

Implementing Integrated Management Business Processes using Reusable Components

A Report on FlowThru Project

Date: 21/1/2000

Editor: David Lewis

Contributors:

N. Agoulmine, University of Versailles	G. Pavlou, University of Surrey
K. Daenen, Alcatel Bel	C. Redmond, Trinity College Dublin
W. Donnelly, Waterford Institute of Technology	T. Richardson, STS
D. Dragan, GMD Fokus	E. Rosa, GMD Fokus
T. Gringel, GMD Fokus	L. Sorensen, UH Communications
J.Hall, GMD Fokus	C. Stathopoulos, Algo Systems
P. Hellemans, Alcatel Bel	M.Tchichholz, GMD Fokus
D. Lewis, University College London	E. Villido, University of Surrey
C. Malbon, University college Dublin	V. Wade, Tinity College Dublin

Abstract:

This report summarises the achievements and results of the FlowThru project, which ran from March 1998 to February 2000. This EU funded R&D project investigated techniques for implementing integrated telecommunications management processes from existing reusable components. It applied this to business processes identified by the TeleManagement Forum and used components based on TINA-C specifications that had been implemented in other projects from the EU's ACTS programme. This report presents the guidelines that resulted from this work, which addressed a suitable development methodology and integration technology for the problem domains. This report also reviews the design and experience gained in actually applying these guidelines with the project in three separate trial systems that implemented business processes performing integrated network and service management tasks.

1	INTRODUCTION	4
2	METHODOLOGICAL GUIDELINES	5
2.1	BACKGROUND	5
2.1.1	<i>Reusable Component Modelling Approach.....</i>	<i>6</i>
2.1.2	<i>Open Business Process Modelling Approach.....</i>	<i>8</i>
2.2	METHODOLOGICAL GUIDELINES	9
2.2.1	<i>Business Requirements Model.....</i>	<i>10</i>
2.2.2	<i>The Projection Modelling Construct.....</i>	<i>13</i>
2.3	APPLICATION OF THE GUIDELINES	17
2.3.1	<i>Evaluation of Approach.....</i>	<i>19</i>
2.4	CONCLUSIONS	19
3	INTEGRATION TECHNOLOGY GUIDELINE	21
3.1	MANAGEMENT SYSTEM DEVELOPMENT REQUIREMENTS	21
3.2	TELEMANAGEMENT FORUM TECHNOLOGY INTEGRATION MAP.....	22
3.3	MANAGEMENT COMPONENT INTEGRATION TECHNOLOGIES	24
3.3.1	<i>TMN Integration.....</i>	<i>24</i>
3.3.2	<i>Component Technology.....</i>	<i>25</i>
3.3.3	<i>TMN-Component Integration.....</i>	<i>27</i>
3.4	WORKFLOW FOR MANAGEMENT	30
3.4.1	<i>Workflow Engines</i>	<i>31</i>
3.4.2	<i>CORBA and Workflow.....</i>	<i>32</i>
3.4.3	<i>Workflow Standardisation.....</i>	<i>33</i>
3.4.4	<i>Case Study – Workflow in the Accounting Trial Business System</i>	<i>33</i>
3.5	CONCLUSION	35
4	SERVICE FULFILMENT	36
4.1	PROBLEM ANALYSIS	37
4.2	INTEGRATED SYSTEM DESIGN	39
4.2.1	<i>Components.....</i>	<i>40</i>
4.2.2	<i>Component Interactions and Modelling.....</i>	<i>42</i>
4.3	CONCLUSIONS.....	44
5	ASSURANCE SYSTEM	46
5.1	RELATED WORK.....	46
5.1.1	<i>TeleManagement Forum Contribution.....</i>	<i>47</i>
5.1.2	<i>TINA-C Contribution.....</i>	<i>47</i>
5.1.3	<i>EURESCOM Project P612 Contribution.....</i>	<i>48</i>
5.2	MULTI-DOMAIN MANAGEMENT IN THE OPEN SERVICE MARKET.....	48

5.2.1	<i>The Envisaged Business Model</i>	48
5.2.2	<i>The Technical Approach</i>	49
5.3	THE MUSIC SHOP SERVICE	51
5.3.1	<i>MusicShop User Interfaces</i>	51
5.3.2	<i>MusicShop Component Description</i>	52
5.4	THE MANAGEMENT SERVICES.....	53
5.4.1	<i>The Service Level TINA Trouble Report System (TTRS)</i>	54
5.4.2	<i>The Network Level Trouble Ticketing System (TTS)</i>	57
5.5	SCENARIOS	58
5.6	CONCLUSIONS	59
6	ACCOUNTING SYSTEM	60
6.1	THE ACCOUNTING TRIAL – BUSINESS MODEL	60
6.1.1	<i>Business System</i>	60
6.1.2	<i>Business Scenarios</i>	61
6.1.3	<i>Mapping of TM Forum Business Processes to TINA Business Model</i>	62
6.2	THE ACCOUNTING TRIAL - SYSTEM OVERVIEW	63
6.3	THE ACCOUNTING SYSTEM – COMPONENTS	65
6.4	CONCLUSIONS	66
7	ACKNOWLEDGEMENTS	67
8	REFERENCES	67

1 INTRODUCTION

The FlowThru project was a collaborative R&D effort running from 1st March 1998 to 29th February 2000, which was sponsored by the European Union under the ACTS programme.

FlowThru aimed to provide the telecommunications management industry with concrete guidance on how to build optimum solutions to specific management problems from the wide range of architectural and technological approaches currently available from bodies such as the ITU-T, ISO, TM Forum, TINA-C, OMG, ETSI and EURESCOM, among others. In particular, guidance was developed on the design issues needed to assemble reusable telecommunications management components from a number of sources to implement integrated operational support systems providing solutions to specific business process problems. Such guidance aimed to allow developers of management systems to make reasoned selections from existing solutions (standardised or otherwise) while ensuring the integrity of process information flows required to satisfy business requirements.

The project developed guidelines that provided recommendations on both an methodological approach and the use of different current and emerging technologies. The principle challenge when making such recommendations to industry, however, is demonstrating their usefulness in order to heighten their acceptance. FlowThru, therefore, undertook not just to generate these guidelines, but also to apply and validate them within the project. This was performed by the construction of three Trial Business Systems that were developed in accordance to the guidelines and which will be publicly demonstrated in order to provide a credible grounding to the dissemination of the guidelines.

To be credible in their validation of the guidelines, the Trial Business Systems must reflect real business requirements, and must be built from existing management components. In locating business requirements FlowThru has used the TeleManagement Forum's Telecoms Operations Map (TOM). This is based on a wide-ranging survey of service providers, and models their major operational concerns into a set of business processes. It is the interactions between these business processes that FlowThru used as the basis for its Trial Business System requirements. The components that were integrated together in the Trial Business Systems originate from other ACTS projects, principally Prospect, REFORM, VITAL, SUSIE and RETINA. These components are implementations of open interfaces from ITU-T, TM Forum and TINA standards and of specific research results from these projects.

The problems addressed by each of the three Trial Business Systems were taken from each of the three business process areas identified in the TM Forum's Telecom Operations Map, namely: fulfilment (i.e. service provisioning and configuration management); assurance (i.e. adherence to SLAs, fault and performance management) and accounting (i.e. service and network metering and charging).

This report consists of five paper published during the course of the project, edited to provide an overview of the approach and results of the FlowThru project. These resulting chapters address:

- The Management System Development Methodology validated in FlowThru based on [lewis99e]
- The Technology Integration Guidelines validated by the project based on [wade99]
- The Fulfilment Trial Business System based on [lewis99d]
- The Assurance Trial Business System based on [agoulmine]
- The Accounting Trial Business Systems based on [hellemans]

Further details of the FlowThru project including papers and presentations can be found at <http://www.ucl.ac.uk/research/flowthru>.

