Policy-Driven

DISTRIBUTED MANAGEMENT ARCHITECTURES
(Towards Open Distributed Management Architecture)

“Integrating Management and Distributed Object Technology”

Kazi Farooqui

(farooqui@csi.uottawa.ca)
Department Of Computer Science,
University of Ottawa,
Ottawa K1N 6N5, Canada.
1. Integrating Management Models and ODP Model

Model for Distributed Management:

Synergy of:

1. Existing Management Models
2. Distributed System Architectures such as RM-ODP.

WHAT ODP Concepts + Principles + Architectural Framework can be used to enhance the OSI Management Model towards a “Distributed Management Model”?

Figure 1. Integration of Distributed Object Technology in Network Management Models
2. MOTIVATION

❖ Why DISTRIBUTED OBJECT TECHNOLOGIES:

☞ Management Activity: Inherently DISTRIBUTED in nature.

☞ Management Applications: Advanced DISTRIBUTED PROCESSING APPLICATIONS.

☞ Management Platforms: Require numerous DISTRIBUTED PROCESSING functions (Transaction, Replication, Group, Event Distribution, Location Transparency, etc.)

❖ How does ODP Help............


☞ META-STANDARD: Applicable to Different Application Domains (Telecommunications, Network Management, Manufacturing, etc.)


☞ STRUCTURING PARADIGM: Different Viewpoints provide clear separation of concerns and focus on specific management aspects. (De-coupling of Management Applications and Management Platform).
Limitations of the OSI Management Model:

- Primitive Point-to-Point Model. (Support for traditional Point-to-Point Management Activity).
- Tied to a specific Communication Platform (CMIS/CMIP). (Tight coupling between management operations and communications infrastructure).

Towards Distributed Management Models:

- Object-Machines for Configurable Distributed Management Platforms.
- Hierarchical Management Models.
- De-coupled Management Applications and Management Platforms.
- Policy-Driven Management Models.
- Management Domains.
Figure 2. ODP Modelling of Management Systems
FOCUS is on:


ODP Computational Model: A Framework for the Definition of Management Application as a Distributed Object World interacting by invoking Operations/Notifications at Interfaces in a “distribution-transparent” manner. Identification of:

1. Management Application Components.
2. Multiple Interfaces of Components.
3. Interactions between Interfaces (Operations, Notifications).

ODP Engineering Model: A Framework for the Definition of an Object-Based Distributed Management Platform. Modular Platform Support for Management Applications and of their interactions:

2. Transparency Support Infrastructure.
3. Domain-Specific Support Infrastructure.
5. Dynamic and Flexible Management Infrastructure.
5. Basic Computational Modelling of OSI Management

What does ODP Computational Model offer to Management Model:

- Basic Object Model.
- Interaction Model.
- Type Model.
- Binding Model.
A Management Object offers multiple interfaces in different roles.

Roles are associated not with Objects but with their Interfaces.

Management Object may act as “Manager” on one interface and as “Managed Object” on another interface.

Supports the Modelling of “Hierarchical Management Applications”:

1. Hierarchy of managers managing a set of distributed managed objects.

2. Delegation of responsibility from a manager to a sub-manager. (Delegation of Management).
INTERACTION MODEL:

Management Application is an object-based distributed application.

Application components interact by sending operation invocations (ODP Interrogations) or notifications (ODP Announcement) at their interfaces.

Management Interfaces specify:
1. signature: management operations and notifications.
2. behavior: ordering of operations and notifications = Protocol between interacting management interfaces.
3. environment constraints: QoS, Distribution Transparency, etc.
   (Notifications be delivered within 5 msec.)
   (Notifications be delivered to a group of managers).
   (Operations to be delivered to all or none of managed interfaces).
Distributed Management Computational Model is Backwards compatible with the existing Point-to-Point OSI Management Model:

1. Managing System:
   a. Management Object = single-interface object with “manager role”.

2. Managed System:
   a. Management Object = Agent Object with a “managed role” interface and a “manager role” interface.
   b. Managing System “manager interface” is bound to the “managed interface” of Agent Object.
   c. Agent offers a local “manager interface” bound to MO.
Figure 6. Type Model: Interactions based on interface type matching.

