UML Profile for DDS

a tutorial for OMG’s Workshop on Real-Time, Embedded and Enterprise-Scale Time-Critical Systems
May 24, 2010

Sam Mancarella
CTO - Sparx Systems
sam.mancarella@sparxsystems.com
Overview

• Part 1 - Introduction
  – DDS Design Challenges
  – Language Architecture & Overview

• Part 2 – Worked Example
  – DCPS
  – DLRL
  – Application Targets
  – MDA, PIM → PSM

• Part 3 - Conclusion
  – Other Applications
  – Concluding Remarks
  – Discussion
Introduction

• A UML Profile designed for the analysis and design of object-oriented systems using Data Distribution Service technology.

• Provides DDS designers, architects and practitioners with a standard, domain-specific modeling language to design DDS-based distributed information systems in a manner not specific to the underlying implementation of that design.
Introduction

• Beta Specification: mars/2008-06-18
• Joint Submission by:
  – PrismTech
  – Real Time Innovations Inc
  – Sparx Systems
• Request For Proposal: mars/2006-09-40
Overcoming the Challenges of DDS Design

- DDS Middleware Specification
  - Provides loosely-coupled data-centric publish/subscribe communications in real-time systems
  - Information producers (publishers)
  - Information consumers (subscribers)

- Quality of Service (QoS)
  - Diverse Quality of Service (QOS) requirements
  - Publishers, Subscribers, Readers, Writers, Topics, Participants
  - Balance predictable real-time behavior with implementation efficiency and performance

- Loose coupling

- Complex data structures

- Publishers
  - Publisher 1
  - Publisher 2
  - Publisher 3

- Topics
  - Topic i
  - Topic ii
  - Topic iii

- Subscribers
  - Subscriber 1
  - Subscriber 2
  - Subscriber 3
Overcoming the Challenges of DDS Design

• DDS is a PIM
  – Provides a platform independent model of entities, roles and QoS Policies
  – PIM is mapped to specific implementations, or platform specific models (PSM)
    • Variety of software languages
    • Variety of runtime platforms
    • Variety of vendors
Overcoming the Challenges of DDS Design

- **Manage Complexity**
  - Complex information models with QoS data

- **Heterogeneous Design**
  - Different implementations, same information model

- **Reuse**
  - Repository, Patterns

- **Change Management**
  - One change in model → 00’s changes in code
Model - Driven Architecture

• Domain-Specific Modeling
  – Taxonomy of constructs, relationships, constraints
  – Notation, presentation, diagrams
  – MOF or UML mappings (UML Profiles)

• PIM → PSM Transformation
  – Platform Independent Model transformed to Platform specific model automatically
  – One domain-specific model to another
Timeline

- RFP Issued September, 2006
- First LOI November, 2006
- First Initial Submission March, 2007
- First Revised Submission September, 2007
- Second LOI October, 2007
- Second Initial Sub December, 2007
- Second Revised Sub February, 2008
- BoD Adoption June, 2008
- FTF Charter June, 2008
- FTF Charter 2 July, 2009
- FTF Report Due June 2010
Vendor Support

- Sparx Systems – MDG Technology for DDS
  - Language Addin for Enterprise Architect
  - DDS-specific Toolboxes, Constructs, Diagrams
  - Automatically generates PSM code for OpenSplice & RTI DDS
    - Other DDS platform targets coming soon!
Language Architecture

• Part 1 - UML Profile
• Defines a collection of constructs that represent:
  – Data Centric Publish Subscribe Entities (+ QoS)
  – Data Local Reconstruction Layer

• Defined a collection of common constructs to define:
  – PSM Application Targets
  – Topic Data Types (IDL)
Language Architecture

- Part 2 – Metamodel
- Defines meta-level artifacts for XMI serialization
Worked Example - NetChat

Stage 1 – DCPS-only Application
Stage 2 – DLRL-Enabled
NetChat Overview

- Hypothetical, peer-to-peer network chat application
- Two Components:
  - “ChatRoom” DDS Dataspase containing conversation threads amongst users
  - “Directory Server” Application to maintain a collection of active NetChat users
- Real-world application of DDS DCPS and DLRL in distributed application designs
DCPS – QoS Policy Library

- qosPolicyLibrary Package
  - Top-Level Classifiers defining ‘default’ QoS Policies
  - Define sets of qosPolicyLibraries for domain-specific applications
  - Template of reusable QoS assets for multiple projects
DCPS – QoS Policy Library
DCPS – QoS Policy Library

Defines QoS policy data as tagged values
DCPS Topics & Data Types

- Data Types
  - Describes the data payloads for DCPS topics
  - IDL-based library
  - structs, unions, arrays
DCPS Topics & Data Types

- **Data Types**
  - Describes the data payloads for DCPS topics
  - IDL-based library
  - structs, unions, arrays

Attributes can be nominated as DCPS Key fields
DCPS Topics & Data Types

- DDS Topics
  - Describes the DCPS characteristics of the published/subscribe data type, constrained to QoS Policy
DCPS Topics & Data Types

- DDS Topics
  - Describes the DCPS characteristics of the published/subscribe data type, constrained to QoS Policy

