensure accurate and up-to-date calculations. GemFire works with enterprise messaging systems through pre-packaged connectors to update the in-memory data stores with changes from the back end systems.

## AN ADVANCE OVER TRADITIONAL TECHNOLOGIES

Many software approaches have been implemented for managing data in enterprise architectures. Disk-based relational database systems and their extension to in-memory databases, messaging systems, and other data sharing solutions have helped to make data management more efficient, but all have suffered drawbacks when faced with the demanding availability, scalability, and performance requirements of large-scale risk management.

Relational database management systems (RDBMS) have traditionally been disk I/O intensive. Over time, these I/O issues have been addressed to a certain degree through advances in database technology, but as a consequence RDBMS has become an extremely CPU intensive technology - as demonstrated by the need to run the database on its own server platform, separate from the applications and software it supports. In-memory databases solved the CPU problem by replicating disk databases in memory, with optimized indexing that is reliant on RAM quantity rather than CPU speed. But both disk and in-memory databases face difficulties in dealing

with data distribution over a network of applications. They remain centralized point sources of data in a decentralized application environment. In-memory databases rely on a single massive memory area in much the same way that RDBMS systems rely on large disk areas... increasingly contradictory to the realities of smaller memory areas distributed across multiple nodes in a network. Consequently, these technologies fail to support concepts such as sophisticated data partitioning based on business logic and therefore cannot support large-scale distributed architectures.

Message servers face the opposite problem. They efficiently distribute data, but are not suited for large-scale on-demand data access requirements. Applications cannot retrieve data from message servers and have to rely on data (as messages or events) being pushed to them. This is part of the reason why messaging has always been a paradigm that resonates better in asynchronous architectures (push-based) as opposed to real-time, client-driven systems.

The power of the GemFire EDF architecture lies in its ability to provide the distribution capabilities of messaging without sacrificing the on-demand, instant data access capabilities of in-memory databases. GemFire is flexible enough to support data dependencies between applications and facilitates intelligent communication to reduce network traffic.

## **THE VALUE IN ADOPTING GEMFIRE EDF ARCHITECTURE FOR RISK MANAGEMENT**

Data infrastructure plays a significant role in the success of risk management processes and strategies. GemFire acts as an enterprise-wide data management service bringing together multiple data sources and applications in a distributed infrastructure. It provides functionality in several critical areas to support the needs of enterprise risk management. Among these are:

- Positioning data close to associated risk computation centers to reduce data latency
- Flexible caching and distribution topology supporting cost-effective distributed data architectures such as grids and blades
- Fast reliable access to data from multiple data stores (market data, reference data, etc.)
- Integration into any kind of heterogeneous environment comprising C/C++ applications, Java applications, databases, message servers, application servers, etc.
- Real-time information exchange between risk, trading, and pricing systems
- In-memory risk servers that facilitate real-time risk information delivery to clients
- Scalable data infrastructure that supports surges in trading/risk data volumes as well as an increase in number of computational nodes.
- Reduced network traffic and improved application resilience to data source unavailability

By implementing the GemFire data fabric in a new or existing risk management design, technical architects can better support the business needs of the organization while ensuring higher reliability, easier application coding and maintenance, and improved scalability of all systems.

More information on GemFire is available from GemStone Systems at [www.gemstone.com](http://www.gemstone.com).

## ABOUT GEMSTONE SYSTEMS, INC.

GemStone Systems is a privately held infrastructure software company that provides the GemFire Enterprise Data Fabric (EDF) for enterprise business architects and data infrastructure managers that are building, enhancing or simplifying access, distribution, integration and management of information within and across the enterprise. Founded in 1982, and with over 200 installed customers, GemStone is recognized worldwide for its unique competency and patented technology in object management, virtual memory architectures, high-performance caching, and data distribution technologies.



**Corporate Headquarters:**

1260 NW Waterhouse Ave., Suite 200 Beaverton, OR 97006 | Phone: 503.533.3000 | Fax: 503.629.8556 | [info@gemstone.com](mailto:info@gemstone.com) | [www.gemstone.com](http://www.gemstone.com)

**Regional Sales Offices:**

New York | 90 Park Avenue 17th Floor New York, NY 10016 | Phone: 212.786.7328

Washington D.C. | 3 Bethesda Metro Center Suite 778 Bethesda, MD 20814 | Phone: 301.664.8494

Santa Clara | 2880 Lakeside Drive Suite 331 Santa Clara, CA 95054 | Phone: 408.496.0242

Copyright© 2005 by GemStone Systems, Inc. All rights reserved. GemStone®, GemFire™, and the GemStone logo are trademarks or registered trademarks of GemStone Systems, Inc. Information in this document is subject to change without notice.