ClusteringAndFederation

Clustering And Federation

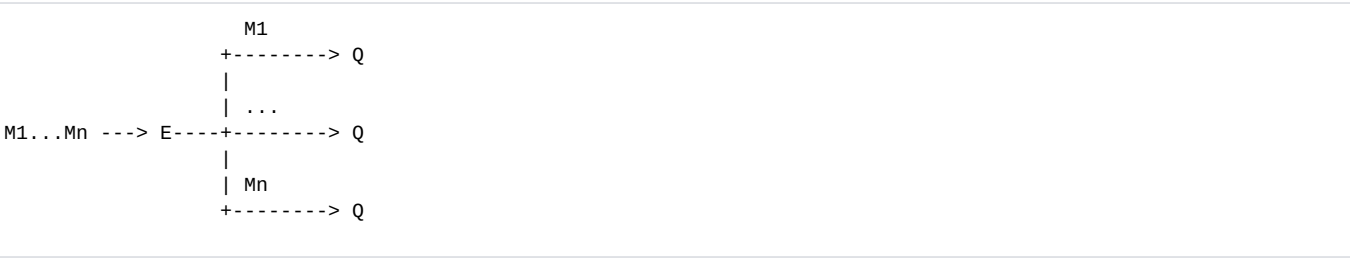
Each diagram below depicts a distributed network of exchanges and queues. The following notation is used in all diagrams:

- M: message
- E: exchange
- Q: queue

Multicast

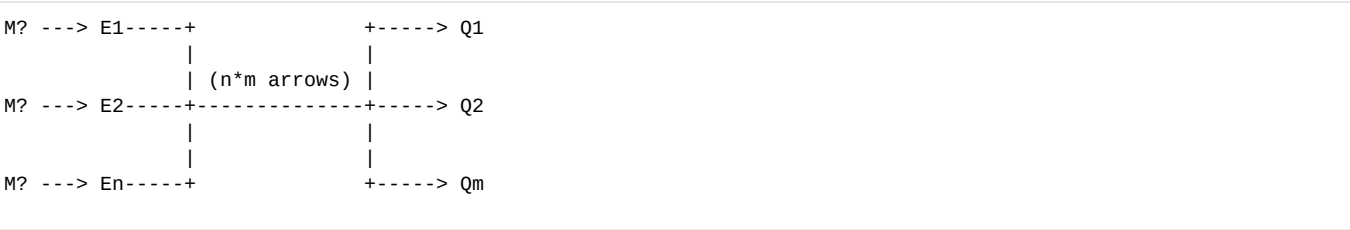
Queue contents are duplicated across all queues. For this scenario PGM would be ideal between E and Q, or even directly between E and consumers.

Load Balancing



No ordering is guaranteed accross different queues. A naive implementation could just be an exchange doing round-robin routing or any algorithm of choice. A more complicated exchange could have flow control between each queue and the exchange.

Multiple Exchanges



Both the Load Balancing and Multicast scenarios can be extended by adding multiple exchange nodes wired into the same (or an overlapping) set of queues. One virtual mega exchange (with relaxed ordering semantics) could be created by segmenting client connections between exchanges. This could be done using a number of strategies, e.g. round-robin dns, name mangling, redirects.

The topologies described above could in theory be use in a variety of scenarios ranging from an an isolated high speed subnet with identically configured nodes to a loosely coupled WAN with separately administered nodes. In fact a single network could include exchanges bound to local queues, remote queues available on an isolated high speed subnet, and remote destinations (exchange or queue) available over WAN/internet. In the last case the exchange may be requred to queue messages routed to the remote destination if the WAN/internet link is down.

In the terminology I've been using, a cluster is a set of machines sharing the same software and configuration, and generally connected via an isolated high speed subnet. A federation on the other hand consists of distinctly configured machines individually wired together. Both clustering and federation **could** share a common protocol for message delivery. This could possibly even be used for multicast if it were a simple stateless store-and-forward protocol. (Note the "store" in "store-and-forward" can mean both store on disk and store in memory.)

With this model the key distinction between a cluster and a federation is that all the nodes in a cluster are managed as a single unit, e.g. one place to start/stop/add/remove/etc. Because of this the nodes in a cluster have to pass control messages to each other distinct from the general message traffic. These control messages need to be isolated from the general message traffic (e.g. on their own subnet). This could be done using JGroups and OpenAIS for Java and C++ respectively.

This document doesn't directly address fault tolerance, but it is assumed that any node/broker that contains state can be configured to have a passive counterpart that supports two methodologies for failover. Broker swapout based on virtual IP, or client reconnect to a backup IP.