



**Poseidon House
Castle Park
Cambridge CB3 0RD
United Kingdom**

TELEPHONE:
INTERNATIONAL:
FAX:
E-MAIL:

**Cambridge (0223) 323010
+44 223 323010
+44 223 359779
apm@ansa.co.uk**

ANSA Phase III

A Guided Tour of RM-ODP Part 3 (Prescriptive Model)

Andrew Herbert

Abstract

This document is a slide presentation to accompany the second CD for ISO 10746-3 Basic Reference Model for Open Distributed Processing - Part 3: Prescriptive Model (July 1993).

The presentation reviews the major concepts in the same order as they appear in the CD with additional remarks giving rationale for the organization and content of the reference model.

APM.1052.01

Approved
Standards Contribution

16 June 1994

Distribution:

Supersedes: APM.1040.00.03

Superseded by:

Copyright © 1994 Architecture Projects Management Limited
The copyright is held on behalf of the sponsors for the time being of the ANSA Workprogramme.



A Guided Tour of the Basic Reference Model for Open Distributed Processing (Prescriptive Model)

**Andrew Herbert
(Editor)
(Technical Director, APM)**



Context

- **Part 1: Overview and guide to use**
- **Part 2: Descriptive Model**
- ***Part 3: Prescriptive Model***
- **Part 4: Architectural Semantics**
- **Trader**



Scope

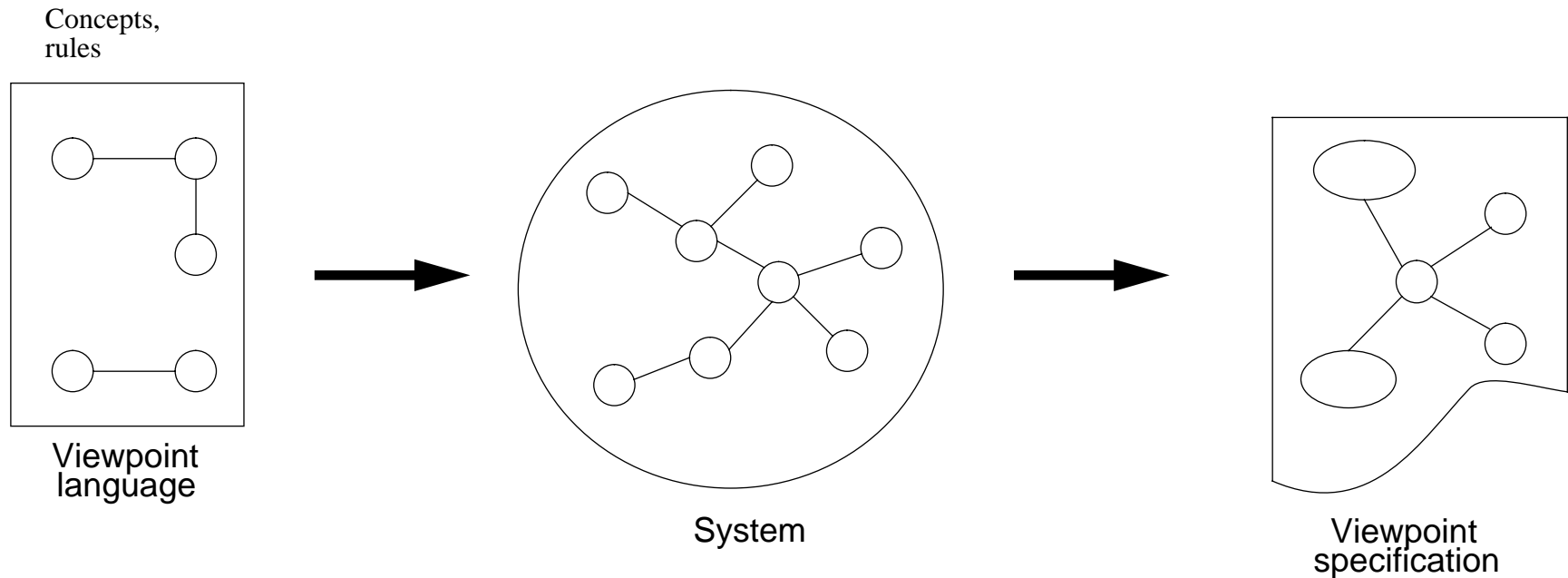
- **concepts and rules for specifying ODP systems**
- **framework for development of ODP standards**
 - **standards for *specification, modelling* and *programming* languages**
 - ***language bindings* (APIs) for ODP systems**
 - **functional components of ODP systems**



