A Methodology for Developing Integrated Multi-domain Service Management Systems

Vincent Wade Trinity College Dublin, David Lewis University College London , Mark Sheppard Broadcom Eireann Research Ltd, Michael Tschichholz & Jane Hall GMD Fokus

Contact: Vincent.Wade@cs.tcd.ie

Abstract

This paper presents a methodology for designing and implementing Intra and Inter Domain (end-to-end) service management systems. In particular the methodology considers the integration & cooperation of management services from different service providers. The methodology uses enterprise modelling techniques and object oriented design for requirements capture and description, system development, implementation and deployment. It also supports the generation of service management system specifications based on ODP viewpoints. The methodology facilitates the reuse pre-existing computational components e.g. TINA C service architecture components in developing and implementing management solutions. To illustrate this methodology a case study is presented based on the development of subscription management services where the final delivered management service is based on the co-operation of different provider service management systems. The methodology was developed and is being trialled within the PROSPECT ACTS project.

1. Scope

With the deregulation of telecommunication network services in Europe, there is increasing interest in telecommunication services being offered by third party service providers over different network providers. The benefit of such an open service environment would be 'one-stop-shopping' delivery of 'tailored' services to end customers without these customers having to deal with the multiplicity of underlying telecommunication services and network providers. The difficulty with such an environment is the complexity of managing the services across the different service provider organisations (administrative domains) i.e. integrating co-operative management services spanning individual service and network providers, value added service providers and final end users. The design facilitates both the design and development of individual management services and their co-operation and interoperation to support source-to-destination telecommunication services. The methodology facilitates the usage of Open Distributed Processing standards for the generation of system specification & the re-use of management component designs and implementations based on the TINA C standards. However the methodology itself is independent of these standards.

The paper first identifies the current trends in system analysis/design techniques and ODP based distributed systems development techniques. In accordance with these trends the paper describes the Prospect Design cycle which is at the center of the methodology. The paper also describes how ODP based specifications can be generated from stages in this design cycle. To illustrate the usage of the methodology the design of a multi domain subscription management service which spans several service and network providers is described. Finally conclusions are drawn as to the usage and benefits of the methodology.

2. Designing Multidomain Management Services

The late eighties and early nineties has seen the increased usage of 'second generation' object oriented analysis and design techniques. Principle among these

methodologies are Rumbaugh's Object Modelling Technique (OMT) [Rumb-91], Ivar Jacobson's Use-Case driven OO software Engineering Model [Jacob-92] and Grady Booch's Object Oriented Analysis and Design (OOAD) methodology. Another more recent object oriented methodology has been the FUSION methodology developed by HP Labs. The current trend in object oriented methodologies is to harmonise existing approaches rather than develop brand new modelling techniques. An example of this trend is the proposal of Unified Modelling Language. Also the Object Management Group (OMA) - the consortium responsible for CORBA standards, have recently called for standardisation of OO design techniques.

Since inter & intra domain Management Services inherently require distributed solutions, an open distributed approach must be adopted for multi domain *system description*. Currently several standards address the issue of specifying open systems. Principle among these is the Open Distributed Processing [ODP-94] standard which suggests the specification of five different viewpoints of the system (Enterprise, Information, Computation, Engineering and Technology). These viewpoints highlight, and provide a basis for the separate discussion of, different aspects of the design and implementation of the system. Several standards have taken these viewpoint description concepts and enhanced them for describing network and service management systems e.g. TINA [TINA-95] and ODMA.

In order to capture the idea that (management) services can be used independently of each other as well as being used in cooperation, the end-to-end management system must be viewed in two complementary (overlapping) models: (i) Inter Domain Management Model: This models the management services in co-operation to support end-to-end management services. (ii) Intra Domain Service Management Model: These models describe the management services (e.g. configuration, accounting etc.) of each of the tele-services in an Open Service Market e.g. management services for VPN, MultiMedia Conferencing, HyperMedia Services.