2 DEVELOPMENT METHODOLOGY GUIDELINES

Integrated service and network management for telecommunications is an area that is currently not well addressed by standards. The TMN family of standards focus largely on network and network element problems, with relatively few standards available that are applicable to the integration of service and network management. Two industrial bodies that have addressed integrated telecommunications management development in some detail at the Telecoms Management Forum (TM Forum) and the Telecommunications Information Networking Architecture Consortium (TINA-C). Both these bodies, in addition to developing some specific service and network management interfaces, have also provided some guidance on how Management Systems (MS) could be developed within their particular architectural frameworks. The TM Forum has suggested an approach to open telecommunications management interface development that draws heavily on current object-oriented analysis and design techniques and notations. In particular, the TM Forum in its specification methodology [vincent] the use of use cases and graphical modelling using the OMG's Unified Modelling Language (UML) [booch]. The use of UML is now also being adopted by the ITU in its revision of its interface specification methodology [m3020] This is done with a framework of telecoms management business processes (termed the Telecoms Operation Map or TOM) used to identify which management tasks should be analysed to develop industry interface agreements. The TINA-C development approach is heavily based on the use of the five viewpoints specified in the ITU-Ts Open Distributed Processing (ODP) recommendations [x901]. This makes heavy use of multi-interfaces distributed objects (computational objects), which, by separation into service-specific and service-independent sets, provide strong support for software reuse when implemented in CORBA.

Both of these approaches have merit, the TM Forum one for its focus on addressing the real needs of industry through analysing business processes, and TINA-C's for its support for component reuse. This chapter presents a methodology that attempts to combine the best of these approaches to developing MS. This methodology integrates the analysis of business processes and the design and implementation of systems built largely from reusable components. Maximum leverage has been made of existing software engineering techniques and tools, with UML used throughout. Aspects of business process re-engineering techniques, and their application using UML activity diagrams, have been adopted for matching system requirements to component capabilities. This matching task is supported by modelling components at high levels of abstraction using constructs based on the widely used Object Oriented Software Engineering (OOSE) technique proposed by Ivar Jacobsen [jacobsen97].

The techniques suggested for this methodology have been validated and refined through management development in several EU-funded ACTS projects [lewis95][lewis99c]. The specific techniques proposed in this paper have been trialled and validated in a current EU project, [FlowThru](#). Here three telecommunications management scenarios have been analysed, design and implemented using the methodology. Each scenario represents process interactions from one the process areas identified in the TM Forum TOM [tmf-910], i.e., fulfilment, assurance or billing. The implementation of each scenario has been constructed from existing management components, reused integrated using the proposed modelling constructs.

2.1 BACKGROUND

An analysis of the problems facing MS developers [lewis99a], suggested that this area can benefit from an approach to modelling and integrating MS that is commonly understood by MS developers, Commercial Off-The-Shelf (COTS) Software Vendors and developers of service management standards. These stakeholder types must interact in developing their products, i.e. either MS, COTS software for MS or interface standards, as indicated in Figure 2-1. Such interactions have therefore been identified as points of communication where a common development methodology would be most beneficial.

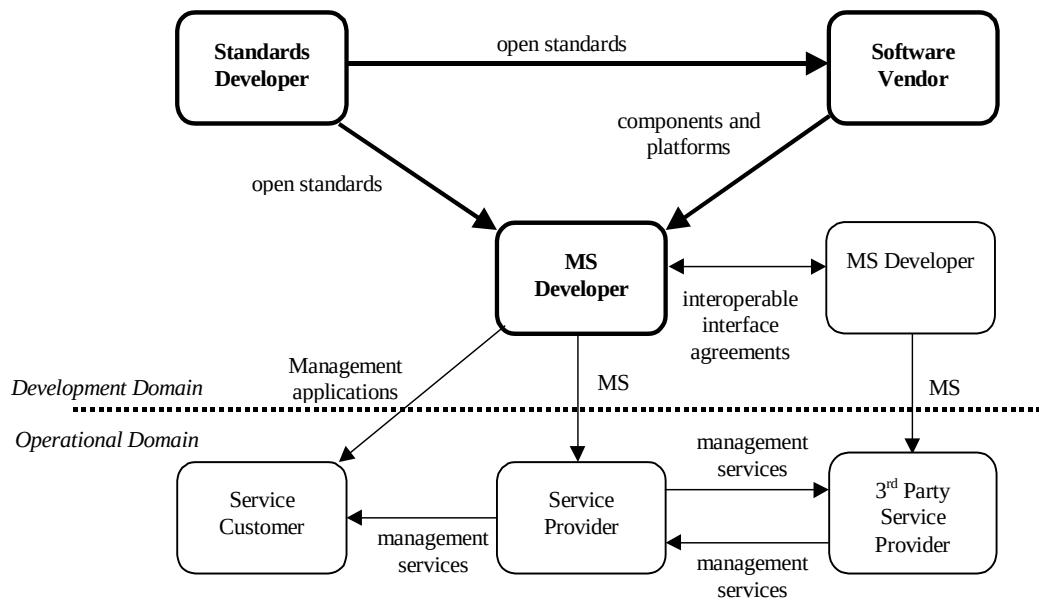
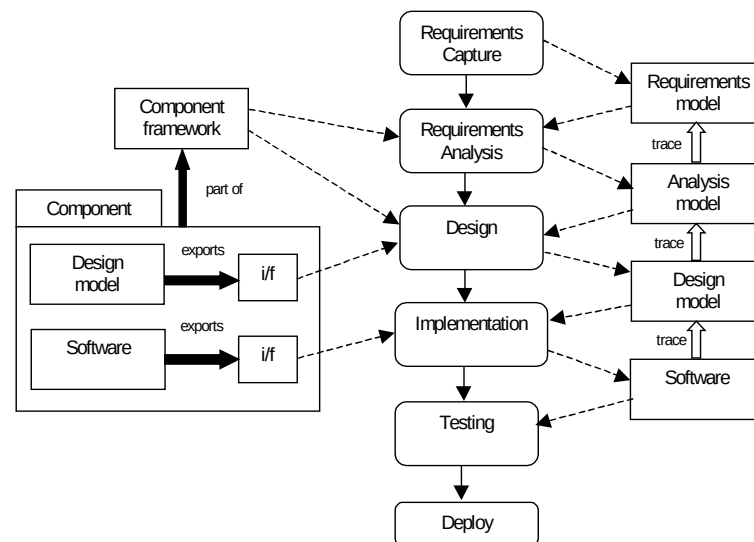


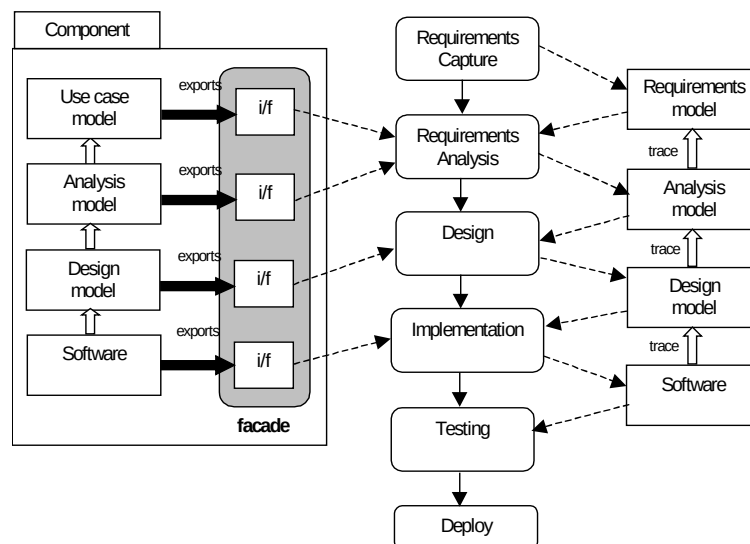
Figure 2-1: Relationships between stakeholders in MS development

2.1.1 REUSABLE COMPONENT MODELLING APPROACH

Reusable components are typically presented to system developers as sets of libraries, i.e. as a set of software modules and the definition of the individual operations they provide. The component is presented in terms of its design model and software. This may cause problems in the development of systems that reuse the component, since any changes required to accommodate the reuse of components are only likely to become apparent during the design process, therefore possibly countering aspects of the system's analysis model. Components are often part of a framework. The framework may be general, e.g. CORBA Services, or aimed at a particular problem domain, e.g. the TINA Service Architecture. In either case, the framework will provide some high level architectural and technological guidance on how components can be integrated together and how they can support the development of a system. Such frameworks are often considered at the analysis stage to ensure that the system's analysis model is structured in a way that will accommodate the inclusion of the framework's components at the design stage. This situation is depicted in Figure 2-2a. However, frameworks typically only give general guidance on the use of components. The suitability or otherwise of individual components in satisfying requirements still needs to be considered in the design activity. For MS development, such a typical component reuse situation is difficult to standardise because there is no commonly accepted framework that supports a suitably wide range of components. The methodological guidelines for component reuse presented here are motivated by the absence of such a framework. As such, the methodology presented in this paper attempts to provide guidance on how components can be specified in a more self-contained manner that is easily understood by those performing the analysis of the system. In this way, decisions about reuse can be made based on the suitability of individual components rather than on a wider assessment of the suitability of an entire framework. The approach is also aimed at supporting reuse decisions based on the architectural and functional aspects of a component rather than its implementation technology. A component's technology is treated as an orthogonal issue, with heterogeneity handled primarily through the employment of suitable gateways.



