CORBA /DDS, COMPETING or COMPLEMENTING THE TECHNOLOGIES?
Motivation

- Can I use CORBA to distribute data on the network?
- Is the notification service able to provide similar functionalities as DDS?
- Can I use DDS as a Request Broker?
- Can I use CORBA with DDS?
- Can DDS interwork with CORBA effectively?
- When Shall I use CORBA and when shall I use DDS?
DDS Genesis
Data Centric Canonical Models

- Data writers and Readers accessing to a common Global Data Space (GDS)
  - GDS can be:
    - Shared Memory,
    - Centric or Replicated Databases,
    - Distributed Databases.
  - Writers and Readers can share a unique view of data or they can have their own dataview.
- Having a unique GDS Holder for the whole system leads to:
  - Single point of failure,
  - Decrease scalability and performance,
- Distributed DBMSs are heavy and inappropriate for reactive and near realtime systems:
  - Don’t offer standard behaviors to synchronize and notify readers on data availability etc ...
  - Don’t offer enough data lifecycle control,
  - Involves heavy mechanisms to guaranty high level data consistency (distributed transactions, distributed locking protocols etc ...).
Virtual GDS should be managed and maintained by a Data Centric Middleware which is able to split and publish the right dataview, to the right reader at right time.

- Data Centric Pub-Sub
  - Virtual Global Data Space is split in a multitude of data views or topics
  - DCPS Purpose: QoS Driven Distributed Data management

- The Data Centric Middleware must be able to reconstruct the data using the Reader’s own native information model
  - Object Oriented model,
  - Relational Model, etc …
  - Data Local Reconstruction Layer
    - DLRL Purpose: provide an OO-model to access data as Local Objects

- DDS : DCPS + DLRL
The Data Distribution Paradigm

- DDS is used for Information centric rather Service centric applications
  - Focusing on **data** instead of Processing and Services
  - Model your information exchange using **high level data Models** instead of exchanging elementary data units (Messages, Events…)
  - **Delegate** Information dissemination, processing and management to the Middleware
    - Specify your **own view on data schema** and let the middleware build it for you!

---

**Publisher**
- Send
- Publish

**Subscriber**
- Receive
- Access data

**Data layer**
- Event/Messaging Oriented Middleware

**Publisher Complex (server)**
- Process & Update

**Subscriber Complex (server)**
- Process & Update

---

**Service Oriented Middleware**
- Information Centric Oriented Middleware
DDS and CORBA, congruous or incongruous?
As an MDA compliant OMG Spec, DDS is CORBA independent but not CORBA-incompatible,
- DDS PSM is in CORBA IDL
- DCPS data model can be represented with IDL data types and structures
  - Struct, Object by Values etc …
- CORBA and DDS have more than Common Data Representation and IDL in common
CORBA and DDS Underlying Concepts
Similarities and Differences

- DDS Concepts
  - Handle, manage and dispatch complex data models.
    - Portability, language and platform transparency
    - Platform independent Data-model
    - Strongly typed interfaces,
    - Realtime,
    - QoS Management,
    - Location Transparency
    - Data Integrity Control and Filtering
    - Domain managements
    - Data History
    - Fault tolerance (implementation dependent)

- Equivalent CORBA Concepts
  - Handle and manage most of the distributed objects interactions aspects, including objects state
    - Interoperability, Portability, language and platform transparency
    - Platform independent Object model
    - Dynamic and Strongly typed interfaces
    - Realtime
    - QoS management
    - Location Transparency
    - Specialized Filtering and domain management capabilities applying to event driven communication only
    - Security, Transaction, Time, Fault tolerance, Openess (Interworking with other technologies)
DDS and CORBA Design Principles,
Great Differences in Design

- DDS more than a Pub/Sub Middleware
  - DDS is a data broker and a data manager
- CORBA more than a Request Broker
  - CORBA manage Object states too (data)

- Design principle, the Integration of data distribution, data management, and data location functions
- Design principle, the Separation between Object management functions

- Data management functions are defined as profiles
  - Data LifeCycle,
  - Data Query,
  - Local Object Relationship management,
  - Data Concurrency (ownership),
  - Data Persistency,
  - Data (topic) discovery,
  - Data Dissemination ...
- Object management functions are defined as Common Object Services
  - LifeCycle CoS,
  - Query CoS,
  - Object Relationship CoS,
  - Concurrency CoS,
  - Persistent CoS,
  - Naming/Trading CoSs
  - Notification, Event and Logging CoSs
DDS with a CORBA infrastructure?
Any attempt should at least reuse Notification concepts

- It can use iiop multicast
- What can be done with an NS
  - DDS Typed Data exchange
    - Typed Events
  - DDS Sharing Subscription
    - Notification Sharing Subscription
  - DDS Topic discovery
    - Dynamic Discovery of Event Type (Event Repository)
  - DDS Content subscription
    - Filtering
    - The use of SQL is possible
  - DDS Reliability
    - Event Reliability notification QoS
  - DDS Domain partition
    - Event Domain management QoSs
  - DDS data deadline
    - Event deadline, with no guarantee
  - All the other DDS QoS can not be directly mapped to Notification QoS

And with an Event-based Implementation things are even worse
Beyond QoSs, What can’t be Achieved with Notification

- Rebuilding of dataObject
  - should be handled by Application
- Aggregation of data items
  - Should be handled by the application
- Propagation of Atomic data items
  - Could be done with the help of transaction service,
    - Could impact performance.
- Partial modification of a data object
- Data change state notifications
- Data ownership
- Handling of multiple datatypes
  - Should be handled by the application
- Realtime propagation
- Support of the most current data values
  - Notification does not support the LIFO policy,
- Dynamic discovery of datatype
  - Could be done by using a Trader Service
  - Trader management should be handled by application
- DataHistory
  - Could be done with the integration of Query Service
    - Query management Service should be handled by application
Building full DDS-like architecture with existing CORBA infrastructure and services is very difficult.

