GEMSTONE®

GEMFIRE®

ENTERPRISE

Early Access Features

Version 5.7

September 2008

Send comments on this manual to docs@gemstone.com

INTELLECTUAL PROPERTY OWNERSHIP

This documentation is furnished for informational use only and is subject to change without notice. GemStone Systems, Inc. assumes no responsibility or liability for any errors or inaccuracies that may appear in this documentation.

This documentation, or any part of it, may not be reproduced, displayed, photocopied, transmitted, or otherwise copied in any form or by any means now known or later developed, such as electronic, optical, or mechanical means, without express written authorization from GemStone Systems, Inc.

Warning: This computer program and its documentation are protected by copyright law and international treaties. Any unauthorized copying or distribution of this program, its documentation, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted under the maximum extent possible under the law.

The software installed in accordance with this documentation is copyrighted and licensed by GemStone Systems, Inc. under separate license agreement. This software may only be used pursuant to the terms and conditions of such license agreement. Any other use may be a violation of law.

Use, duplication, or disclosure by the Government is subject to restrictions set forth in the Commercial Software - Restricted Rights clause at 52.227-19 of the Federal Acquisitions Regulations (48 CFR 52.227-19) except that the government agency shall not have the right to disclose this software to support service contractors or their subcontractors without the prior written consent of GemStone Systems, Inc.

This software is provided by GemStone Systems, Inc. and contributors “as is” and any expressed or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall GemStone Systems, Inc. or any contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage.

COPYRIGHTS

PATENTS

GemFire is protected by U.S. patent 6,360,219. Additional patents pending.

TRADEMARKS

GemStone Systems, Inc.
1260 NW Waterhouse Avenue, Suite 200
Beaverton, OR 97006

This document describes the early access features included in this release.

The functionality documented here is included as part of an early access program, with specific, targeted customers in mind. It may not yet be fully implemented, and it has not been subject to the full set of rigorous testing procedures that our General Availability software receives. If you wish to use this feature prior to its General Availability, we highly advise you to first inform your GemStone technical representative so additional support can be provided and any new requirements can be considered for implementation into the full release.

GemStone Systems, Inc. provides these features without warranty, and does not currently endorse their use in any critical applications.

For information on general access features provided with this release, refer to the *GemFire Enterprise Release Notes* for version 5.7. Refer to the programming guides and programming API documentation included with the product for information on using GemFire Enterprise to develop applications. For detailed information on administering GemFire Enterprise systems, see the *GemFire Enterprise System Administrator's Guide*. For details on programming with GemFire Enterprise, see the *GemFire Enterprise Developer's Guide*.

GemFire Enterprise 5.7 includes these early access features:

Region Snapshots - Early Access

Snapshots are managed through the `saveSnapshot` and `loadSnapshot` methods of the `Region` interface. These are the key considerations for snapshots:

- Taking a snapshot does not stop region activity, so modifications to the region that are made while the snapshot is being taken may or may not be saved to disk.
- Snapshots are not recursive. The region's entries are saved, but subregions and subregion entries are not.
- Loading a snapshot removes all existing distributed region contents before loading the data from disk. All existing subregions and entries are destroyed before the snapshot is loaded. Remote regions with the same name are cleared of all subregions and entries, and remote regions that are configured as replicas do a new `getInitialImage` operation to get the data from this snapshot. All existing references to the distributed region become unusable and, if used, throw a `RegionReinitializedException`. All applications with references to the region must reacquire the region using `Cache.getRegion` or `Region.getSubregion`.

For additional information on snapshots, see the online Java documentation.

EntryEvent Get Serialized Values - Early Access

You can retrieve serialized values from `EntryEvent`. This is useful if you get values from one region's events just to put them into a separate cache region. The standard `getOldValue` and `getNewValue` functions deserialize the values, which would then be reserialized when you `put` them into the cache. There is no counterpart `put` function as the `put` recognizes that the value is serialized and bypasses the serialization step.

For details on the new API supporting this functionality, see the online Java documentation in `com.gemstone.gemfire.cache` for the interface `SerializedCacheValue` and for the `EntryEvent` methods `getSerializedNewValue` and `getSerializedOldValue`.

Bulk Operations Using getAll - Early Access

The `Region.getAll` method gets values for all keys in the input collection. The method returns a map of values for the input keys. See the `Region` interface description in the Java online API documentation for details about using `getAll`.

Load Balancing in Multi-site Installations - Early Access

GemFire Enterprise now allows you to balance the load of gateway communication between multiple VMs. The basic approach is to define multiple gateway hubs and then split outgoing region event messaging between them.

With the new functionality, you can:

- Specify multiple gateway-hubs in your cache
- Indicate for each VM whether a hub should start as primary or secondary
- Point each gateway-enabled region to a specific gateway-hub

Only the primary hub instances send region events to remote sites. By splitting your hub primaries between multiple VMs and splitting your region events between hubs, you can achieve a level of load balancing in your outgoing hub messaging.