a) Conventional (design model level) component reuse



b) Analysis model level component reuse

Figure 2-2: Differing Approaches to Component Reuse

The approach is derived from that described in Jacobsen's OOSE methodology [jacobsen97]. The basis of the approach is that components are not presented just as units of design and of software within an encompassing framework. Instead, they should be packaged together with the requirement statement and analysis model of the component for presentation to potential reusers. If the modelling techniques used for the requirements capture and analysis modelling of the component are similar to those used for modelling the system in which it might be included, then it becomes much easier for an analyst to assess whether the component is suitable for use in the system. In addition the system's analysis model can directly import the analysis abstractions of the various components it reuses, easing the task of requirements analysis and ensuring, at an early stage, compatibility between components and the system requirements. This analysis model-based reuse approach is depicted in Figure 2-2b.

The presentation of a component for reuse in this way is termed a facade. A facade presents the re-user of a component with only the information needed to effectively reuse the component, while at the same time hiding from the re-user unnecessary design and implementation details about the component. The usefulness of the facade is strengthened if there is clear traceability between the different models, so that re-users can easily determine which parts are useful to them by matching

facade use cases and analysis objects to their requirements and tracing to relevant design model elements. Obviously, the construction of a facade from the internal development models of a component will be greatly eased if the same type of modelling approach was used for developing the component in the first place. Also, traceability in the facade will be greatly eased if the models of the underlying component are strongly traced.

One of the problems raised from examination of the previous case studies was that the boundaries between the different development activities were not always well defined, especially between requirements capture and analysis and between analysis and design. This meant that the level of abstraction used in the models resulting from these activities varied, making it difficult to define traceability mechanisms between the different models. Defining the structure of the different development models was therefore essential to applying useable traces between them. As use cases had already proven effective for MS and component development in the previous case studies, Jacobsen's suggestion of using use cases for the requirements model and the closely related robustness model for the analysis model was adopted. Jacobsen suggests three types of object that may be used in forming a robustness model:

- *Entity objects* that represent long-lived data in the system under analysis.
- *Boundary object* that deal with the interactions between the system and its environment.
- *Control objects* that deal with the dynamic behaviour of the system as described in use cases, and in particular the interactions between boundary and entity objects.

The object types were therefore used as the basis of an enhancement of the component façade concept, referred to here as a *Projection*.

2.1.2 OPEN BUSINESS PROCESS MODELLING APPROACH

The attempt to provide a standardised architectural framework for analysing business requirements for MS have centred either around the definition of distinct business roles and the reference points that exist between them, as in TINA, or on the definition of a common business process model as in the TM Forum's TOM. These two inputs were therefore chosen as the basis for a model that will enable business process modelling to be applied to the multi-domain problems of MS development, but in a way would support the on-going standardisation of telecommunication management functions within these two bodies. The approach taken in mapping the TOM to the TINA business model and reference points was to identify which TM Forum processes operate in which TINA Business Roles. A mapping of the TM Forum business processes onto TINA business roles, initially presented in [Lewis99a], is given in Figure 2-3.

The TOM provides a model of suitable business processes which reflect the typical operations of a service provider. This mapping, therefore, helps in the analysis a Service Provider's business processes in order to identify where existing solutions, possibly available as reusable components implementing reference point segments, can be applied. The analysis of business processes is typically performed by identifying discrete activities and the events that propagate the control of execution of a task between activities. A common representation of such a control flow is event-driven process chains, and the inclusion of activity diagrams allows UML to support a similar type of model.

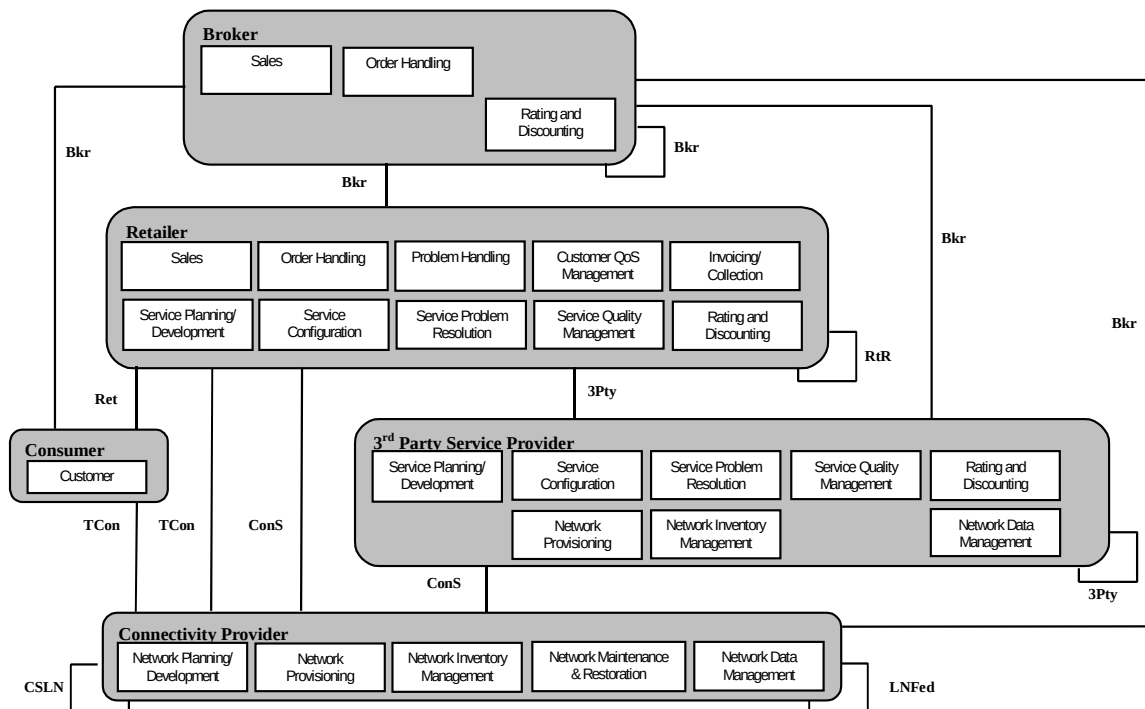


Figure 2-3: Mapping of TM Forum Business Processes onto TINA Business Roles

2.2 METHODOLOGICAL GUIDELINES

A series of case studies were conducted around the development of MS in a number of research projects. These culminated in the FlowThru Project, which developed and validated specific techniques for constructing from reusable component the MS that implemented specific process information flows. FlowThru provided evidence of the development techniques that developers found most useful in practice. Based on this evidence and the analysis of current development techniques in management system development, the following general recommendations were made:

- Use case modelling should be used for describing the external functionality of telecommunication management standards, systems and components.
- For multi-domain MS analysis, business roles and business processes should be used to supplement use cases.
- The UML notation should be used both internally for the different stakeholder's development processes and externally for exchanging models between developers involved in these processes.
- The Projection packaging construct should be used for publishing COTS software, publishing standards and for documenting internally developed reusable software.
- Where possible, an analysis and design process that uses OOSE analysis modelling should be adopted.

