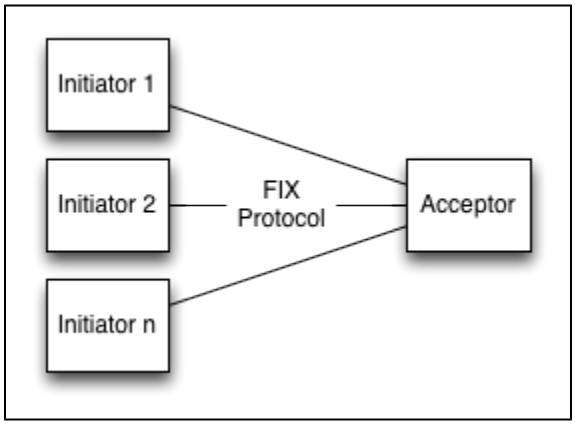
Overview

FIX Overview

The Financial Information eXchange (FIX) protocol defines a standard format for exchanging trade-related messages. Developed by a consortium of institutions to improve the interoperability and efficiency of automated trading systems, FIX defines the protocol but does not specify an implementation. FIX developers can select from a number of commercial and open-source products that implement the protocol.

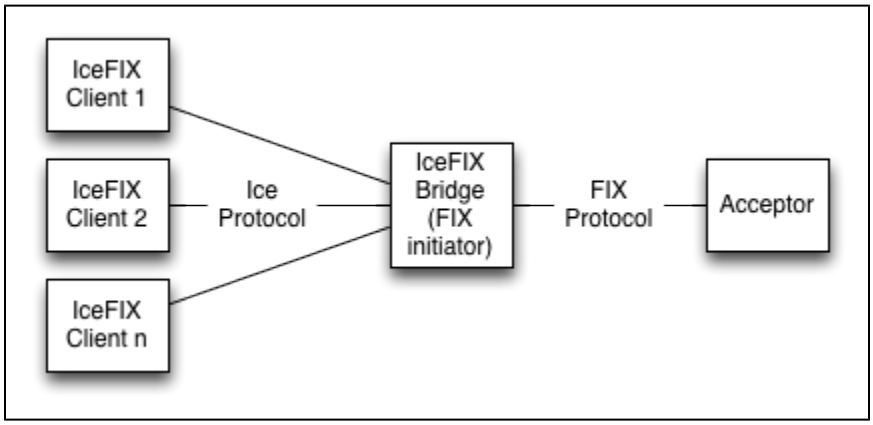
In FIX terminology, an *initiator* establishes a session with an *acceptor*. During this session, FIX protocol messages flow in both directions between the trading partners and correspond to trade events such as orders, confirmations, and cancellations. The figure below provides a simple illustration of several initiators communicating with the same acceptor:



The details of session establishment and message exchange are determined by the FIX implementation being used. Products typically include classes that encapsulate common activities such as creating and maintaining a session, encoding and sending messages, and receiving and decoding messages.

IceFIX Overview

IceFIX is a collection of tools that simplifies the development and deployment of FIX applications using Ice. The primary IceFIX component is a *bridge* that acts as both a FIX initiator and a reliable store-and-forward Ice message-delivery platform. Each instance of an IceFIX bridge establishes a session with a single remote acceptor. Any number of Ice clients can use that bridge to exchange messages with its acceptor, as shown below:



Developers gain several important advantages by using an IceFIX bridge as an intermediary:

- **Session management**
  Applications no longer need to be concerned with creating and maintaining FIX sessions, as the bridge now handles these tasks. The bridge defines a straightforward Slice API for client registration and message exchange.

- **Interoperability**
  IceFIX does not define Slice data structures corresponding to the FIX protocol message types but rather transfers FIX messages in encoded form. Developers are free to use any FIX implementation to encode and decode their messages.

- **Reliability**
  The bridge stores message traffic in a database to improve reliability. If the bridge receives a message from the acceptor for an inactive client, the bridge stores the message and forwards it to the client upon its next activation. Similarly, if a client sends a message via the bridge but the session is currently inactive, the bridge stores the message until the session can be restored.

- **Flexibility**
  The bridge is implemented as an IceBox service, which gives you a great deal of flexibility for configuring and deploying bridge instances. For example, you can deploy each bridge as a standalone server, or combine several independent bridge services into the same server. You can also use IceGrid to automate these tasks, and take advantage of all that IceGrid has to offer.

- **Integration**
  IceFIX allows FIX applications to be integrated into the Ice platform. Developers can choose from any of the programming languages and operating systems that Ice supports.

- **Administration**
  The bridge exposes an administrative API and includes a command-line utility for managing deployed bridge instances. If you use IceGrid to deploy your bridges, the IceGrid administrative tools will also simplify your management tasks.

IceFIX uses QuickFIX to implement the bridge. The code examples in this user guide all use QuickFIX for encoding and decoding messages.

IceFIX works with any standard FIX acceptor, such as the sample programs that accompany the QuickFIX distribution. To simplify the testing process, the IceFIX distribution also contains a sample FIX acceptor.

Previous Next