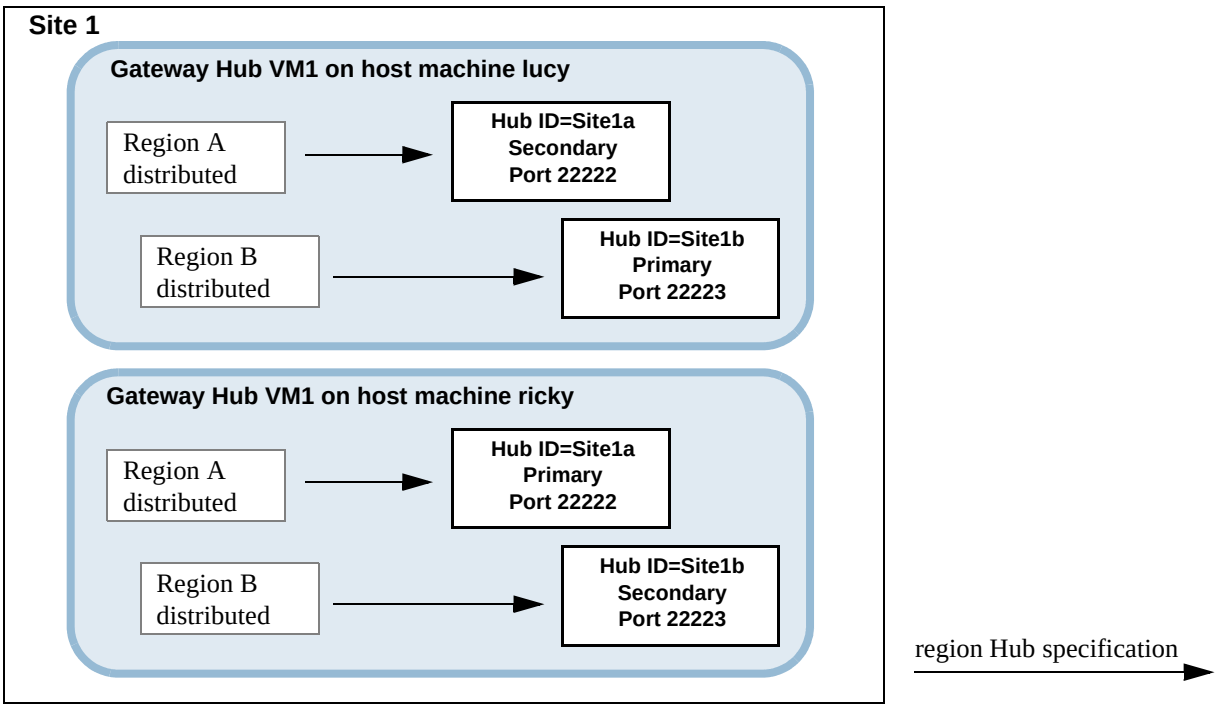
The Multiple-Hub Configuration

Figure 1.1 shows a typical multiple-hub configuration. There may be any number of VMs in this distributed system in addition to the ones that host the gateways hubs.

In this configuration:

- There are two VMs hosting gateway hubs.
- There are two gateway hubs in each hub VM. One hub is primary in VM1 and the other hub is primary in VM2. Only the primaries send data to remote sites.
- Each hub has some of the region traffic pointing to it.