This section defines the notations that should be used by MS development stakeholders and the meta-models, i.e. the structure of information, to which models expressed in these notations should conform. As recommended above, the core notation used is UML, specifically the OMG's current version 1.1 [ad/97-08-03]. UML is, however, a general purpose modelling language and its designers acknowledge that it is necessary to extend and profile it to suit software development requirements of specific problem domains. This section therefore uses the UML stereotyping mechanism to propose extensions to UML for the MS development framework. This is presented in terms of stereotypes defining new modelling elements and the meta-model that defines the relationships of these elements with each other and with existing UML v1.1 elements. Class diagrams are used to show these relationships with existing UML model elements, identified for convenience by the stereotype marker

<<uml1>>. Existing UML model elements are written in double inverted commas when first introduced, and subsequently where needed to avoid ambiguity. The specific modelling constructs defined here are:

- A *Business Requirements Model* combining Business Process, Business System and Use Case Models
- A *Projection* modelling construct which is a refinement of Jacobsen's facade construct.

2.2.1 BUSINESS REQUIREMENTS MODEL

The Business Requirements Model is a stereotype of “model” that aims to support the identification of requirements in complex multi-domain situations. It consists of a *Business System Model* together with a *Use Case Model*, of the type already described in UML, and a *Business Process Model*. All three are UML “model” stereotypes. Model elements in the Business System Model are associated with model elements from the other two models as depicted in Figure 2-4.

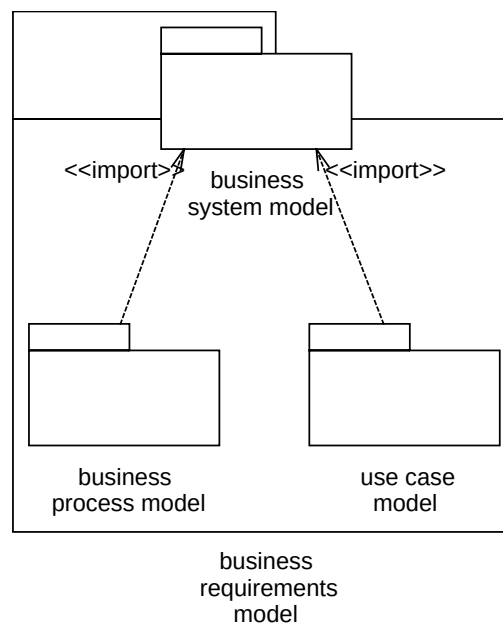


Figure 2-4: Structure of the Business Requirements Model

The contents of the Business System Model and the Business Process Model, and the association between elements in all three models are summarised as follows:

Business Process Model:

This contains the following modelling elements::

- *Business Process*: This represents a process that must be conducted to perform the business functions required of the system. It is a high level identification of an ongoing business task rather than specific identification of an activity with defined initiation and termination conditions and the flow of control between them as used in UML activity diagrams.
- *User*: This acts as a source and/or a sink of information that must be handled by one or more Business Processes. The set of users in the model defines the environment that motivates the flow of information between business processes. A User must be mapped to an actor in the use case model.
- *Information Flows*: This represents the flow of information that may pass between Business Processes or between Business Processes and Users.

Business System Model:

This contains the following modelling elements:

- *Organisational Domains*: This represents an organisation involved in the business scenario under analysis, e.g. a service provider or a customer.
- *Business Role*: This is a role played by a User within a specific Organisational Domain, e.g. service user or service administrator.
- *Responsibility*: This is a unidirectional relation between two Business Roles defining the contractual obligation one has to the other, e.g. “pay charges by due date”.
- *Management Systems*: This represents the system under analysis, which may be one of several operating within an Organisational Domain.
- *Contract*: This represents the set of functions that may exist between two Management Systems.

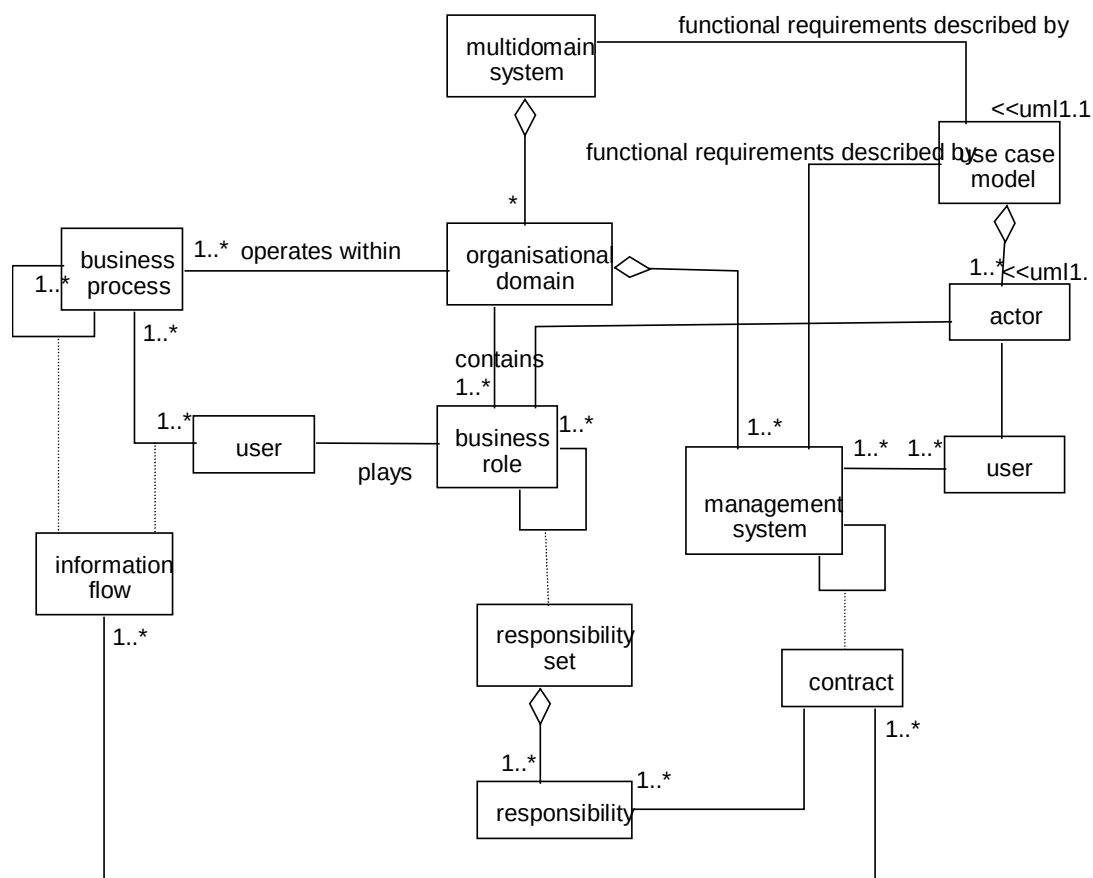


Figure 2-5: Relationships between the Elements of the Business Requirements Model

The following relationships exist between the modelling elements in a Business Requirements Model. They are also depicted in Figure 2-5:

- Business Roles should map one to one to actors in the use case model, so the descriptions a Responsibility between two Business Roles should be consistent with the corresponding actor to use case interactions and User to Process Information Flows.
- Business Processes should be wholly instantiated in within an Organisational domain.
- Individual Systems should exist wholly within one Organisational Domain

The identification of these modelling elements and their relationships enable the business requirements to be expressed in terms of requirements upon specific contracts in terms of Responsibilities and Information Flows. This is particularly useful for defining reference points

between Organisational Domains that do not involve direct interactions with actors and are therefore not addressed directly by the use case model.

The process of generating a Business Requirements Model consists of the following steps:

First, establish a multi-domain organisational model (part of the Business System Model) that identifies Organisational Domain, Business Roles with those domains and Responsibilities between them. This can be done using a UML class diagram together with corresponding textual Responsibility descriptions.

Second, establish a use case model where the actors represent the different Business Roles from the multi-domain organisational model and the use cases describe a system consisting potentially of multiple Organisational Domains.

Third, establish a Business Process Model where the users are the different multi-domain user case actors. This can be done using a component diagram to show which Business Processes interact with which Users in which Organisational Domain. Figure 2-6 shows such a diagram for the processes, roles and domains identified in Figure 2-7 for the fulfilment scenario defined for FlowThru.

