

KEY FEATURES

- *distributed caching with multiple topologies and cache coherency policies*
- *native C++ bindings to avoid inefficiencies*
- *highly parallel distribution with optimized transport*
- *pluggable transport with support for TCP and reliable multicast*
- *no unnecessary user-kernel crossings with no contention or context-switching in the data distribution path*
- *high data availability through replication and mirroring*

GemFire Enterprise — C++: Data Management for Ultra-High Performance and Low Latency Applications

TECHNICAL DATASHEET

Certain kinds of applications, like trading/order management in capital markets, rely on blazing data access and distribution speeds. In such scenarios, profitability is a direct function of the speed of data retrieval, updates and propagation, wherein even milliseconds of latency can be a significant competitive advantage and can help prevent monetary loss.

GemFire Enterprise — C++ offers a high-performance data management solution for such environments. It provides an efficient distributed data caching infrastructure that is based on a implementation that utilizes the kernel level APIs to facilitate extremely high-speed data transport. Through features like mirroring, selective replication, various cache consistency models, it provides the enterprise application developer with all the necessary dials and knobs to create fine-tuned, massively parallel, distributed data management solutions. This data fabric has built-in mechanisms for actively managing scarce system resources like memory, sockets, ports etc. to ensure that there is no degradation in system performance due to the usage of these resources.

TECHNICAL FEATURES

High performance data caching: GemFire Enterprise — C++ is built on a high-performance distributed caching framework that guarantees incredibly high get and put rates from and into the cache respectively. The current release of the product (v1.0) supports C++ access APIs, with support for a Java and .NET/C# API planned for a future release. GemFire Enterprise — C++ efficiently manages a distributed set of caches and delivers synchronized data to any node in a cluster or to remote sites, allowing massively scalable, load-balanced architectures. Upon a request from an application, GemFire Enterprise — C++ can perform a search across different nodes on a distributed network and return the requested information back to the client application with almost zero latency. Other cache management features include the ability to expire data out of the cache, connectivity to backend enterprise data sources to load data into the fabric and the ability to perform write through caching. Applications can also utilize GemFire Enterprise — C++ for localized caching, wherein objects are not made visible to the distributed system and is used by a specific application for high read/write throughput.

A typical deployment topology comprises of a set of caches that are deployed on a peer-to-peer network, with the option of configuring one of the nodes as a cache server that can serve out relevant data elements to other caches on the distributed system. Data held in the cache can be invalidated through settings like time-to-live (TTL) and/or idle-time or can be evicted programmatically.

Low Latency Data Distribution: Enterprise class applications operate under a variety of network infrastructures and enabling the GemFire data fabric to work in these environments allows applications to use the fabric optimally. A highly scalable data distribution layer forms the heart of the GemFire Enterprise — C++ system. This layer delivers in-line message processing with no context switching or contention in the distribution path. It also provides a pluggable transport option and can accommodate both TCP and reliable Multicast distribution protocols. Reliable Unicast (UDP) is planned for the next major release.

As and when objects in the cache get updated, changes can be distributed in two modes - optimistic or pessimistic. In the optimistic mode, changes to an object are relayed out to other cache instances and the sender does not block on that thread while waiting for acknowledgements. For scenarios where dirty-reads are unacceptable, a pessimistic protocol can be adopted, wherein the sender of an update waits for acknowledgements from all recipients before resuming activity on its thread. Since GemFire Enterprise — C++ is aware of cache membership and the objects that are relevant to each cache through a transparent mechanism called 'interest-lists', updates are distributed only to relevant cache members thereby avoiding network bottlenecks even under point-to-point settings. Further, data distribution can also be configured to transport entire objects upon updates or have only the cache-keys replicated, with the object delivery happening only upon an explicit client request.

Applicability as a messaging bus: GemFire Enterprise — C++ provides a mode in which a node can subscribe to receive all

data updates for a particular region without ever caching anything. In an enterprise system that includes data sinks (User Interfaces are a good example), this allows the system to produce very responsive event processing applications without the need to cache anything on the node.

High Availability: High availability of operational data in the data fabric is of utmost importance to the applications that rely on it. GemFire Enterprise — C++ provides a variety of means to ensure data availability in the system. Through a combination of mirroring/replication of data and the use of the membership management API to keep track of mirrors in the data fabric, GemFire Enterprise — C++ guarantees that cached data will be available in the system. A replicated node acts as a mirror-image of the primary node and all data changes that happen on the primary are communicated to the replicated node instantaneously. In case the application controlling the primary cache fails, the replicated node can still hold the data and guarantee the availability of the same to the other distributed nodes. When the failed application recovers, it can restore its state by reloading the cache from its replica. Being a memory-based system, GemFire Enterprise - C++ also isolates applications from backend data source unavailability and/or network failures as the data held is close to the consuming application. Application callbacks provide connectivity to backend data sources that can be used to store data to persistent data stores. Future releases will include the ability to overflow or persist data to disk allowing for extremely fast recoverability of data in case of system crashes and also the ability to scale each node of the system to hold much higher volumes of data

System Management: An enterprise ready application needs the ability to monitor, inspect and control elements of the data fabric in order to react to fast changing events in the enterprise. The ability to inspect the data fabric also helps application developers debug performance and scalability issues when the applications are under development. GemFire Enterprise — C++ provides many in-built statistics

and metrics that can be viewed through the graphical GemFire console in real time. The application developer can additionally create their own statistics using the API to incorporate monitoring of application elements and view them through the console in real time. GemFire Enterprise — C++ also includes VSD (Visual Statistics Display), which allows the creation of templates for customized views of metrics across multiple system members.

SYSTEM REQUIREMENTS

Sample performance benchmarks may be obtained for this product by contacting sales@gemstone.com.

Operating System	Minimum Platform	RAM	Swap Space	Disk Space
Windows	Intel Pentium Class (>299 MHz)	256 MB RAM	256 MB of virtual memory	127 MB
Linux		256 MB RAM	256 MB of virtual memory	171 MB

Operating System	Kernel version	glibc version
Suse Linux Enterprise 8	2.4.21-241-default	2.2.5
Suse Linux Enterprise 9	2.4.21-215-default	2.3.2
Red Hat Enterprise ES 3	2.4.21-9.0.1.ELsmp	2.3.2
Red Hat Enterprise AS 3	2.4.21-9.0.1.ELsmp	2.3.2

Corporate Headquarters:

1260 NW Waterhouse Ave., Suite 200 Beaverton, OR 97006 | Phone: 503.533.3000 | Fax: 503.629.8556 | info@gemstone.com | 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.