8 Platform-Specific Models

8.1 DDS Data Model PSM

The DDS Data Model PSM defines a set of IDL files for the Data Model packages defined by the PIM.

Comments are added to the IDL files to reflect the mapping rules below.

The detailed rules for the MDA code generation from the Data Model PIM to the DDS PSM IDL are as follows:

• PIM attributes and compositions are mapped to IDL attributes;

• Optional PIM attributes are mapped to a union type with a single member present when the exists case attribute is true;

• Collections in the PIM are mapped to IDL sequences; a Length tag determines the sequence bounds;

• Specialization / Generalization PIM relationships are mapped to IDL unions. Additional data classes are introduced for non-abstract generalization classes that have attributes

8.2 DDS Services PSM

The DDS Services PSM defines IDL files for each package defined in the Services PIM. For each method on each interface class an IDL struct for a DDS topic named for the method is generated; each parameter is mapped to an attribute of the IDL struct. Note that the PIM only defines parameters with an 'in' mode, there are no 'return' parameters defined and all methods have at least one parameter. Comments are generated to match the PIM notes and to include the version number of this standard in each file.

Additionally the struct contains a subsystem_id key attribute of type subsystem_id_type. This indicates which subsystem published the data or is intended to read it as a subscriber.

Operations that require a response contain a request_id in the PIM that logically links request and response instances. In the DDS PSM, each request_id operation parameter is mapped to a keyed attribute of the DDS topic so that distinct request and response pairs can be retrieved from the DDS data space.

To robustly and efficiently ensure that the data exchanged between a particular subsystem and a CMS is

recognized correctly, topic samples pertaining to a particular subsystem are published on the partition

corresponding to the name used in the Subsystem Identification use case. Also, the CMS uses the

receive_cms_identification_data topic to allocate a subsystem_id to a subsystem; the subsystem sets the

subsystem_id to zero for the receive_subsystem_identification_data topic, for which the CMS subscribes

on the wildcard partition "*". Subsequently, for data intended for all subsystems, the CMS publishes

samples on partition "*" with a subsystem id of zero.

However, the Register Interest use case is mapped to the DDS DCPS Reader Listener interface and the

Provide Subsystem Services use case is mapped to the DDS DCPS Data Reader and Data Writer

interfaces, so there are no IDL files for these use cases.

8.3 GraphQL Data Model and Services PSM

The GraphQL PSM defines a set of schema definition language files, one for each Service interface

defined by the PIM; each of these files represents a self-contained service and contains definition for the

types represented. Comments are added to these files to reflect the mapping rules below.

The detailed rules for the MDA code generation from the Data Model PIM to the GraphQL are as follows:

• Enumerations are mapped to GraphQL enums;

• PIM Classes with an 'idlStruct' stereotype are mapped to both a GraphQL object and input type;

• Scalar idlTypedef stereotyped classes are inlined to primitive GraphQL types in the types that use

them;

• PIM attributes and compositions are mapped to GraphQL object and input attributes;

• Non-optional PIM attributes are mandatory GraphQL attributes;

• Collections in the PIM are mapped to GraphQL lists (which are unbounded);

• Specialization / Generalization PIM relationships are mapped to GraphQL union and interface object

types and an input type with optional attributes (and the same semantics). Additional data classes are

introduced for non-abstract generalization classes that have attributes

The GraphQL services derived from CMS interfaces allow the CMS to query and subscribe to operations

invoked by a Subsystem, whilst Subsystem can invoke the interface by making mutations. Services

derived from Subsystem interfaces allow the Subsystem to query and subscribe and the CMS to mutate.

Each GraphQL service contains:

• A schema object declaring query, mutation and subscription attributes;

• Query (also used for subscription) with an argument list allowing filtering by subsystem and whether

simulated

• Mutation object types each returning lists of operations;

• A union type with choices for each operation on the interface;

• A options input type with optional attributes for each operation on the interface for mutations;

• An object type for each operation including a argument list containing each key in the operation types

(as well as request_id if present) and an additional list of subsystem names returning a list of

operations;

• An input type for each operation including an additional list of subsystem names;

• Sensor Assessment, Supplementary Measurement and Track Reporting operations also support

additional arguments to filter by environment and area