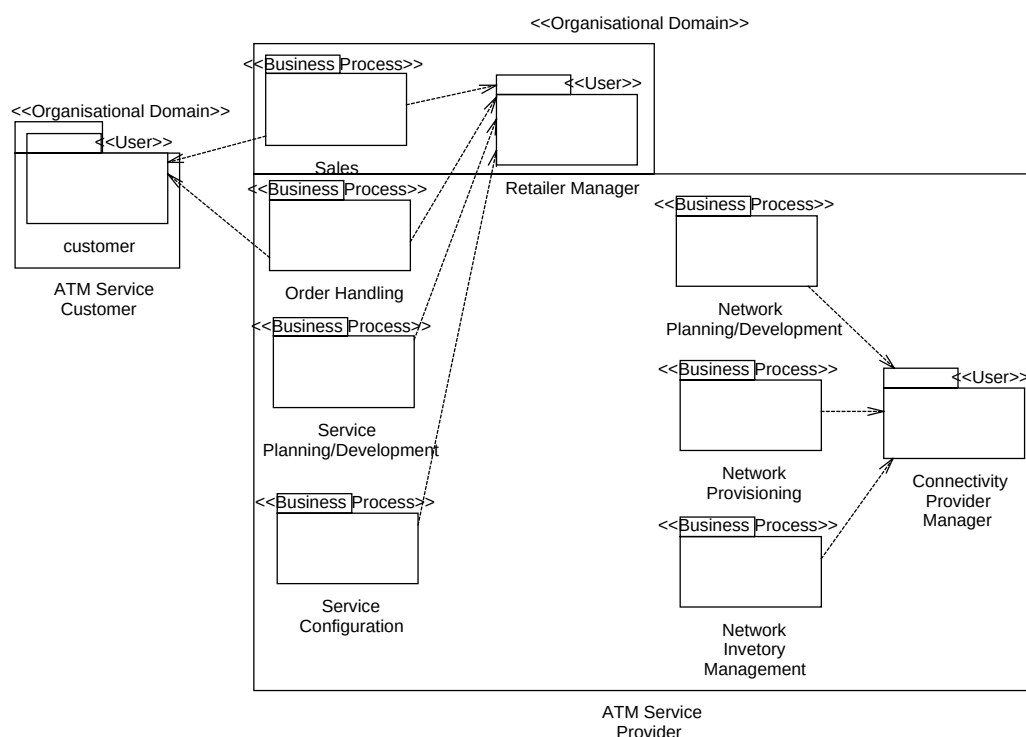


Figure 2-6: Example of static Business Process Model using a UML Component Diagram

Fourth, refine the Business Process Model to show for each multi-domain use case the information flow that must flow between Business Processes and between Business Processes and Users. This can be performed using UML sequence diagrams. An example is given in Figure 2-7 for the processes and users in Figure 2-6, for the “subscribe to ATM service” use case.

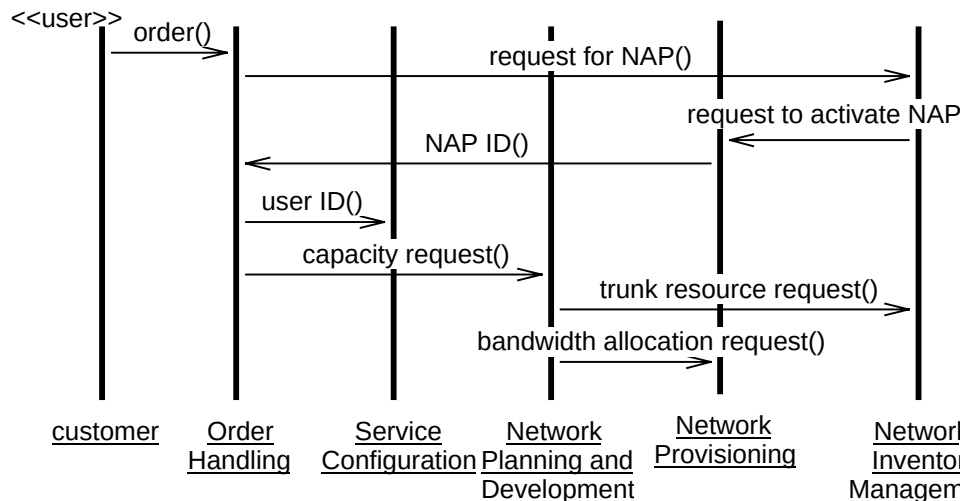


Figure 2-7: Example of Dynamic Business Process Model using a UML Sequence Diagram

Fifth, establish a Business System Model that shows the MS under analysis and the other MS and the Business Roles with which it interacts in the same or in collaborating Organisational Domains. The model should identify the Contract via which interactions are performed. This model can be represented using a UML component diagram, an example of which is give in Figure 2-8 for the MS making up the fulfilment trial business system.

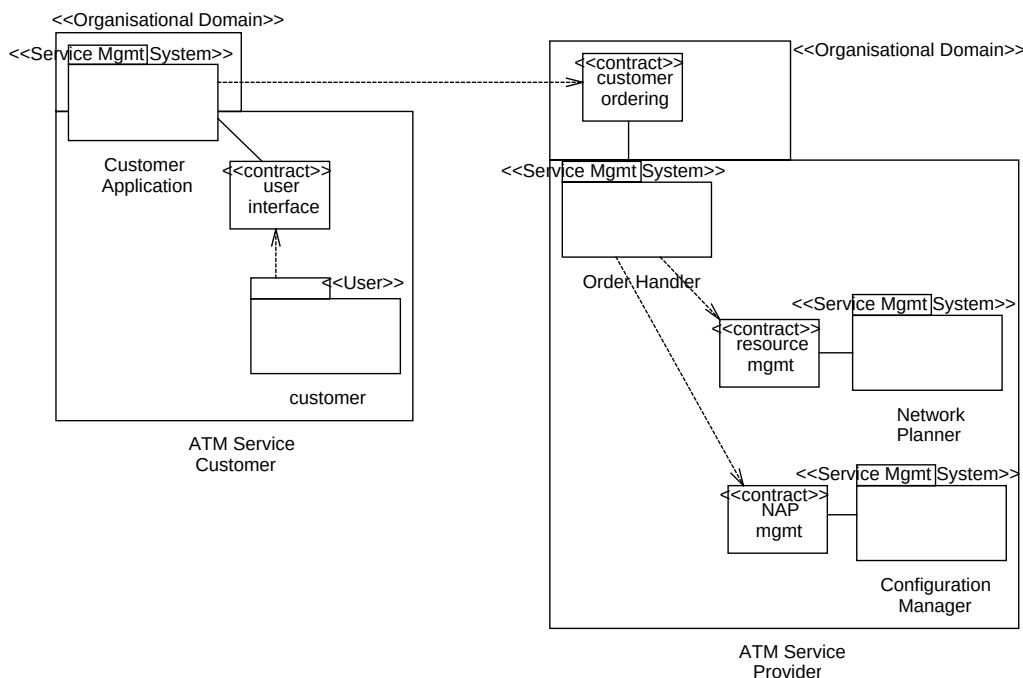


Figure 2-8: Example of MS Level Business System Model using a UML Component Diagram

Finally a use case model can be generated for the MS under analysis, with actors representing the users and other MS with which it interacts. The individual use cases involved should be triggered by inward Information Flow for a Business Process handled by this MS, as identified by the dynamic Business Process Model. As the actors represent the Users and MS that interact with the MS under analysis via Contracts, the aggregation of the all interactions between the user cases and a specific actor will define the functions required at the corresponding Contract.

2.2.2 THE PROJECTION MODELLING CONSTRUCT

A Projection allows models to be exchanged between development stakeholders whose internal models may not yet conform to the common structure used in the Projection. In such cases a mapping must exist between the meta-model used internally by the stakeholder and the Projection meta-model. As with facades, a Projection can provide a selective view of a system, revealing only the details judged by the owner of the system as needed for a specific type of user of the system, e.g. a software reuser or a specification reuser. Consequently a system may support several Projections in parallel. The Projection construct has a more defined structure than the façade construct currently defined in UML. This structure is shown in Figure 2-9.