<table>
<thead>
<tr>
<th>tags</th>
<th>expression = kind = STANDARD type = MessageType</th>
</tr>
</thead>
<tbody>
<tr>
<td>qos</td>
<td>deadline destination_order durability history</td>
</tr>
<tr>
<td></td>
<td>latency_budget lifespan liveliness ownership</td>
</tr>
<tr>
<td></td>
<td>reliability resource_limits topic_data transport_priority</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>key</th>
<th>Index: long</th>
</tr>
</thead>
<tbody>
<tr>
<td>idField</td>
<td>message: string</td>
</tr>
<tr>
<td>foreignKey</td>
<td>name: string</td>
</tr>
<tr>
<td>userId</td>
<td>user: long</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>tags</th>
<th>expression = kind = STANDARD type = UserType</th>
</tr>
</thead>
<tbody>
<tr>
<td>qos</td>
<td>deadline destination_order durability history</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>type</th>
</tr>
</thead>
<tbody>
<tr>
<td>idStruct » Message Type</td>
</tr>
</tbody>
</table>
DCPS Topics & Data Types

- DDS Topics
  - Describes the DCPS characteristics of the published/subscribe data type, constrained to QoS Policy

ContentFilteredTopic, MultiTopic denoted by kind, expression tags
• Domain Entity
  – Logical ‘grouping’ of DCPS Topics, & DomainParticipants
DCPS Domain & Entities

- DomainParticipant Entity
  - DDS Publish/Subscribe entity
DCPS Domain & Entities

- DomainParticipant Entity
  - Participate in domain nominated by tagged value
DCPS Domain & Entities

- DomainParticipant Entity
  - Qos applied as properties, typed by the QoS Policy types in the qosPolicyLibrary
DCPS Domain & Entities

- Added Publisher, Subscriber Entities
DCPS Domain & Entities

- Added DataReaders, DataWriters Entities
DDS Topics connected to DataReaders & DataWriters
Application Targets

- **ddsAppTarget**
  - Binds one or more DomainParticipants to a PSM configuration
Application Targets

• ddsAppTarget
  – Binds one or more DomainParticipants to a PSM configuration

‘usage’ Dependency binds the DomainParticipant to the Target
Application Targets

- **ddsAppTarget**
  - Binds one or more DomainParticipants to a PSM configuration

Tagged values specify the desired PSM output
Tool Specific: Enterprise Architect prompts the user to designate the application targets to a specific DDS output platform.
```java
private int CreateParticipant()
{
    // Create the participant
    participant = m_factory.create_participant(m_domainId, m_participantIndex, "ChatRoom", null,
    if (participant == null) {
        LogError("create_participant error");
        return -1;
    }
    return 0;
}

private int CreatePublisher()
{
    if (participant != null) {
        // Create the publisher
        publisher = m_factory.create_publisher(participant, "pub", null, StatusKind.STATUS_MASK_NONE);
        if (publisher == null) {
            LogError("create_publisher error");
            return -1;
        }
    }
    return 0;
}

private int CreateSubscriber()
{
    if (participant != null) {
        // Create the subscriber
        subscriber = m_factory.create_subscriber(participant, "sub", null, StatusKind.STATUS_MASK_NONE);
        if (subscriber == null) {
            LogError("create_subscriber error");
            return -1;
        }
    }
    return 0;
}

private int CreateTopics()
{
    if (participant != null) {
        // Register types before creating topic
        topicName = UserTypeTypeImport.setTypeName("/1");
    }
```
Worked Example

DLRL
DLRL – Class & Type Mapping

• dlrlClass
  – DLRL Class representing a subscribed DCPS Topic Type
DLRL – Class & Type Mapping

- `drlClass`
  - DLRL Class representing a subscribed DCPS Topic Type
DLRL – Class & Type Mapping

- dlrlAttribute
  - DLRL Attribute representing mapped DCPS Type fields
DLRL – Class & Type Mapping

- relation
  - Association used to aggregate multiple classes using DLRL foreign keys
DLRL – Local Reconstruction

- Cache
  - Describes a DLRL cache entity used to provide dlrl class access to the user
DLRL – Local Reconstruction

- Cache
  - Describes a DLRL cache entity used to provide dlrl class access to the user
DLRL – Local Reconstruction

- `objectHome, topicManager`
  - Binds the cache to DataReaders to access the specific DCPS Topic, Types
Application Targets

- ddsAppTarget
  - Binds one or more DomainParticipants to a PSM configuration
  - Binds at most one DLRL cache to the PSM configuration
Conclusion & Wrap Up
Other Applications

- Not just a DDS architecture description
- Not just a PIM
Other Applications

- Model Driven Architecture
  - UML based DSL
  - Interoperable with SysML, SOAML, UPDM, MARTE, etc…
  - Integrated, traceable ‘architectable’
  - Part of the ‘big picture’
Other Applications

• XMI Serialization → Direct Deployment
  – XMI Document describes the DDS application configuration with Participants, Topics, QoS, etc
  – Configuration loaded by runtime to configure nodes
  – No source code
Other Applications

- Visual Deployment Interface
  - DDS discovery to create a DDS model which visualizes a running deployment
  - Field Engineers interact with the DDS model to make changes to the deployment
  - Maintenance, re-engineering, documentation applications
Concluding Remarks

• UML Profile for DDS exemplifies the co-operation of multiple OMG standards to:
  – Overcome the real-world challenges of design complexity management
  – Provide turnkey rapid-development solutions for DDS applications

• Culmination of OMG’s
  – Real-time distributed data middleware technology
  – UML extensibility (domain-specific languages)
  – Model-Driven Development / Architecture
  – XML Metadata Interchange specifications
Concluding Remarks

• Next Steps
  – Complete the FTF submission
  – Final Publication & market adoption
  – RTF

• For More information
  – Contact us
    • sam.mancarella@sparxsystems.com
    • http://sparxsystems.com/dds
  – Visit the Sparx exhibit for more information & demo
Thank you for your attention!