Languages and Viewpoints (1)

- ***Framework*** consists of five *viewpoint languages* and a set of *ODP functions*
- Each viewpoint language enables specification of an ODP system or component
- Languages are structured so that *consistency checking* between alternative viewpoint specifications is possible
- The chosen set is *necessary* and *sufficient* for needs of ODP
- Each language consists of *concepts* (vocabulary) and *rules* (grammar)

Languages and Viewpoints (2)



- **Mathematical basis in *projection* of a set of concepts and abstraction/ specialisation relations over a semantic net.**



Viewpoints

- **Enterprise - purpose, scope and policies for a system**
- **Information - semantics of information and information processing in a system**
- **Computational - functional decomposition into objects suitable for distribution**
- **Engineering - the infrastructure required to support distribution**
- **Technology - the choice of technology to support distribution**



Conformance

- **Reference point - a potential conformance point**
- **Positioned by computational and engineering languages**
- **Specified in (some combination of) enterprise, information, computational and engineering languages**
- **Information specification determines universe of discourse for the reference point**
- **An ODP system is one which conforms to ODP standards at all reference points asserted to be conformance points**



Enterprise language

- **Role: agent, artefact**
- **Policy maker, administrator, arbitrator**
- **Resource, resource manager**
- **Community: set of objects with an objective (*contract*)**
- **Performative action**
 - **incur an obligation**
 - **fulfil an obligation**
 - **waive an obligation**
 - **acquire permission**
 - **be forbidden**



Federation

- **Federation: a *voluntary* community in which administration is a cooperative activity on a peer to peer basis**
- **Federation principles**
 - freedom to join
 - freedom to leave
 - subject to agreed obligations
 - retain significant autonomy
 - no single administrator
- **Federation colours many of the technical aspects of the Reference Model.**



Information Language

- **Concepts, relations**
- **Integrity rules**
 - **static schema (state and structure)**
 - **invariant schema (independent of behaviour)**
 - **dynamic schema (possible behaviour)**
- **Model behaviour as changes in class membership**



Computational language

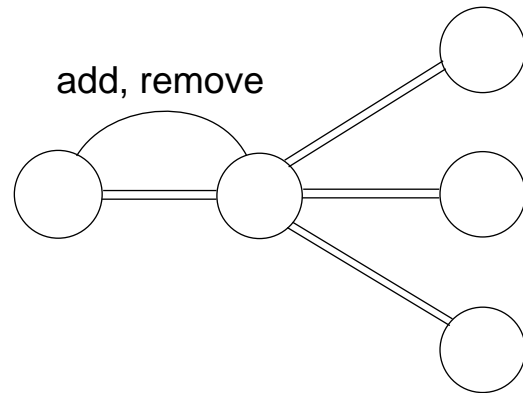
- **Interface**
 - **Operational (interrogations)**
 - **Transactional**
 - **Stream (unstructured)**
- **Interface template**
 - **Signature (operations or stream)**
 - **Behaviour for operational interfaces**
 - * **ordering relative to other invocations**
 - * **conflicts relative to other transactions**
 - * **operation behaviour**
 - **Environment constraint**
 - **Roles: client, server, producer, consumer**