The reason for this decomposition is that each management service could exist on its own (i.e. is a usable management service regardless of its cooperation with other services management providers). This reflects the view of the Open Service Market where the management of a tele-service has its own objectives whether or not they are infact integrated across provider organisations Thus inter domain management captures the end-to-end management chain while the intra domain models capture the management services of each individual service in isolation. However, these models are not completely orthogonal as components used for inter domain management are also used in the intra domain management.

3. A Service management System Development Design Cycle from which Viewpoint Models can be derived

Rather than developing new modelling or systems specification techniques, the Prospect methodology harnesses existing modelling tools and concepts within a design cycle which facilitates the development of the inter and intra domain management systems. The methodology supports each stage of the design process lifecycle. The ODP viewpoints provide an well established approach to system model description. However, they do not provide a prescriptive methodology that can be followed in developing management systems [ITU-96]. Therefore unlike previous design methodologies which have attempted to specify design steps which allow the generation of one ODP viewpoint from a previous viewpoint, the

PROSPECT design cycle sees the development of a full object oriented model as central to the overall design process. Figure 1 illustrates the Prospect design cycle, which at its centre is concerned with the design, development, implementation, testing & trialling of inter/intra domain management system.

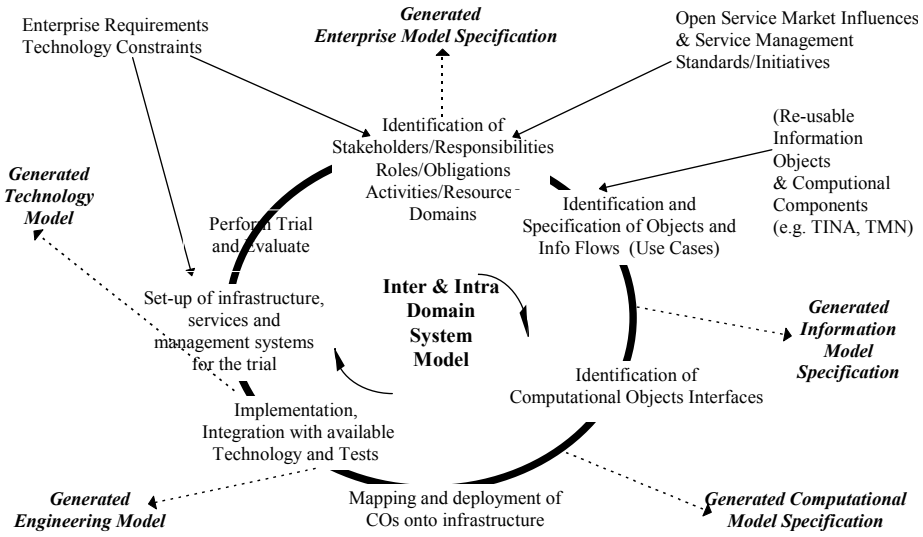

Figure 1: PROSPECT Design Cycle

The design cycle provides a structured way of developing, implementing and testing these object designs. It specifies the design steps from developing multi domain business models (which includes the representation of stakeholders, assignment of responsibilities, identification of obligations and activities etc), use case definition analysis, object identification and relation representation, definition of computational components, the integration and extension of pre-existing computational components (e.g. from TINA C Service Architecture), distributed placement of computational components, definition of platform architecture and platform services, generation of test sets and trial execution.

The Prospect design cycle also identifies key places in the development process from which specific viewpoint models can be generated and prescribes the contents of each viewpoint model specification and illustrates how these can be generated. The information needed to generate the specifications are a subset of the information needed to design and implement the actual systems. The design cycle ensures the consistency between each stage of the system model development and therefore provides a means of tracing the interrelationships between the ODP viewpoints. The benefit of generating the ODP viewpoint specifications is that a clear separation of the different aspects of the system can be captured. Also it provides a structured means of comparing different subsystems and services. The Prospect design cycle is iterative, allowing progressive deepening of the system models by iterating the cycle several times.

