OMG Data-Distribution Service (DDS): Architectural Overview

Gerardo Pardo-Castellote
Real-Time Innovations, Inc. (RTI)
Phone: 1-408-734-4200, x106
Email: pardo@rti.com

Topic Areas	Software Architectures, Reusability, Scalability, and Standards
	Middleware Libraries and Application Programming Interfaces
	Fault-Tolerant Hardware and Software Techniques

Summary

The OMG Data-Distribution Service (DDS) is a new specification for publish-subscribe data-distribution systems. The purpose of the specification is to provide a common application-level interface that clearly defines the data-distribution service. The specification describes the service using UML, providing a platform-independent model that can then be mapped into a variety of concrete platforms and programming languages.

This paper introduces the OMG DDS specification, describes the main aspects of the model, compares it with related technologies, and gives examples of the communication scenarios it supports.

This paper and presentation will clearly explain the important differences between **data-centric publish-subscribe** and **object-centric client-server** (e.g. CORBA) communications, along with the applicability of each for real-time systems.

The OMG DDS attempts to unify the common practice of several existing implementations enumerating and providing formal definitions for the Quality of Service (QoS) settings that can be used to configure the service.

Publish-subscribe networking is a key component of the Navy Open Systems Architecture (**Navy OA**) initiative. This talk will also highlight practical publish-subscribe implementations in Navy systems such as LPD 17, SSDS, and DD(X).

Background

The goal of the DDS specification is to facilitate the efficient distribution of data in a distributed system. Participants using DDS can "read" and "write" data efficiently and naturally with a typed interface. Underneath, the DDS middleware will distribute the data so that each reading participant can access the "most-current" values. In effect, the service creates a global "data space" that any participant can read and write. It also creates a name space to allow participants to find and share objects.

DDS targets real-time systems; the API and QoS are chosen to balance predictable behavior and implementation efficiency/performance. We will note some of these tradeoffs in this paper.

Data-Centric versus Object-Centric communications models

Central to understanding the need for this new standard is an examination of the fundamental architectural differences between a "data-centric" and "object-centric" view of information communicated in a distributed real-time system.

DDS provides a natural counterpoint to the existing well-known CORBA model in which method invocations on remote objects are accessed through an interface defined in the Interface Descriptor Language (IDL). With CORBA, data is communicated indirectly through arguments in the method invocations or through their return values.

However, in many real-time applications the communications pattern is often modeled as pure data-centric exchange where applications publish supply or stream) "data" which is then available to the remote applications that are interested in it. Of primary concern is the efficient distribution of data with minimal overhead and the need to scale to hundreds or thousands of subscribers in a robust, fault-tolerant manner. These types of applications can be found in C4I systems, distributed control and simulation, telecom equipment control, and network management.

Comparison to Distributed Shared Memory

Additional requirements of many real-time applications include the need to control QoS properties that affect the predictability, overhead, and resources used. Distributed shared memory is a classic model that provides data-centric exchanges. However, this model is particularly difficult and "unnatural" to implement efficiently over the Internet.

Therefore, another model, the Data-Centric Publish-Subscribe (DCPS) model, has become popular in many real-time applications. While there are several commercial and in-house developments providing this type of facility, to date, there have been no general-purpose data-distribution standards. As a result, no common models directly support a data-centric system for information exchange.

The OMG Data-Distribution Service (DDS) is an attempt to solve this situation. The specification also defines the operations and QoS attributes each of these objects supports and the interfaces an application can use to be notified of changes to the data or wait for specific changes to occur.

Comparison to existing OMG Notification Service

This paper will examine the fact that, while it is theoretically possible for an application developer to use the OMG Notification Service to propagate the changes to data structures to provide the functionality of the DDS, doing this would be significantly complex because the

Notification Service does not have a concept of data objects or data-object instances nor does it have a concept of state coherence.

Comparison to existing High-Level Architecture (HLA) Run-Time Infrastructure (RTI)

HLA, also known as the OMG Distributed Simulation Facility, is a standard from both IEEE and OMG. It describes a data-centric publish-subscribe facility and a data model. The OMG specification is an IDL-only specification and can be mapped on top of multiple transports. The specification address some of the requirements of data-centric publish subscribe: the application uses a publish-subscribe interface to interact with the middleware, and it includes a data model and supports content-based subscriptions.

However, the HLA data model supports a specialization hierarchy, but not an aggregation hierarchy. The set of types defined cannot evolve over time. Moreover, the data elements themselves are un-typed and un-marshaled (they are plain sequences of octets). HLA also offers no generic QoS facilities.

Applications

This paper will describe the successful implementation of data-centric publish-subscribe communications in distributed modeling and simulation (M&S) as well as deployed Navy systems (pending release permissions). The presentation can include examples (depending on audience interest and familiarity) such as:

Ship:	Raytheon/Lockheed Martin LPD-17 Program
Ground:	CLIP/LINK tactical communications Program
Air:	F-35 JSF EW Subsystem
Space:	NASA Robonaut Program