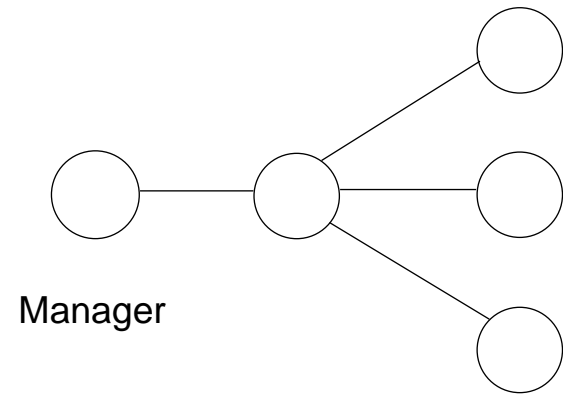
Binding

- ***Implicit* binding for operational interfaces**
- ***Explicit* binding for all interfaces**
- **BIND <binding type> {set of interfaces}**
 - **instantiates a new binding object**
- **Binding object has a control interface for supervising the binding**
 - **adding/removing interfaces**
 - **stopping/starting information flows**
 - **monitoring events**

Binding examples



Stream multi-casting



Managed objects

Management domain



Operational interface typing

- operation invocation is a *request-reply* model
- type checking for safety - no surprises principle
- server must provide at least operations required by client
- client must accept all possible terminations from server
- client arguments “smaller” than server requires
- server results “smaller” than client accepts
- arguments and results can be interface identifiers
- each operation has one or more *terminations*



Transactions

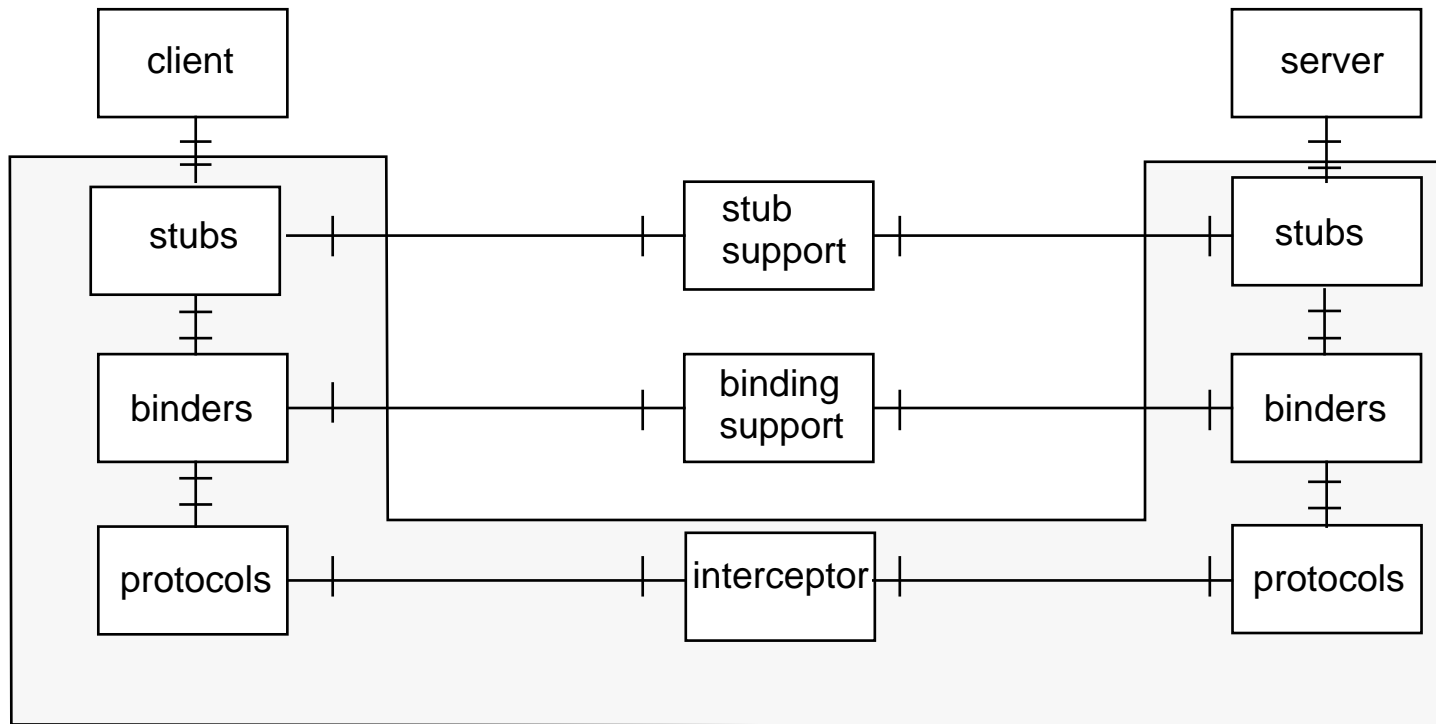
- **Very general model - specific models defined in terms of**
- ***visibility* - how much interaction possible with objects outside the transaction**
- ***consistency* - invariant to be fulfilled at end of transaction**
- ***recoverability* - how much of transaction is undone after failure**
- ***permanence* - the degree to which failures can alter the effects of a transaction**
- ***dependency* - influence of other transactions on success or fail**
- **bracket transactions as operations**
- **label terminations *success or fail***
- **spawn actions generate independent transactions**