The modelling techniques integrated in the design cycle are *Organisational Requirements Definition for Information Technology* [ORDIT-93] , Object Modelling Technique (OMT) [Rumb-91], OMG based interface definition language. The methodology has been influenced by the modeling work of previous ACTS projects

PREPARE [Hall-96] and PRISM [Berq-96] and the TINA-C Service Architecture [TINAC-94] and modelling approaches.

4. Iterating the Inter-Domain System Development Methodology

This section illustrates the design decisions and experiences encountered when applying the Prospect design cycle. The example for this illustration is the design of multi-domain management services for a tele-educational service (which is itself composed of several teleservices e.g. Hypermedia information service, video conferencing service and makes use of connectivity services e.g. VPN service and ATM service). The example is based on the work of the Prospect ACTS project ACT 052 which are currently being trialled within this project.

4.1 Business Model and Use cases

The methodology for deriving the PROSPECT business model is based on that of the ESPRIT project ORDIT (Organisational Requirements Definition for Information Technology). ORDIT developed a model with roles, responsibilities and obligations that can be used for establishing relationships between the various parties involved in a socio-technical system which captures organisational requirements. It can be used at a number of different levels of abstraction and is iterative, allowing for revision and growth. Figure 2 illustrates the contractual relationships between the stakeholder organisations for the case study.

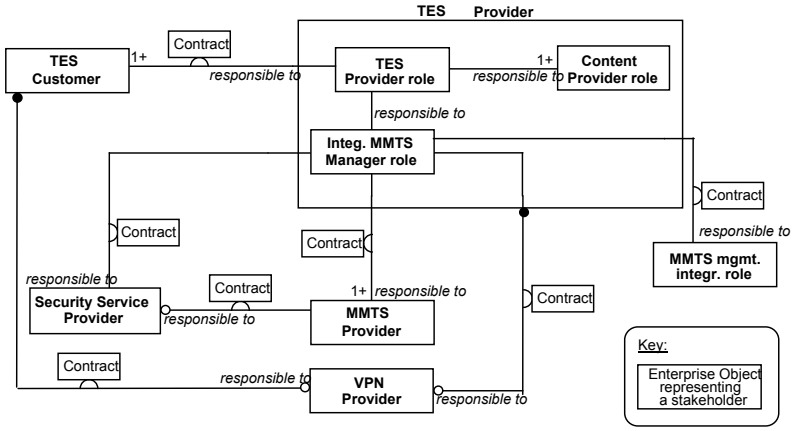

Figure 2 Contractual Relationships between Stakeholder Organisations

Once the model of stakeholder and roles had been established, the requirements on the stakeholder's systems were focused on through the definition of use case for the customer, provider and end user roles of the TES stakeholder. These use cases included; subscription to the TES, inclusion of a customer network site in a TES subscription, authorisation of a TES end user under a subscription and actual use of the service. To assess the inter-domain implications of these use-cases, i.e. the requirements they placed on the different stakeholders in the enterprise model, high level sequence diagrams were drawn up to help define the information that needs to flow between the different stakeholders and roles. An example of such a diagram for the "authorise a TES end user" use case is show in figure 3.

