IceStorm Delivery Modes

Previous Next

The delivery mode for events sent to subscribers is controlled by the proxy that the subscriber passes to IceStorm. For example, if the subscriber subscribes with a oneway proxy, events will be forwarded by IceStorm as oneway messages.

On this page:

Subscribing with a Twoway Proxy

In this mode each event is sent to the subscriber as a separate twoway message. This allows the subscriber to enable server-side active connection management (ACM) without risking lost messages, because IceStorm will re-send an event if the subscriber happens to close its connection at the wrong moment.

If you combine a twoway proxy with a `reliability` QoS of `ordered`, messages will be forwarded to the subscriber in the order in which they are received. This is guaranteed because IceStorm will wait for a reply from the subscriber for each event before sending the next event.

Without ordered delivery, events may be delivered out-of-order to the subscriber because IceStorm will send an event as soon as possible (without waiting for a reply for the preceding event). If the subscriber uses a thread pool with more than one thread, this can result in out-of-order dispatch of messages in the subscriber.

For single-threaded subscribers and subscribers using a serialized thread pool, twoway delivery always produces in-order dispatch of events in the subscriber.

With twoway delivery, IceStorm is informed of any failure to deliver an event by the Ice run time. For example, IceStorm may not be able to establish a connection to a subscriber, or may receive an `ObjectNotExistException` when it forwards an event. Any failure to deliver an event to a subscriber (possibly after a transparent retry by the Ice run time) results in the cancellation of the corresponding subscription.

Back to Top ^

Subscribing with a Oneway Proxy

In this mode each event is sent to the subscriber as a oneway message. If more than one event is ready to be delivered, the events are sent in a single batch. This delivery mode is more efficient than using twoway delivery. However, the subscriber cannot use active connection management without the risk of events being lost. In addition, if something goes wrong with the subscriber, such as the subscriber having destroyed its callback object without unsubscribing, or having subscribed an object with the wrong interface, IceStorm does not notice the failure and will continue to send events to the non-existent subscriber object for as long as it can maintain a connection to the subscriber's endpoint.

For multi-threaded subscribers, oneway delivery can result in out-of-order delivery of events. For single-threaded subscribers and subscribers using a serialized thread pool, events are delivered in order.

Back to Top ^

Subscribing with a Batch Oneway Proxy

With this delivery mode, IceStorm buffers events from publishers and sends them in batches to the subscriber. This reduces network overhead and is more efficient than oneway delivery. However, as for oneway delivery, the subscriber cannot use active connection management without the risk of losing events. In addition, events can be delivered out of order if the subscriber is multi-threaded. Batch oneway delivery, while providing better throughput, increases latency because events arrive in "bursts". You can control the interval at which batched events are flushed by setting the `IceStorm.Flush.Timeout` property.

Back to Top ^

Subscribing with a Datagram Proxy

With this delivery mode, events are forwarded as UDP messages, optionally with multicast semantics. This means that events can be delivered out of order, can be lost, and can even be duplicated. In addition, IceStorm cannot detect anything about the delivery status of events. This means that if a subscriber disappears without unsubscribing, IceStorm will attempt to forward events to the subscriber indefinitely. If you use datagram delivery, you need to be careful that subscribers unsubscribe before they disappear; otherwise, stale subscriptions can accumulate in IceStorm over time, bogging down the service as it delivers more and more events to subscribers that no longer exist.

Back to Top ^

Subscribing with a Batch Datagram Proxy

With this delivery mode, events are forwarded as batches within a datagram. The same considerations as for datagram delivery and oneway batched delivery apply here. In addition, keep in mind that, due to the size limit for datagrams, batched datagram delivery makes sense only if events are small. (You should also consider enabling compression with this delivery mode.)

Back to Top ^

See Also

- Active Connection Management
- Oneway Invocations
- The Ice Threading Model
- Batched Invocations
- IceStorm Properties

Previous Next