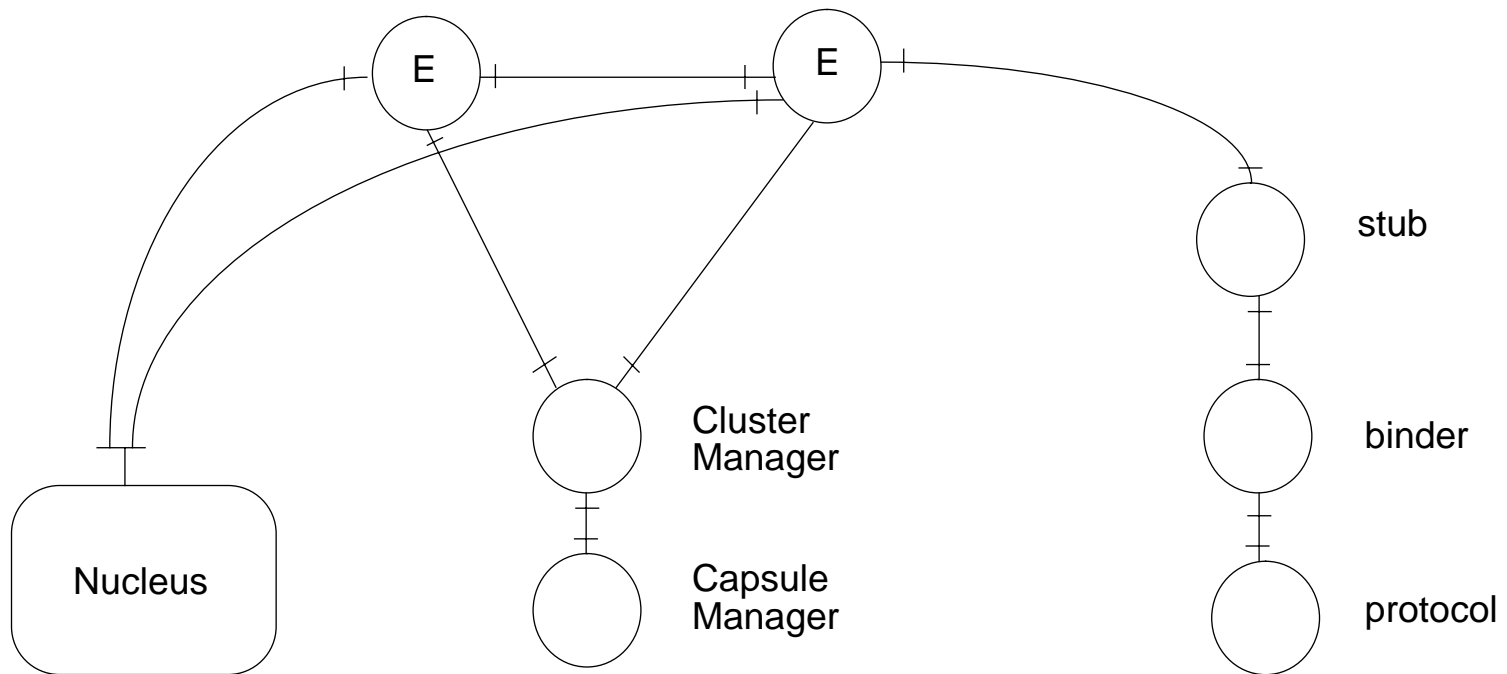
Engineering language

- **structures for implementing infrastructures supporting *selective transparency***
- ***channels* for communication**
 - simple client server
 - multipoint channels
 - stream channels
- **clusters for activation, deactivation, migration**
- **capsules for resource allocation, protection**
- **nodes for network addressing**