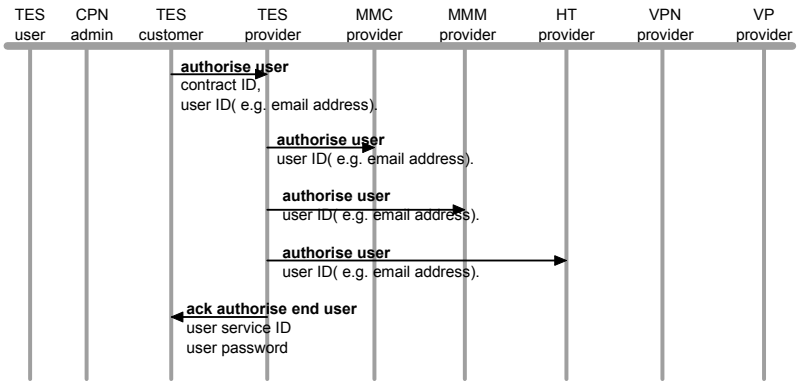

Figure 3 Use Case information flow between stakeholders

4.2 Reuse of Existing Models

At this stage functional requirements for the systems within each stakeholder that would be involved in the use cases had been outlined so other existing management system specifications were analysed to see if they could be reused in meeting the requirements. This is in line with current management system methodologies, e.g. TMN M.3020 and the NMFs Ensemble approach, which aim to make maximum reuse of existing functional components specifications and information model when designing new management system.

However, in the particular service management areas covered by the use cases, little was available in the way of existing specification, either in the TMN functional and information specification or in NMF solution sets. The TINA Consortium had however been examining areas of service management in detail in its service architecture (SA) [TINA-94]. In particular they has defined a generic service model for user telecommunications services that was closely integrated with management components for both subscription management and accounting management. This combined service and management model was seen as suitable for satisfying many of the requirements imposed by the use case for the Prospect systems and were therefore selected for reuse as the basis for the management systems of the TES and MMTS providers' systems. However, the TINA service architecture assumed only a single provider offering services to customers, whereas the use cases placed requirements on the TES system to integrate the MMTSs offered by other providers into a single service offering. This required the modification of the TINA architecture components used in order to suit the use case requirements.

The TINA service architecture is expressed in ODP viewpoints, and provided (i) an *Information Viewpoint* model in terms of information object (IO) descriptions and OMT object diagrams to express the relationships between the objects. (ii) a *Computational Viewpoint* model in terms of computational object (CO) descriptions and object block diagrams showing the client server relationships between objects.

Details of these viewpoints, expressed in TINA using Quasi-GDMO [TINA-95] and Object description Language [TINA-93] were not publicly available, however draft interpretations of some parts of the models by other projects were available.

The information and computational models were therefore required as the basis from which to extend the models to satisfy the requirements present in the use cases and

from which to generate a detailed design specification sufficient to implement the components required. It was found however that the TINA design specification in the form of these two viewpoints was inadequate for this task. This was primarily due to the lack of an explicit linkage between the two viewpoint models, i.e. the mapping between IOs and COs was not present in any clear manner. This prevented both a clear understanding of the system from being made and hid the overall object model needed to actually implement this system. The first step to resolving this problem was to use the use cases as the basis for sequence flow diagrams showing describing the flow of information between COs. This illuminated the dynamic aspects of the model and in the process clarified the relationships intended between the COs and IOs and their behaviour.

As COs are taken to be units of object distribution, some mechanism was required to map CO definitions to a form suitable for implementation on a distributed platform. The engineering viewpoint model for all TINA architectural components and defines a Distributed Processing Environment, providing distribution transparencies to engineering computational object based on the COs of the computational viewpoints. However, no practical implementation of the DPE platform implementation was available to the project. Instead a commercial CORBA 2.0 [CORBA-95] implementation (Orbix from Iona) had been chosen as the platform for the components based on the TINA SA. This required mapping between the multiple interfaces of a TINA engineering computational object to the single interfaces of CORBA objects. This mapping exploited the similarity between ODL and CORBA's IDL, with ODL CO interfaces being mapped individual IDL interface definitions, grouped in a module mapped from the CO definition.

The combination of OMT IO definitions, IDL definitions of CO interfaces and sequence diagrams showing interactions between COs via the IDL interfaces provided enough detail for developers to understand the TINA SA components selected for implementation. In these cases a single object model was deemed too costly to synthesis, so extensions to these SA components were specified using the same combination of notations. The following section provides more details on how these notations were used in practice by examining the extension of the TINA SA subscription management component to satisfy the multi-domain requirements of the use cases.

