OMG: Data Distribution Service (DDS)

return to the OMG Standards

Table 1: Data sheet for Data Distribution Service (DDS)

Title	Data Distribution Service
Acronym	DDS
Version	1.4
OMG Document Number	formal/15-04-10
Release Date	March 2015
About Specification	https://www.omg.org/spec/DDS/
Document	https://www.omg.org/spec/DDS/1.4/PDF

Note: The following is an excerpt from the actual document. It is provided here as a convenience and is not authoritative. Refer to the original document as the authoritative reference.

Overview

The DDS specification describes a Data-Centric Publish-Subscribe (DCPS) model for distributed application communication and integration. This specification defines both the Application Interfaces (APIs) and the Communication Semantics (behavior and quality of service) that enable the efficient delivery of information from information producers to matching consumers.

- *The purpose of the DDS specification can be summarized as enabling the "Efficient and Robust Delivery of the Right Information to the Right Place at the Right Time."*

The expected application domains require DCPS to be high-performance and predictable as well as efficient in its use of resources. To meet these requirements it is important that the interfaces are designed in such a way that they:

- *Allow the middleware to pre-allocate resources so that dynamic resource allocation can be reduced to the minimum,*
- *Avoid properties that may require the use of unbounded or hard-to-predict resources, and*
- *Minimize the need to make copies of the data.*

DDS uses typed interfaces (i.e., interfaces that take into account the actual data types) to the extent possible. Typed interfaces offer the following advantages:

- *They are simpler to use: the programmer directly manipulates constructs that naturally represent the data.*
- *They are safer to use: verifications can be performed at compile time.*
- *They can be more efficient: the execution code can rely on the knowledge of the exact data type it has in advance, to e.g., pre-allocate resources.*

It should be noted that the decision to use typed interfaces implies the need for a generation tool to translate type descriptions into appropriate interfaces and implementations that fill the gap between the typed interfaces and the generic middleware.

QoS (Quality of Service) is a general concept that is used to specify the behavior of a service. Programming service behavior by means of QoS settings offers the advantage that the application developer only indicates 'what' is wanted rather than 'how' this QoS should be achieved. Generally speaking, QoS is comprised of several QoS policies. Each QoS policy is then an independent description that associates a name with a value. Describing QoS by means of a list of independent QoS policies gives rise to more flexibility.

This specification is designed to allow a clear separation between the publish and the subscribe sides, so that an application process that only participates as a publisher can embed just what strictly relates to publication. Similarly, an application process that participates only as a subscriber can embed only what strictly relates to subscription.