- Management Applications are complex applications which can be created out of available components.

- A repository of management components and their interface types enables faster application creation.

- Management Application creation/composition based upon type-checked (late) binding of (interfaces of) application components.

- Substitutability of one application (component) by another. (A manager interface can be substituted by its sub-type).

- Current ODP Type Model: Based on operation signature compatibility. Subtyping can be extended to QoS compatibility and behavior compatibility to meet the needs of management applications.
BINDING MODEL:

Support for management application-specific interaction pattern between multiple managers and managed objects.

*Binding Objects* enable complex binding structures to be supported between management application components. (Multicasts of Notifications issued by an Event Forwarding Discriminator to multiple managers).

Applications have control on *configuration* and *quality of service* of the complex binding (Add / Delete Managed Objects from the configuration).

What does ODP Engineering Model offer to Management Model:

☞ Model of an Object-Based Distributed Platform:

1. Concepts, Rules, Structuring Framework (of nodes, capsules, clusters, objects) for the organization of a Management Platform that supports execution and interaction of Management Application Components.

☞ Distribution Transparency Support Mechanisms:

1. An engineering framework of “transparency support mechanisms” for the provision of location transparency, migration transparency, transaction transparency, group transparency etc. to the interactions between Management Application components.

☞ Model of a Configurable and Policy-Driven Platform.
Basic Engineering Concepts of OSI Management:

1. **Basic Engineering Object**: Engineering representation of “management computational object”.

   ![Diagram of Computational Object and Cluster](image)

   Figure 8. Computational Object and Cluster: A powerful modelling paradigm.

   **Legend**:
   - **mco**: management computational object
   - **meo**: management engineering object

2. **Cluster**: A group of related management objects at a node. (A set of related managers synchronized via internal interfaces and accomplishing a common management function OR a set of MOs related to different aspects of a single resource).

3. **Cluster Manager**: Part of Agent Functionality.

4. **Capsule**: A group of “management clusters” with unrelated management responsibilities.

5. **Capsule Manager**: A Super-Agent.

6. **Nucleus**: Access to CMIP communication facility.
7. **Node**: An open system consisting of both manager and managed interfaces.

8. **Stub Object**: A marshalling/information modification functionality that transforms management operations into messages that are exchanged as CMIP PDUs.


10. **Protocol Object**: CMIP and the underlying stack of OSI/TCP-IP communication protocols.

11. **Channel**: A configuration of stubs, binders and protocol objects between manager and managed interfaces is the key to the design of a policy-driven management platform.
Extending the current OSI Management Model:

Engineering Functions clubbed in the Agent:

1. Scoping (performed in a local system)
2. Filtering (performed in a local system)
3. Synchronization (performed in local system).
4. Event Notification
5. Object Management.

Scope of these functions can be extended in a “distributed management context:

2. Synchronization -----> ODP Transaction Function
3. Event Notification -----> ODP Event Notification Function
4. Object Management -----> Cluster Manager/Capsule Manager
SCOPING:

Scoping can be performed on Managed Objects (MIBs) in multiple systems using the “ODP Group” concept.

Figure 9. Grouping and Scoping in a Distributed Management System.

MIBs on distributed systems are organized as an “ODP Group” with Base-Level MOs as members.

A management operation multicast by a manager is received by all group members.

Group Invocation: Addressing and selection of base-level MOs in different systems.

Scoping: Selection of MOs in the Local System.
SYNCHRONIZATION:

- In a distributed management model, manager requires a management operation performed atomically on all MOs distributed on different open systems.

- Synchronization can be performed on Managed Objects (MIBs) in multiple systems using the “ODP Transaction” concept.