4.3 Extending Reused Components

Since the design of the TINA SA subscription management component had been presented as a presumably consistent set of structures IOs and CO, and since the relationships between these sets of objects have been clarified through detailed sequence diagrams, the extension of the component was most readily performed by using the same notational structure.

Figure 4 shows a portion the OMT object diagram for subscription management from the TINA SA representing the parts that were actually implemented in Prospect. The shaded objects shown are those which were added to the model to extend it to handle the multi-domain requirements on subscription management. In this case information relating to subcontracted providers and the mapping of subscription-related IO in one domain to those in subcontractors domains was required. The intention of these extensions was to support the functionality required, while preserving the integrity of the existing information model.

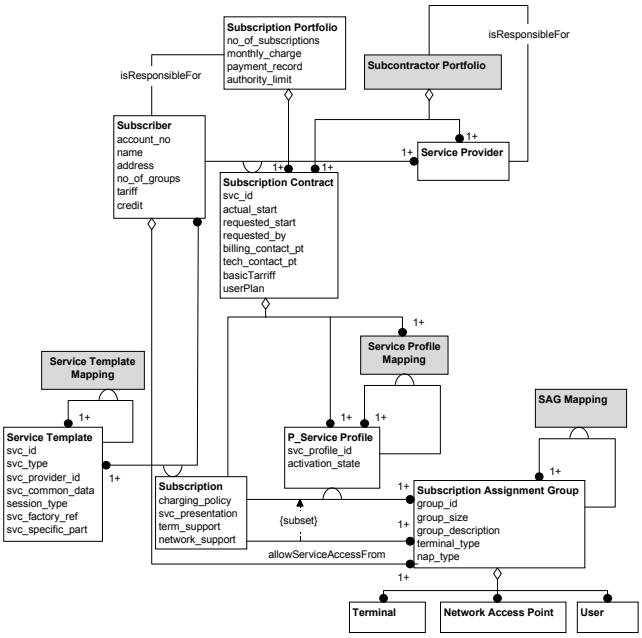

Figure 4 Extended Susbcription management information model

A similar approach was taken when applying the extension to the computational model. It was deemed advantageous to retain as much as possible of the interface definition of the existing COs when developing the extensions required. In this way components designed to interact with an original CO interfaces of the component (SubMgmt in the figure) could also interact with an extended component (SubMgmt* in the figure) with minimum modification. This was performed simply by designing SubMgmt* as a wrapper for SubMgmt, with the new COs introduced to implement this wrapper (shaded in the figure) inheriting IDL interfaces from COs in SubMgmt. The SubMgmt* COs provide the functionality needed to interact with SubMgmt components as used in other domains, thus exploiting the same CO interfaces and minimising the complexity of information processing that needed to be performed.