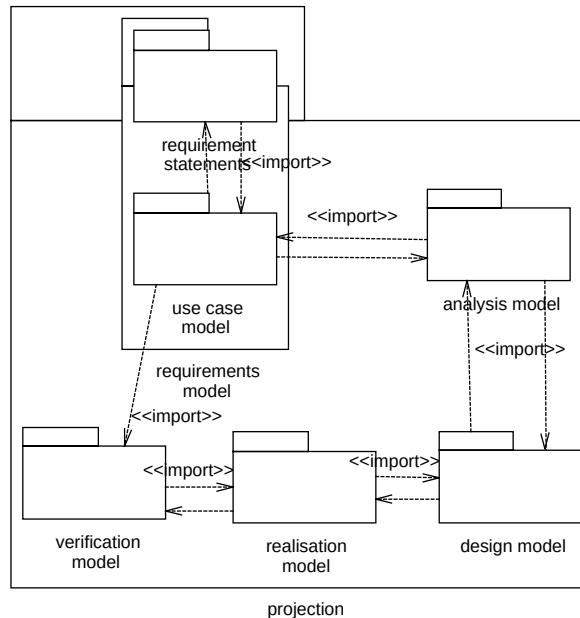


Figure 2-9: Structure of the Projection Modelling Construct

A Projection consists of the following elements, each a stereotype of “model”, and dependencies between their modelling elements:

Requirements Model:

This contains a complete requirements statement for the system concerns addressed by the Projection. It consists of two parts:

1. A set of textual requirements statements that are uniquely identifiable within the context of the Projection and which fall into one of the five requirements categories defined in the TM Forum development methodology, i.e. Structural Information, Dynamic Information, Abnormal Conditions, Expectations and Non Functional Requirements and System Administration Requirements.
2. A Use Case Model of the system being modelled addressing only the concerns relevant to the details revealed by the Projection.

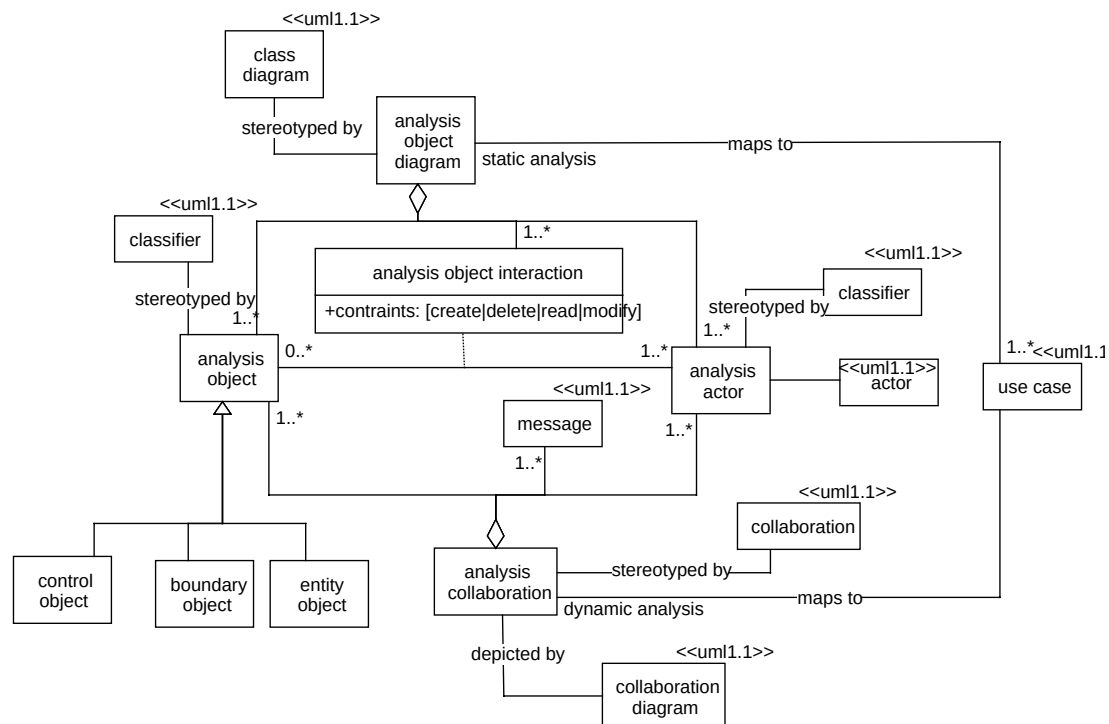


Figure 2-10: Relationship between Elements of the Projection Construct's Use Case and Analysis models

Analysis Model:

This provides an analysis of the requirements presented in the Requirements Model. It contains: classes of the analysis object types defined by Jacobsen, i.e. the control, boundary and entity object types; actors that place requirements on the system and identification of interactions between analysis objects and between analysis objects and actors. Interactions may be classified by the one or more of the following types: create, delete, read, and modify. The static view of the Analysis Model may be represented in an analysis object diagram, which is a class diagram that supports the analysis object stereotypes. In parallel the Analysis Model should contain collaboration diagrams which show the dynamic behaviour of the classes in the analysis object diagram in terms of messages that pass between instances of them. The structure of the Analysis Model is driven by the Requirements Model. Each use case in the latter should be reflected by at least one analysis object diagram and one analysis collaboration in the former. In addition the analysis actors should have a one to one mapping to the use case model actors in the Requirements Model. These model elements and their relationships to each other and to elements in the use case model are depicted in Figure 2-10. An example of an Analysis object Diagram is given in Figure 2-11.

- Analysis collaboration diagrams to design interaction diagrams

Realisation Model:

This defines the physical realisation of the design model in terms that support its integration into other models by the Projection's user, including the constraints of any configuration that can be performed by the user. The Projection definition does not prescribe the notation for this model, though UML deployment and component diagrams may both be useful here.

Verification Model:

This defines the information and procedures needed by the user of the capabilities of the system defined by the Projection in order to ensure that it is operating consistently with its requirements in the environment in which the user has placed it. The Projection definition does not prescribe the notation for this model.

2.3 APPLICATION OF THE GUIDELINES

The Projection modelling approach was applied within FlowThru to the integration of components in three separate scenarios demonstrating different business process areas from the TM Forum's Telecoms Operations Map. The components from which these systems are constructed were from different EU funded projects, and as such they were originally developed using a variety of notations. To validate the reusable component modelling approach described above the specifications of these existing components had to be recast as Projections. The core aim was to be able to export the Projection in a form that presented each of its constituent models and the traces between them. An HTML-based approach was therefore taken in order to allow traces to be implemented at hyper-links. The publication of the facade on the web is recounted in more detail in [lewis99b] and a sample model can be viewed online at: <http://www.cs.ucl.ac.uk/research/flowthru/models/>.

In mapping use cases to an analysis model for a facade, entity objects were derived from noun phrases that occur in one or more use cases. Boundary objects were identified by any interactions between the users and the system. Control objects were initially allocated per use case, and then consolidated as functional commonalities are identified between use cases. Interaction diagrams were required at this stage to detail the lifecycle of objects and dynamic aspects of the relationships between them. Refining the analysis model into the design model for a Projection took into account design level issues, such as scalability, load balancing, platform dependencies, database schemes etc. Strong tracing between objects in the analysis model and the design model had to be maintained during this process.

Though it is preferable to generate a Projection from a component model of the same structure, it is not a requirement for the component to have been originally developed and documented using an OOSE-like process. Indeed, part of the benefit of the use of Projections is that they can hide the internal model if necessary. In FlowThru the latter situation was demonstrated by developing Projections for components that had been developed and documented using ODP viewpoints. The following mappings were used to help reverse engineering the ODP models into the Projection format:

- Roles in the enterprise viewpoint onto actors in the use case modelled in the Projection.
- Computational objects from the computational viewpoint onto control objects in the Analysis Model.
- Computational object interfaces, individually or grouped, onto boundary objects in the Analysis Model.
- Information object onto entity objects in the Projection's Analysis Model.
- Where sequence diagrams had been used to clarify the interactions between computational and information objects, then these formed the basis for reverse engineering use cases, otherwise use cases were based on the component's requirements.

```

graph TD
    subgraph Customer_Interface_Management [Customer Interface Management]
        Start(( )) --> PlaceOrder[Place Order]
        OrderFailed(( )) --> OrderFailedLabel[Order Failed]
        OrderComplete(( )) --> OrderCompleteLabel[Order Complete]
    end

    subgraph Order_Handling [Order Handling]
        AcceptOrder[Accept Order for ATM Service] --> CreditCheck{ }
        CreditCheck -- "[Credit check fail]" --> OrderFailed
        CreditCheck -- "[Credit check OK]" --> CreateAccount[Create Customer Account]
    end

    subgraph Service_Configuration [Service Configuration]
        DetermineCoS[Determine available CoS] --> AuthorizeUsers[Authorise users to use service]
    end

    subgraph Network_Planning_and_Development [Network Planning and Development]
        UpdatePredictions[Update network usage predictions] --> VPCapacity{ }
        VPCapacity -- "[VP network capacity inadequate]" --> UpdateTopology[Update network topology]
        UpdateTopology --> PhysicalCapacity{ }
        PhysicalCapacity -- "[physical network capacity inadequate]" --> ReconfigureVP[Reconfigure VP network]
        PhysicalCapacity -- "[physical network capacity adequate]" --> EndMain[ ]
    end

    subgraph Network_Provisioning [Network Provisioning]
        ActivateNAP[Activate Network Access Points] --> EndMain
    end

    subgraph Network_Inventory_Management [Network Inventory Management]
        LocateNAP[Locate NAP] --> NAPCheck{ }
        NAPCheck -- "[network access point in place]" --> ActivateNAP
        NAPCheck -- "[network access point not in place]" --> InstallHardware[Install NE hardware and lines to NAP]
        InstallHardware --> ReconfigureVP
    end

    PlaceOrder --> AcceptOrder
    CreateAccount --> DetermineLocation[Determine Location of Customer Site(s)]
    DetermineLocation --> DetermineCoSAtSite[Determine CoS to be used at each site]
    DetermineCoSAtSite --> DetermineUsers[Determine the users to use each CoS at each site]
    DetermineUsers --> ActivateCoS[Activate CoS user groups]
    ActivateCoS --> AuthorizeUsers
    AuthorizeUsers --> UpdatePredictions
    UpdatePredictions --> VPCapacity
    VPCapacity --> UpdateTopology
    UpdateTopology --> PhysicalCapacity
    PhysicalCapacity --> ReconfigureVP
    PhysicalCapacity --> EndMain
    ReconfigureVP --> EndMain
    EndMain --> OrderComplete
  
```

The diagram illustrates the process flow for ATM service activation, organized into six swimlanes:

- Customer Interface Management:** Starts with 'Place Order'. If the order fails, it leads to 'Order Failed'. The process ends with 'Order Complete'.
- Order Handling:** 'Accept Order for ATM Service' leads to a decision point. If the credit check fails, it leads to 'Order Failed'. If successful, it leads to 'Create Customer Account'.
- Service Configuration:** 'Determine available CoS' leads to 'Authorise users to use service'.
- Network Planning and Development:** 'Update network usage predictions' leads to a decision point for 'VP network capacity'. If inadequate, it leads to 'Update network topology', which then leads to a decision point for 'physical network capacity'. If inadequate, it leads to 'Reconfigure VP network'. If adequate, it leads to the end of the process.
- Network Provisioning:** 'Activate Network Access Points' leads to the end of the process.
- Network Inventory Management:** 'Locate NAP' leads to a decision point. If the network access point is in place, it leads to 'Activate Network Access Points'. If not, it leads to 'Install NE hardware and lines to NAP', which then leads to 'Reconfigure VP network'.

Figure 2-12: UML Activity Diagram for the Subscribe to ATM Service Use Case

The overall system analysis model was then defined in terms of the interactions between analysis model elements from the imported component Projections, and in particular the bindings between their boundary objects. The overall system design model consisted of the modified IDL for each component and detailed sequence diagrams showing inter-component interactions that enacted the system's use cases.

2.3.1 EVALUATION OF APPROACH

The experiences of this case study relate the generation of the Projection for pre-existing components and to how this may be integrated into the development of an MS in a way that ensures the satisfaction of business process requirements and ready integration with existing open standards. The Projection presents the component model at levels of abstraction, i.e. as Use Case and Analysis Models, which facilitate the components integration into the MS. It was found that the division in the Projection between the Analysis Model and the Design Model provided a good basis for delineating between the exposure of internal details of a component needed for its features and capabilities to be understood. At the same time the Projection hid all detailed design issues except those relating to the interfaces via which it is re-used.

Publishing the Projection in HTML with a structure suitable for re-users to make best use of the traces between elements in the different models was found to be relatively simple with Paradigm Plus. However, the fact that the same teams were involved in Projection generation and the design of the systems that reused the components meant that the effectiveness of the Projection construct in component selection and comprehension could not be assessed. In addition, therefore, a questionnaire was used to get a more structure view of the developers' experiences, against which the above observations may be compared. Fourteen completed questionnaires were returned. The responses mostly were in line with the above observations and the findings of the previous case studies. The business process modelling was rated highly for both system and component development. The Projections were also rated highly by system designers, though the rating for the Projection's Use Case Model, indicated that this was not clearly related to the Analysis and Design Model in all cases.

The responses to some open questions included in the questionnaire were as follows:

- In response to the question "Do you think the use of component Projections resulted in a better-designed component?" six replied yes, three no and four did not express an opinion. Comments were made that in most cases the component design was fairly mature, so the Projection generation was purely a documentation exercise.
- In response to the question "Do you think the design of the trial business systems was made easier or not by the use of the component Projection structure?" seven replied yes, two no and five did not express an opinion.
- In response to the question "Do you think the modification of components for reuse in the trial business system was made easier or not by the use of Projections?" five replied yes, none no and nine did not express an opinion.

2.4 CONCLUSIONS

This work has provided experience in the application of UML and commercially available CASE tools for telecommunications management process analysis and component reuse. It is hoped these results will prove useful to industry practitioners faced with building telecommunication management systems. The methodology and experiences presented here are currently being disseminated via the ACTS program as well as providing input to the TM Forum, TINA and ITU workgroups.

The business process to reference point mapping model was found useful in bringing together the business process model approach to establishing management requirements from the TM Forum and the component-based reference points based defined in TINA. This assisted the understanding of developers with backgrounds in TINA who needed to use their systems to satisfy business process requirements. This mapping has been presented to the TM Forum and TINA-C by the author. It has

been received with interest by both and at the time of writing is forming the basis of negotiation on a possible liaison agreement between the two bodies. Such a mapping, combined with the representation of reference points as the Projections of their constituent components, points the way to the integration of existing component and interface specifications with business process driven MS development. It also provides a path to expanding the scope of reference points based on existing business process analyses.

3 INTEGRATION TECHNOLOGY GUIDELINE

The liberalisation of telecoms markets across Europe and the world has exposed service providers to a high level of competition. This competition is forcing them to reduce costs, improve customer service and rapidly introduce new services. One key way in which these pressures can be addressed is through the improved integration of the many software systems operated by a service provider. This includes amongst others, the integration of different operation support systems.

Component based reuse is seen as an increasingly important software development aid, both within the telecoms industry and in the wider IT community. Building systems from components that interact through well defined interfaces offer a route to reusing software across projects within a telecom system developer and to integrating commodity third party software into the system. Both of these offer development cost savings and improvements in reliability and maintainability. Emerging standards such as Enterprise JavaBeans and CORBA Components are encouraging the development of platforms that directly support component integration. This is prompting the telecoms industry to move towards the widespread adoption of component-based architectures. For example BT and MCI/Worldcom have already published architectural and requirement documents that encourage the migration of their OSS architectures to component-based platforms. This movement is now also being supported within the TeleManagement Forum. Bellcore has long established its Information Networking Architecture (OSCA/INA) with a similar aim. This has been evolved and validated by the TINA-Consortium, resulting in the standardisation of ODL in the ITU where it is likely to become the basis for the further standardisation of IN Capability Sets. However, many existing architectures, such as TMN, do not directly support component-based systems, and not all the notations and tools currently used in telecoms can fully represent components, e.g. GDMO, UML. The alignment of notations and tools with the modeling constructs and development activities associated with component-based software development must therefore be addressed.

This chapter examines the current status of component integration technology and assesses it against the requirements of OSS developers. The technologies addressed are:

- CORBA Components: based on the joint revised submission to the corresponding RFP, which is currently under consideration by the OMG.
- Enterprise Java Beans.
- Workflow technologies based on both OMG and WfMC specifications.
- Contemporary TMN technology, specifically gateways between CMIP and other distributed technologies such as CORBA and SNMP, and the C++ TMN API.

