Design Pattern: One to Many

TYPICAL APPLICATIONS

Radar Track Distribution

Financial Services Market Data Distribution

Enterprise Data Access

Overview

A common use case in distributed applications is a few devices distributing data to many consuming applications. Typically in this architecture, each client application is acting on a portion of the data. In distributed publish/subscribe architectures, this is known as a One-to-Many distribution pattern.

Design Considerations

The primary design consideration for this pattern is to minimize the network traffic regardless of how many applications are subscribing to the data. This functionality is typically handled by the use of multicast addressing within the network topology. Unicast addressing can also be used in this design pattern, however, this will result in a separate message for each individual subscribing application and an increase in overall network bandwidth usage. Through the use of multicast, a single network packet sent by a publisher can be received by multiple subscribers. These transfers can be set up on a single multicast address or, depending on the load of the system, can also be setup on multiple Multicast address groups. Once the data is sent, each subscribing application is required to filter out the data that is not relevant. Figure 1 shows there are two groups of subscribing applications, each application group is using its own multicast group, and therefore each sending of data will result in two packets of data going out on the network, once for each multicast group. Multicast delivery of data can dramatically decrease the effective network bandwidth usage on a system where one application is sending data to many other applications. The performance graph shown here also shows there is minimal degradation on overall throughput capability when using multicast to many subscribers using reliable QoS settings.

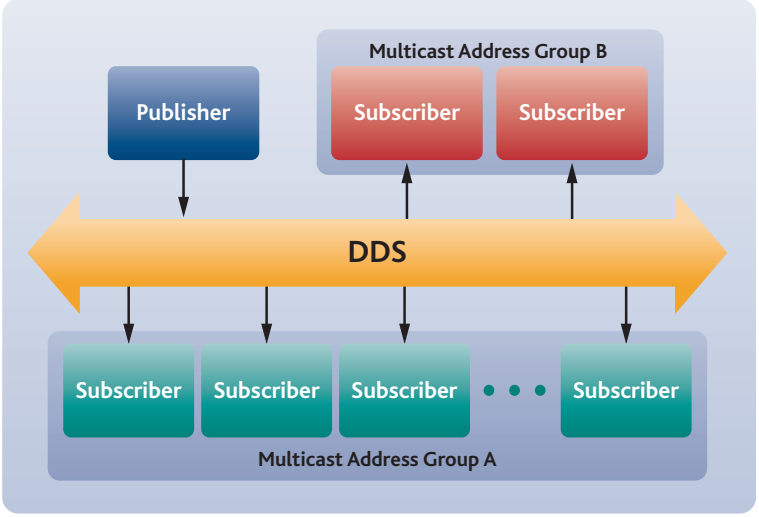
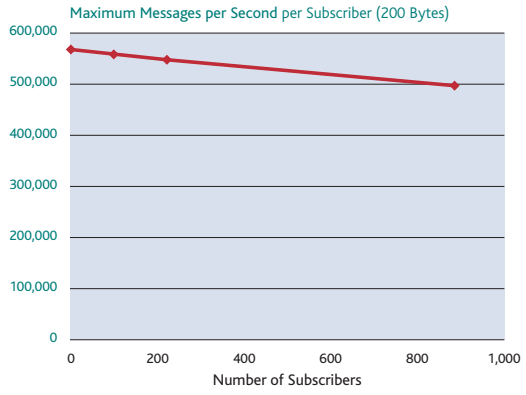


Figure 1.

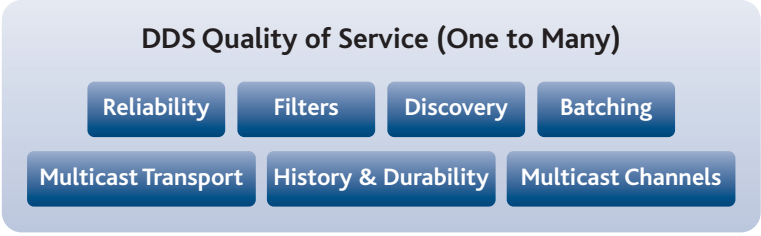


Applicable DDS Quality of Service Parameters

Reliability (DataReader, DataWriter)

Even with multicast in use to minimize traffic, reliability is still implemented on a per reader basis. Keep in mind, however, that for each reliable reader, there will still be additional reliability network traffic as heartbeats and acknowledgments (& Nacks) are all done on individual

unicast addressing. Disabling positive acknowledgements can also reduce reliability overhead for a system with high fan-out.

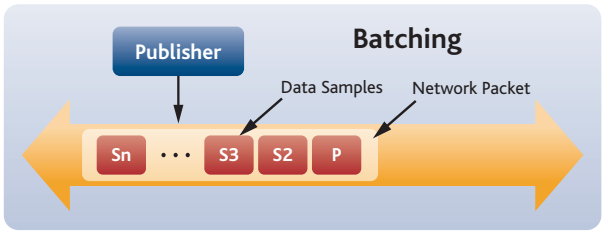


History & Durability (DataWriter)

This Quality of Service enables your application to specify how much data should be kept around for late joining applications. This can be set to a specific depth or to "Keep All". In this case where there are many consumers of data, writer side history with Durability allows each reader to start up any time and be able to access previously sent data, and eliminates the need to enforce a startup order of publishers/subscribers.

Batching (DataWriter)

When your data samples are small in size (< 400 bytes) and your publishing rate is high, it is not very efficient to send each one individually. By batching together several small packets into a single network packet enables higher throughput and less overhead. Combining this with multicast can achieve very high scale data publication rates. This can be very helpful in high fan-out cases to further reduce bandwidth on the network.



Content Filtering (DataReader)

With many subscribers, each one may only be interested in a small subset of the data. Using Content Filters, each subscribing application can further filter which data it wants to receive by specifying an SQL-like filter clause. This filter enables an application to only receive data when its value is above a particular threshold or maybe in a specified range of values. Filters could also be setup to only look for error conditions or alarm conditions. Content filters can be applied to any of the fields in the topic and are dynamically changeable so that applications can easily change their region of interest. When using multicast, content filters will be applied at the data reader side.

MultiChannel (DataWriter)

While multicast is very useful, the employment of a single multicast address means each application has to do more filtering of data than may be necessary. If the data set is large enough, you can break up your data set on to different multicast channels. Then when your subscribing application only require data on a specific channel, additional data filtering is not incurred on the subscribing application. Also, through the use of the multiple multicast addresses, standard hardware routing boxes, using IGMP snooping, can segment your traffic appropriately.

Transport Multicast (Data Reader)

Set the multicast address to be used. As shown above, setting a multicast receive address can greatly reduce bandwidth usage when a large number of subscribers are receiving data.

Discovery (DataReader)

Often, consumers do not need to discover one another, only the producer. To achieve this behavior—while improving scalability and reducing resource use accordingly—change the Discovery QoS policy settings on the readers to only perform discovery with the writer's unicast address.

U.S. HEADQUARTERS
Real-Time Innovations, Inc.
385 Moffett Park Drive
Sunnyvale, CA 94089
Tel: (408) 990-7400
Fax: (408) 990-7402
info@rti.com