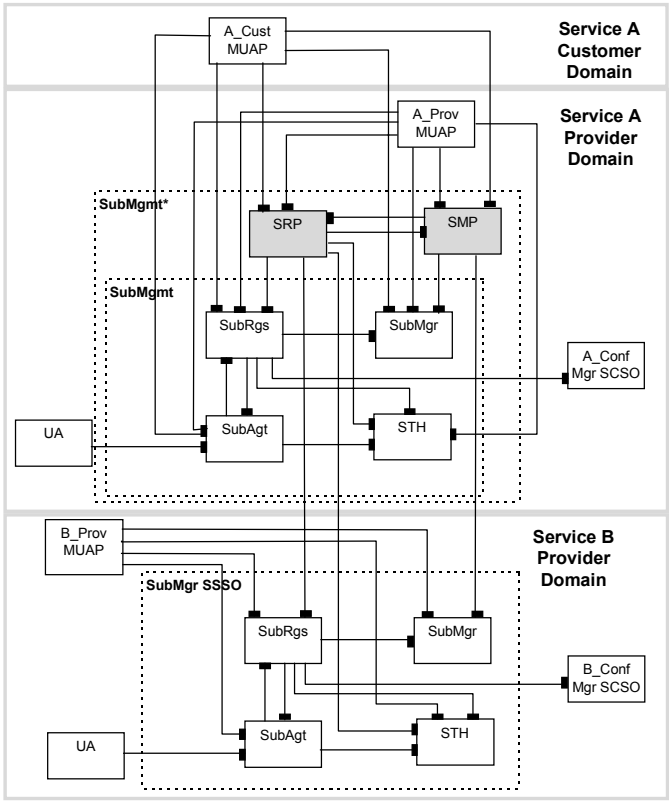

Figure 5: Extended subscription management computational object model

As had been performed with the SubMgmt object in the original TINA specification, the SubMgmt* COs were documented as a detailed block diagram identifying the specific server interfaces offered by the CO using the IDL interface names. In addition the other COs to which the CO was a client are also identified. An example of the notation used for this is given in Figure 6.

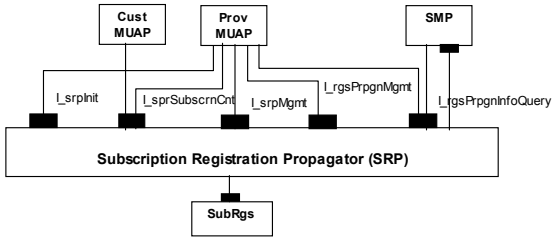

Figure 6: Sample of detailed CO block diagram

Such diagrams were accompanied with details of which IOs were handled by the CO and descriptions of the functionality provided by the different interfaces. As for the original TINA COs, sequence diagrams showing interface interactions between CO were used to develop these interface definitions and clarify which IOs are held in which COs. An example of such a sequence diagram is given in Figure 7.