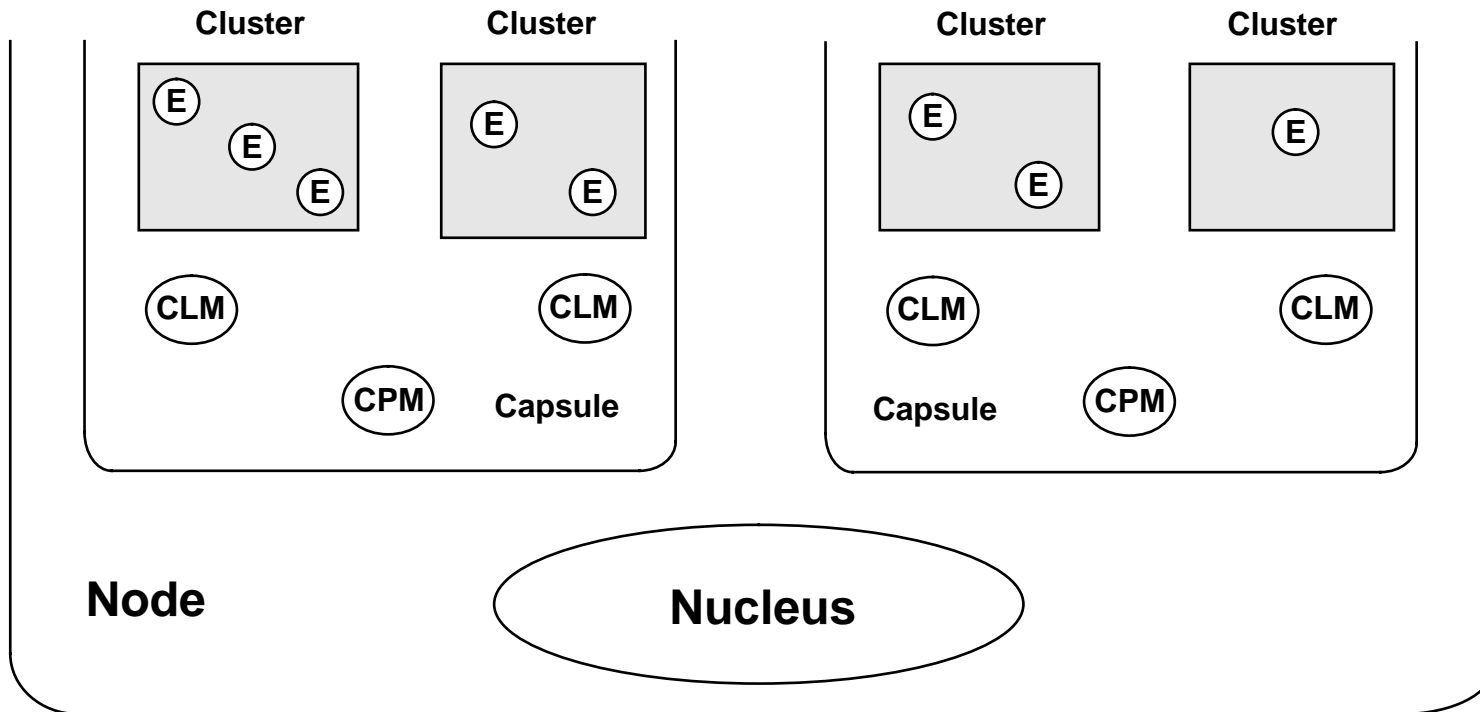
Client-Server Channel



Key engineering objects



Engineering structures





Engineering a binding

- **Engineering location**
 - location of protocol object interfaces
 - *binding template* for stubs and binders
- **Engineering location of an interface which can migrate is volatile**
- **Interface reference provides basis for (re) establishing binding - process may be multi-stage**
- **Interface reference**
 - identifies binding template to instantiate
 - identified communications end points at which to start binding process
 - engineering interface identifier
- **Engineering interface identifier**
 - unambiguous
 - synonyms permitted
 - equivalent undefined



Technology language

- **Implementable standard: template for a technology object**
- **Implementation Extra Information for Testing**
 - **technology specification in a standard defines a proforma IXIT**
 - **IXIT defines templates and names of interfaces required for testing**



Consistency constraints

- **Can never be complete**
 - enterprise and technology aspects are based on *connotation*
 - information, computational and engineering aspects are based on *denotation*
- **Rules give a least guarantee**
 - give a transformation between a specification in one viewpoint language and another



Functions

- **Management (object, cluster, capsule, communications, interface reference)**
- **Coordination (transactions, groups)**
- **Repository (Storage, relocation, types, trader)**
- **Security**
- **Transparency**
 - **recipes for using other functions to extend transparency of the infrastructure**



Management Functions (1)

- **nucleus**
 - threads
 - channel setup
 - capsule manufacture
- **object management**
 - objects manage themselves
 - cluster managers ask objects to terminate, snapshot etc.
- **cluster management**
 - deactivate, migrate, terminate



Management Functions (2)

- **capsule management**
 - cluster instantiation, reactivation
 - termination
- **communication domain**
 - link to network management