Figure 1.1 Two-Hub Configuration With Primaries in Different VMs

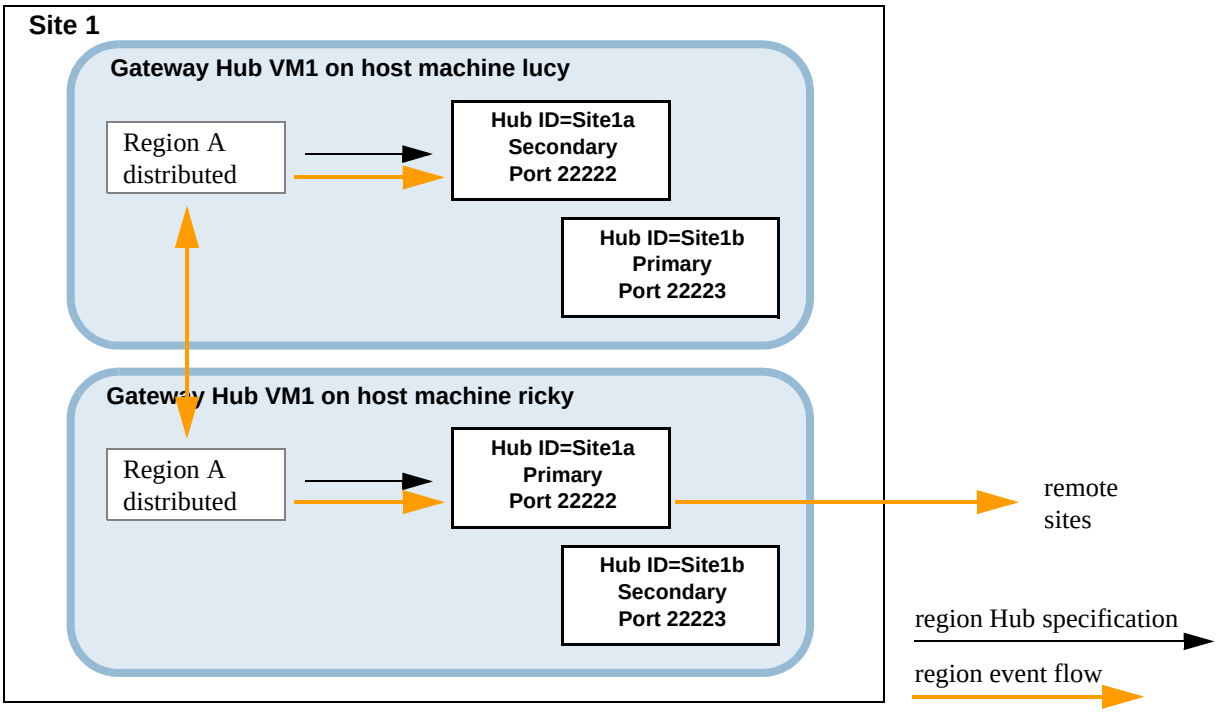


Event Flow With Multiple Hubs

Figure 1.2 shows event flow for region A in the multiple-hub configuration of Figure 1.1 on page 5.

- Region events originating in VM1 are stored in the local secondary Hub Site1a and are also distributed to VM2. Then in VM2, the events arriving in the local Region are sent to the VM's primary Hub Site1a, which then forwards them to remote sites.
- Region events originating in VM2 are sent directly out of the local primary Hub Site1a. They are also distributed to VM1 for cache update and storage in the secondary Hub Site1a.
- HubSite1b never sees this region's outgoing events.

Figure 1.2 Outgoing Region Event Flow for Two-Hub Configuration



If you have multiple hubs, make sure you explicitly assign one hub to each distributed region using the `hub-id` *region attribute. If you do not specify a hub for a region, its events will go to all hubs and each hub's primary will send the message to the remote sites, resulting in duplicate sends.*

Configuring the Multiple-Hub Site

These tables list the new configuration attributes for multi-site load balancing.

Table 1.1 Gateway Hub Attribute

gateway-hub attribute	Description	Default
`startup-policy`	Specifies whether the hub should attempt to start as primary or secondary hub. Valid values are `primary`, `secondary`, and `none`.	`none`

Table 1.2 Region Attribute

region attribute	Description	Default
`hub-id`	Specifies the destination hub for region events. *For multiple-hub systems this must be specified. If it is not, region events go to all available hubs, which results in duplicate sends to remote sites.*	`""`

Hub Configuration

The `cache.xml` in this example defines the gateway hub configurations for VM1 in Figure 1.1 on page 5.

Example 1.1 Two-Hub Gateway Hub Configuration in cache.xml

```
<cache>
   <gateway-hub id="Site1a" port="22222" startup-policy="secondary">
      ...
   </gateway-hub>
   <gateway-hub id="Site1b" port="22223" startup-policy="primary">
      ...
   </gateway-hub>
```

This shows the API equivalent.

Example 1.2 Two-Hub Gateway Hub Configuration Through the API

```
// Create or obtain the GemFire cache
Cache cache = ... ;

// Create the first Gateway Hub
GatewayHub site1aHub = cache.setGatewayHub("Site1a", 22222);
site1aHub.setStartupPolicy("STARTUP_POLICY_SECONDARY");

// Create the second Gateway Hub
GatewayHub site1bHub = cache.setGatewayHub("Site1b", 22223);
site1bHub.setStartupPolicy("STARTUP_POLICY_PRIMARY");

// Define gateways and endpoints for the gateway hubs ...
```

Region Configuration

When you use multiple hubs, you must point each gateway-enabled region at one of the hubs. If you do not specify a hub, region events are delivered to every hub and then sent to remote sites by each hub's primary instance. The region configuration must be consistent across the distributed system, even in caches where there are no hubs running.

Split region events between your hubs as evenly as you can. If you have two hubs and three regions, for example, with one region having the bulk of the events, put that region on one hub and the others on the other hub.

This `cache.xml` assigns the example regions in Figure 1.1 on page 5 to their respective hubs.