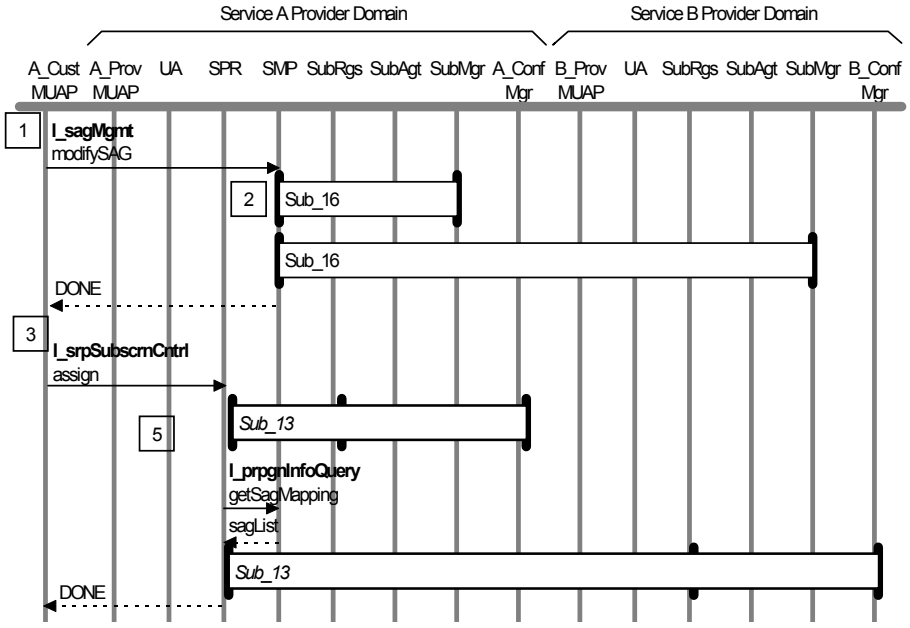

Figure 7: Example of sequence diagram showing interactions between COs

The functionality covered by these sequence diagrams was taken directly from the use case information flows used in the analysis, thus ensuring that the requirements were fully met by the design. Figure 7 shows the interactions that implement the use case information flows of Figure 3. As sequence diagrams showing interactions between multiple COs in multiple domains could easily become large and complex, a box notation was used to refer to sequences of interactions that were represented in other diagrams, .e.g. the boxes marked Sub_16 and Sub_13 in Figure 7. This form of nesting sequence diagrams also simplified the drawing of situations were sequences of interactions were repeated. The boxed numbers referred to accompanying notes that explained each significant interaction in more detail, in particular referring to their effect on IOs contained within the COs shown.

As well as proving essential in clarifying the behaviour of CO interfaces and their internal operations on IOs, the sequence diagrams were also found to be ideal for producing test documentation. Integration tests performed between components implemented by different developers were specified by defining pre-and post condition values for IOs at the beginning and end of sets of interactions represented on an sequence diagram. Values could also be provided for the parameter of interface operations performed, so that appropriate test harness software could be developed and operated. This was especially important where interactions involved a chain of several COs, and these needed to be tested individually and in small groups before finally being able to test the complete end-to-end interaction.

7 Conclusions and Further Work

The methodology has been used to develop mult-domain subscription, accounting and configuration management services. It has proved very useful both in supporting the full process development lifecycle and allowing the reuse, integration and

extension of pre-defined (standard) computational components. Experience using the methodology clearly showed that a deeper understanding that just IO and CO models of pre-existing components is required and the methodology provided techniques (use trace & interaction diagrams) to assist in the understanding within the context of the particular problem domain.

Acknowledgment
The work presented in this paper was conducted with partial funding of the European Commission under the Prospect project (AC052).

References

[Corba-95]*Object Request Broker 2.0*. Object Management Group, 1995.
[ITU-96] Text of draft recommendation G851-01, Study Group 15, ITU June 1996
[Jacob-92] *Object-oriented software engineering: a use case driven approach,* Jacobson Ivar ACM Press Wokingham Addison-Wesley 1992
[ODP-94] *Reference Model for Open Distributed Processing*, Part 1 Overview and Part 2 Foundations. ISO/IEC 10746-1 (DIS) & 10746-2 (IS) ITU-T X901 & X902. 1994
[ORDIT-93] *ORDIT* process manual, version 0.5 December 1993
[Hall-96] *'Modelling and Implementing TMN based multi domain management,* PREPARE Consortium, J Hall (ed) 1996
[Berq-96] *Succeeding in Managing Information Highways*, PRISM Consortium, Springer Verlang, Berquist, A. 1996
[Rumb-91] *Object Oriented Modelling and Design*, J Rumbaugh et al, Prentice Hall 1991.
[TINA-94] *TINA-C Service Architecture*, TINA Baseline document TB_MDC.018_1.0_94, Berndt H, Minerva R,
[TINA-93] *Computational Modelling Concepts*, TINA Baseline document TB_A2.NAT.002_3.0_93, December 1993
[TINA-95]*Information Modelling Concepts*, TINA Baseline document TB_EAC.001_1.2_94, H. Christensen, E. Colban, Version 2.0, April 1995

Stage in Design Cycle from which Viewpoint is derived	***ODP Viewpoint***
ORDIT Method to describe (in text form) - Stakeholders, Relationships,Roles, Obligations, Activities OMT Object Notation diagrams for stakeholders,relationshp Jacobson USE CASES (text based descriptions of user interactions) describe what actions are required	Enterprise Model
Use OMT Object Model diagrams e.g. class, aggregation, etc	Information Model
Use Object Classes which satisfy the USE Cases. Use traces diagrams to describe and detail the interactions between object classes. Use IDL to specify CO interfaces	Computational Model

Table 1 Mapping from elements of inter & intra domain models to ODP viewpoint models