Concurrent Proxy Invocations

Proxy objects are fully thread safe, meaning a client can invoke on the same proxy object from multiple threads concurrently without the need for additional synchronization. However, Ice makes few guarantees about the order in which it sends proxy invocations over a connection.

To understand the ordering issue, it's important to first understand some fundamental proxy concepts:

- At any point in time, a proxy may or may not be associated with a connection.
- A new proxy initially has no connection, and Ice does not attempt to associate it with a connection until its first invocation.
- If Ice needs to establish a new connection for a proxy, Ice queues all invocations on that proxy until the connection succeeds.
- After a proxy is associated with a connection, the proxy may or may not cache that connection for subsequent invocations.
- Proxies share connections by default, but an application can force proxies to use separate connections.

Ice guarantees that ordering will be maintained for invocations on the same proxy object, but only if that proxy caches its connection. This guarantee also holds for invocations on two or more proxies that programmatically compare as equal, meaning the proxies have the same identity and other configuration settings, and furthermore these proxies must cache connections as well.

Ice does not guarantee the order of invocations for any other situation.

The order in which the Ice run time in a client sends invocations over a connection does not necessarily determine the order in which they will be executed in the server. See Thread Pool Design Considerations for information about ordering guarantees in servers.

Back to Top ^

See Also

- Proxies
- Connection Establishment
- Thread Pool Design Considerations