Example 1.3 Enabling Region Communication With Gateway Hubs in cache.xml

```
<region name="A">
   <region-attributes ... enable-gateway="true" hub-id="Site1a"/>
   ...
</region>
<region name="B">
   <region-attributes ... enable-gateway="true" hub-id="Site1b"/>
   ...
</region>
```

This is the same configuration through the API.

Example 1.4 Enabling Region Communication With a Gateway Hub Through the API

```
// Region A
AttributesFactory factory = new AttributesFactory();
factory.setEnableGateway(new Boolean(true));
factory.setGatewayHubId("Site1a");
Region multiSiteRegionA = cache.createRegion("A", factory.create());

// Region B
AttributesFactory factory = new AttributesFactory();
factory.setEnableGateway(new Boolean(true));
factory.setGatewayHubId("Site1b");
Region multiSiteRegionB = cache.createRegion("B", factory.create());
```

How Hub Startup Policy Affects Startup Behavior

Every hub must have one primary instance running to maintain communication with remote sites. The other instances are secondaries that act as backups to the primary.

You can specify which hub instances should start as primary and which as secondary in your distributed system. When a hub is initialized, it attempts to assume the role specified, but will start up even if it must assume a different role.

This table describes startup behavior for the three policies.

Table 1.3 Gateway Hub Startup Policy and Startup Behavior

Startup Policy	Startup Behavior
`none` (default)	If no primary is running for the id, the hub starts as primary. Otherwise it starts as secondary.
`primary`	Same behavior as for `none` with the addition that the hub logs a warning if it starts as secondary.
`secondary`	If a primary is present, the hub starts as secondary. If there is no primary, the hub waits for up to one minute for a primary to start. If no primary starts in that time, the hub starts as primary and logs a warning.

Configuring Remote Sites to Point to Your Multiple-Hub System

The regions' hub specifications only affect which hub processes their outgoing events. For incoming events, hubs process every event received regardless of their destination regions. For high availability, specify all of your hubs as gateway endpoints on your remote site. For any event, the remote site will send to only one endpoint.

This is a valid remote gateway specification pointing to our example site.

Example 1.5 Remote Site Configuration Pointing to the Multiple-Hub Site

```
<cache>
   <gateway-hub id="Some remote gateway hub ...
      <gateway id="Site1a" ...
         <gateway-endpoint id="Site1aLucy" host="lucy" port="22222"/>
         <gateway-endpoint id="Site1aRicky" host="ricky" port="22222"/>
         <gateway-endpoint id="Site1bLucy" host="lucy" port="22223"/>
         <gateway-endpoint id="Site1bRicky" host="ricky" port="22223"/>
      </gateway>
   </gateway-hub>
```

Basic Configuration Steps

To implement a multiple-hub system, first determine the configuration of the connections between the multiple distributed system sites. Then, for each site, follow these steps:

1. Identify the VMs that will host your gateway hubs, including which machines they will run on the the addresses they will use for gateway communication.
2. Decide the gateway hub startup policy for each VM's hubs.
3. Decide which regions will point to which hubs.
4. Following your planned configuration, for each hub define the gateways to the remote sites, including:
   - Endpoints - point to every hub instance on the remote site
   - Queue attributes
5. Enable regions to communicate updates to the gateways, specifying:
   - enable gateway
   - hub ID
6. Start all host VMs in the distributed system at the same time. This allows the primaries to assume their roles before any secondaries default to primary.

For details on the basic configuration of gateway communication, see the information on multi-site configuration in the *GemFire Enterprise Developer's Guide.*

Manual Start for Gateway Hubs - Early Access

You can now declare gateway hubs in your `cache.xml` without automatically starting them. Before this release, any gateway hub declared in your `cache.xml` would automatically start when the cache was created. This is still the default behavior. You can now disable the hub startup by setting the new `gateway-hub` attribute `manual-start` to `true` in the XML file. If manual start is enabled, you must then start the hub through the API using one of the `GatewayHub start` methods.

This option allows you to use the normal XML for hub configuration and to control when your system starts communicating with remote sites. This is useful for involved startup procedures, such as those required when recovering from a crash, or for any situation in which you want to ensure that cache initiation finishes before remote site operations are allowed in.

Write Behind Cache Listener Using Gateway Hubs- Early Access

You can now implement a write-behind cache listener in your gateway hubs. This is accomplished by creating multiple gateways in a single distributed system member and configuring one of them with no incoming port specification. The hub with no incoming port will only process events from inside the distributed system, allowing you to run the hub with write-only behavior. Without a port specified, the hub does not start a server socket to listen for incoming communication. It only receives events from inside its own distributed system, notifies its listeners, and sends the events to any configured remote sites.

To support the new multi-hub option, a new `Cache` API method `getGatewayHub(String id)` retrieves a gateway hub for a specific ID.