- Several basic CoSs (COTS) are not available on market anymore (Query, Persistence, Lifecycle …).
- Application level have to endorse the integration of the CoSs (Notification, Query, LifeCycle, Persistence) to build data-items, coordinate their state and build additional features.
- No Realtime Typed Notification Service implementation available.

Performance, Fault-tolerance, and high-availability, continues to present a concern:

- No major FT-CORBA COTS available
- FT-CORBA impact Performance
- ... ROMiop and Miop are not the clear favorites in the race for the definition of a DDS interoperable wire protocol …
Where CORBA Can Complement DDS?

- Request/Reply transportation
  - Application will should handle request processing
    - Use topic keys like RequestID/ReplyID
    - Correlate Requests with replies
      - …
  - Don’t offer an object oriented view of the Server interfaces …
  - Synchronous remote invocations are not possible

- CORBA and DDS are complementary and compatible
DDS CORBA Interworking
The Problem to solve

- Carry DDS data items to the CORBA world
- Carry CORBA invocations to the DDS world

```
Struct track {
    TrackID id; // key
    Position p;
}
```

```
Interface SignalIntruder{
    Void Signal (track c);
}
```

Execution flow:

1. DDS FooReader FooWriter Classes
2. Application typed helper functions
3. Application data types
4. CORBA generated Stubs and Skletons

Splice DDS IDL Compiler

ORB IDL Compiler

Application mixing DDS & CORBA
DCPS CORBA Interworking: Some Work to be Done

If the CORBA - DCPS interworking is handled by application developers:

- From CORBA to DDS
  - Each CORBA Call need to be converted to a topic
    - CORBA, op(x1,x2) -> To DDS topic(x1,x2)
      - Restrictions: Xi could not be OBV nor a Struct
      - ValueType can not be defined at the DCPS level

- From DDS to CORBA
  - Data instances are , either mapped to RMI parameters or used to build an Object request (DII)
  - DDS Typed Interfaces makes the design of Generic bridging technology difficult,

- First Solution
  - Half of the Corba-DDS bridge could be made generic easily
  - Complicates the Application Programming Model and limits the application portability
  - Error-prone

- Second Solution
  - Bridge Code Generation
    - Using IDL topic description and Corba Object IDL interfaces

- Third Solution
  - Interoperability handled at the transport level
  - Encapsulation of the DDS Message in GIOP
  - Encapsulation of the GIOP Message in DDS
  - Main issue: DDS does not provide any common transport protocol, yet .. !!
DLRL: Toward Smooth Integration Between DDS and CORBA

- DLRL Objects maps to ValueTypes
- DLRL does not directly distribute these valuetypes over the network
- CORBA can distribute valuetypes
  - DLRL Objects can be passed as arguments to Remote Method Invocation, or
  - DLRL Object can be wrapped into a CORBA object to make it reachable remotely
  - A CORBA-DDS container could be defined or generated to wrap the DLRL objects
SPLICE-DDS: an Example of a SMART DDS-CORBA Interworking

IDL, a Common definition language:
- For CORBA-interfaces & DDS topics,
  - same IDL, name-space, same IDL to native language mapping
  - Code generation: Typed data generation as well as (typed-)interfaces

Seamless Runtime Cooperation
- Shared types allow direct passing-on of RPC-obtained information into DDS-topics
- Fully autonomous runtime-systems (no dependency, no real-time influence)
Conclusion

- Building full DDS-like architecture with existing CORBA infrastructure is not viable
  - Require intensive development and Finalization of Other technologies …
    - Realtime notification service, …
- Building a full CORBA-like architecture with DDS infrastructure is not viable either
- DDS-CORBA Interworking is well suited for growing demand of data centric and service centric mixed architectures
  - Network Centric Warfare, etc …

- DDS is suitable for Tier-2 type of application
  
  Fast Information Access Essential
  - Command Execution in CMS
    - Distribution of Track updates and navigation data to OA components
  - ATM, Inter-site connection

- CORBA is more suitable for Tier-3
  
  Service Centric Architecture Essential
  - Command Control & Support
  - ATM, Site-2-Site connection

- SPLICE-DDS can make DDS-CORBA interworking easier
  - Use same name space and IDL for both DCPS-topics and CORBA Object Interfaces
  - Future development of SPLICE-DLRL would make the integration more natural
Questions and Additional Information

- We are very interested by your feedback on CORBA-DDS interworking
  - Your requirements,
  - Your suggestions,
  ...
- Downloads and More information on SPLICE-DDS available at [www.prismtech.com](http://www.prismtech.com)
- Off-line questions can be sent to ramzi.karoui@prismtech.com and Hans van 't Hag at hans.vanthag@nl.thalesgroup.com
- Much more information to come on SPLICE-DDS with Hans’s next presentation