- **interface reference**
 - tracking interface references
 - enables distributed garbage collection



Coordination functions

- **Transactions**
 - **transaction manager**
 - **concurrency manager**
 - **deadlock manager**
 - **stable store manager**
 - **Log manager**
- **Groups**
 - **distribution**
 - **collation**
 - **ordering**
 - **fault detection**



Repository functions (1)

- **Separate out components of repository to enable distribution**
- **Storage function**
 - data storage
 - deactivated cluster repository
 - interface reference fixing done on reactivation
- **Relocation function**
 - supports binder objects
 - retains interface reference to engineering location bindings
 - retains activation process for deactivated (checkpointed) clusters

Repository functions (2)

- **Type repository**
 - represent types as interface to type repository
 - each type can be queried for its relationship to other types
- **Trader**
 - exporter defines type and interface reference
 - importer defines type for trader to match
 - type: at least computational - typically includes other properties
 - trader policy: searching, federation
 - exporter policy: e.g. for resource control
 - importer policy: e.g. to direct/limit search



Security

- **Access control**
- **Audit**
- **Authentication**
- **Integrity**
- **Confidentiality**
- **Non-repudiation**



Transparency

- **Access (marshalling/unmarshalling)**
- **Transaction (for resilient objects)**
- **Location**
- **Failure (checkpointing, replication)**
- **Federation (masking administrative boundaries)**
- **Migration**
- **Group**
- **Resource (transparent deactivation/reaction)**



Status

- **1988 First Working Documents**
- **November 1992 First Committee Draft**
- **June 1993 Second Committee Draft**
- **February 1994 Draft International Standard?**
 - **structures for streams?**
 - **even treatment of all functions**
 - **flesh out transparency recipes**
- **1995 International Standard**
- **Populated via OMG activities**
 - **liaison in place**
 - **function correspondences identified**