![Figure 10. Transaction in a Distributed Management System](image)

- Management operations emitted by the manager are intercepted by the Transaction Function.

- Transaction Protocol is carried over CMIP using M-Action PDUs between the involved manager and managed systems.

- Local Synchronization is performed by LSF.
Both the MOs (in different systems) and the Manager subscribe to the “ODP Event Notification Function” to send and receive events.

ODP Cluster/Capsule Manager perform the object management functions such as MO creation, deletion, checkpointing, deactivation and reactivation.

IN ALL THESE CASES AGENT WORKS AS AN INTERCEPTOR OF MANAGEMENT OPERATIONS.
7. Policy-Driven Distributed Management Models

- Basic Components

1. Monitoring: To obtain information from management interfaces of physical resources

2. Domains: To group management objects and to partition responsibility.

3. Policy: To permit the behavior of automated managers to be modified without re-implementation.

- Management Domain

  A “set of objects” which are related because a certain part of their “behavior” is controlled by the same “authority” (e.g. telecom stakeholder) under a set of “management policies”.

  - Autonomy (Authority)
  - Policy
  - Behavior/Aspect
  - Addressing Unit

X-Aspect Management Domain, Y-Aspect Management Domain

“Start_Test() on all domain members for which the predicate “x-attribute > n” is true.”
Why Domains

1. To group the objects related to a particular service or function.
2. To group the objects to which a policy applies.
3. To specify boundaries of management responsibility or authority.
4. To provide a “naming context” in which managers may assign local names of their choice to the objects they manage.
5. To reflect organizational, geographic or network structure.

Domain Membership

1. Create a domain
2. Delete a domain
3. Add an object
4. Delete an object
5. List members
6. Change member name
7. Query domain’s parents.
Policy-Driven Management

1. **Mechanisms vs. Policies**: Separation of “management policy” from “Automated Managers” which interpret the policies.


3. **Interpretation of Policies**: Interpretation of policies into management actions.

![Policy-Driven Management Diagram]

**Legend:**
- OMI: Object Management Interface
- MPI: Management Policy Interface
- PAI: Policy Administration Interface

**Figure 12. Policy-Driven Management**: Managers interpret Policy to perform management
Figure 13: Computational Model-1: Policy-Driven Managers

Legend:
- OPR: Operation Invocation
- NTF: Notification Invocation
- REP: Reply Invocation
- MMI: Membership Management Interface
- PMI: Policy Management Interface
- DMI: Domain Management Interface
- OMI: Object Management Interface

→: Manager Interface
←: Managed Interface
Figure 14. Computational Model-2: Policy-Driven Managed Objects
Figure 15. Policy Driven Manager and Managed Objects

Legend:
M: Manager Object
MO: Managed Object
PCR: Policy Conflict Resolver Object
GSM: Group Support Machine
ODP Distribution Transparencies in the Management Model:

Access Transparency: Management Applications invoke CMIS service through standardized service primitives.

Location Transparency: Requires location-independent naming model in the management domain.

Migration Transparency: Highly dynamic mobile telecommunications environments.

Transaction Transparency: Management Operations (such as Suspend_Test(), Resume_Test(), Set_Attributes()) require all-or-nothing semantics in their execution.

Group Transparency: Identical management operations on a set of distributed Managed Objects.

Persistence Transparency: Reactivate both manager and managed objects when a notification or an operation is invoked on them.
7. CONCLUSION

Management constitute a significant application domain of ODP. ODP Provides:

☞ Distributed Object Model.
☞ Modeling Frameworks (viewpoints).
☞ Interoperability Model.
☞ Distribution Transparency Model.

Increasing convergence taking place between management models and distributed computing (such as ODMA).

☞ ODMA: Object-Based Management Architecture

- Distributed Management Model
- Technology-Independent
- Management Policy-Driven Platform
- Configurable Platform.

“Distributed-Object Computing Technologies” have a major role to play in the definition of tomorrows MANAGEMENT ARCHITECTURES.