Other relevant technologies such as DCOM, Java Management Extension, WBEM and Distributed Databases and Transaction Processing have not been covered in this chapter but are worthy of further examination in this area. The chapter starts by clarifying the assumptions made about requirements for technology integration in the communications management context and outlining the TM Forums approach to this as presented in its Technology Integration Map. The chapter then discussed the capabilities of current TMN and component related technologies and discussed how they may be integrated in future. Next the chapter discussed the application of workflow to management and provides some detailed of its application in FlowThru through a case study.

3.1 MANAGEMENT SYSTEM DEVELOPMENT REQUIREMENTS

The telecoms industry needs to build solutions to specific management problems from the wide range of architectural and technological approaches e.g. ITU-T, ISO, TM Forum, TINA-C, OMG, ETSI and EURESCOM, among others. In particular, practitioners need to create operational support system solutions from reusable telecommunications management components that may be drawn from multiple origins. Developers of management systems need to be able to make reasoned selections

from existing solutions (standardised or otherwise) while ensuring the integrity of the information flows required to satisfy business requirements. Service providers & system developers need an *open market* for reusable management components, which allow the building software systems from reusable, multi-threaded components are reduced. Key requirements include

- The ability to integrate with legacy systems in a cost effective way.
- The ability for components to interoperate even if they have been implemented using different distribution or programming technologies.
- The ability for components to interoperate even when they offer interfaces that have been defined in different languages, e.g. IDL, ODL, GDMO or SMI.
- An integration mechanism that minimises the knowledge needed of other potential interoperating components when a new component is developed.
- The need for an integration mechanism that clearly supports the needs of specific business processes in a clearly observable manner.

3.2 TELEMAGEMENT FORUM TECHNOLOGY INTEGRATION MAP

The TeleManagement Forum undertook an examination of several different integration technologies for telecoms management called the Technology Integration Map (TIM) [tmf-910]. The overall structure of the TIM is represented in Figure 3-1.

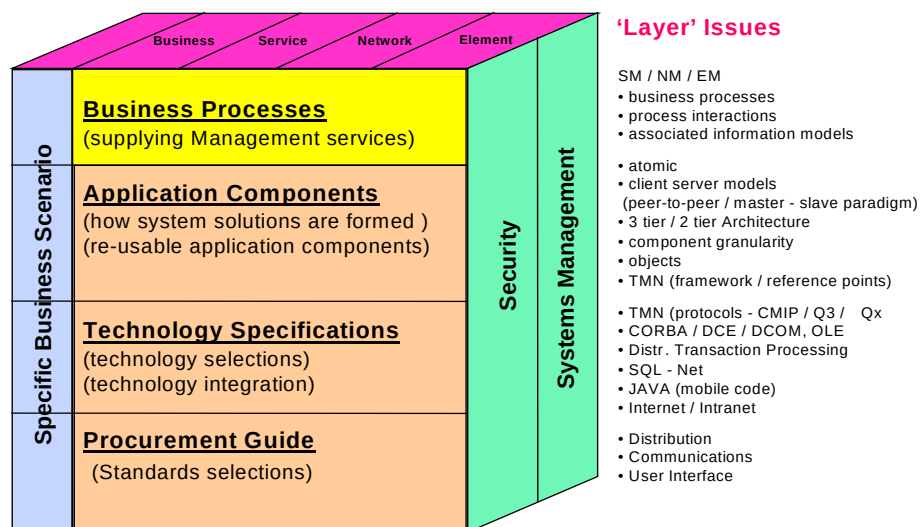


Figure 3-1: TM Forum Technology Integration Map Structure

The TIM suggests a possible mapping of technologies and associated application components to particular telecom business needs. The needs of individual service providers are driven by specific business scenarios. The intention of the TM Forum is to use this mapping for several of its activities, which include:

- Telecoms Operations Map which gives a provider independent view of key business processes.

- Application Component activity, which is at an early stage, but which aims to provide the basis for defining reusable management components.
- Technology Direction specifying the relative merits of different management-related technologies such as CMIP, CORBA, DCE, DCOM, SQL, Java etc.
- Procurement Guide, which would aim to provide guidance to industry practitioners when making purchasing, decisions for components and platforms of management systems.

Currently the TIM focuses on the comparison of the technology specification and does not provide any real guidance on how this might map to *specific business process areas*. It has however recommended where different technologies might be applied to different types of system role. The system type and technology choices are summarised below in Table 1.

Interface	Type	Technology
1	Customer/Operational Staff access	Web browser / JAVA
2	Business process interaction / backbone distribution	CORBA (+ Workflow?)
3	Business process control of network resources	CMIP/GDMO SNMP/MIBs
4	Business process access to operational data (not discussed)	SQL, SQL-Net, ODBC, Data Distribution?

Table 1: Technology selections

This categorisation led to the identification in the TIM of several technology integration points, i.e. points where effective interworking were required between different technologies in different system roles that needed to interact as part of an integrated management business process. These integration points (IPs) are identified and numbered (1-5) in Figure 3-2.

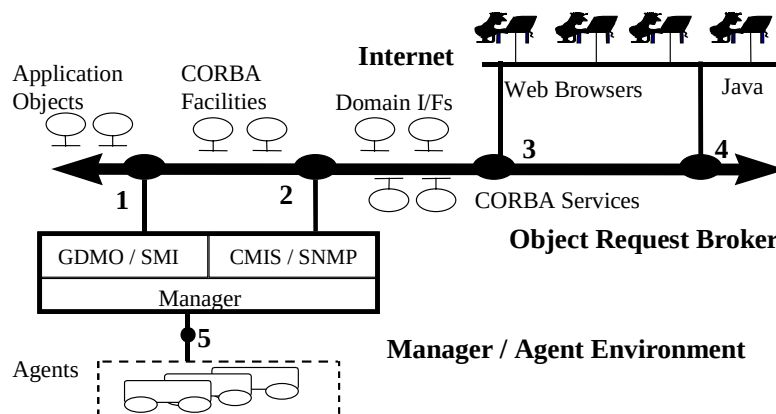


Figure 3-2: Technology Integration Points

Essentially, this figure indicates that from the technologies selected, three technology areas will need to be integrated. These are:

- Internet/Web based services.
- Object Request Broker (CORBA) based services.
- Telecom based Manager/Agent services (i.e. CMIP/GDMO and SNMP/SMI).

In order to provide adequate points of integration between these areas of technology, the five IPs have are identified below:

IP1: Provides mapping of objects defined in CORBA/IDL to managed objects defined in GDMO or SMI.

IP2: Provides mapping of appropriate CORBA Services to CMIS and SNMP services.

IP3: Provides a mapping of Web Browser technology access to CORBA objects (for situations where this may be needed as an addition to/replacement of Browser access to a database).

IP4: Provides a mapping between Java based objects and CORBA objects.

IP5: Provides a high level convenient programming interface for the rapid development of TMN based manager/agent interactions. It also provides a convenient point of integration if it is necessary to separate out the two sides of the manager/agent interface from the point of view of technology selection. For example, allowing the manager role to perhaps be supported in a Web-based environment, but giving a good point of integration with a TMN based agent.

While some of these reference points have been addressed by existing gateway standards, e.g. the JIDM CORBA-CMIP specifications [jldm95][jldm97], key technologies such as Java, CORBA and Workflow have been rapidly evolving towards direct support for reusable components in a way not currently addressed by the TIM. The following sections review some of the relevant interworking technologies and the component oriented developments that have emerged recently.

3.3 MANAGEMENT COMPONENT INTEGRATION TECHNOLOGIES

This section discusses existing approach to integrating TMN/CMIP with other technologies and the potential of component-oriented technologies. It then discusses potential routes for the integration and co-existence of TMN and component technologies.

3.3.1 TMN INTEGRATION

In the context of this paper TMN means the OSI management framework as standardised in the ITU-T X.700 series of recommendations. The definition of the relationship between managers and agents is modelled in GDMO/ASN.1, and the actual exchange of management information between systems is via the CMIP protocol.

The interoperability between TMN based systems is good. Although CMIP is fairly complex, the experience shows that it is possible to configure systems to interoperate at the protocol level (as opposed to some CORBA based systems)[bjerring]. TMN based management systems have reasonable potential in many aspects. If this is true then why is TMN based systems then not deployed more widely? Probably because early TMN systems got the reputation of being expensive and difficult to implement. However, this does not imply that TMN has no future. The difficulties in implementation are reducing, due to new development toolkits. It can be argued that because of TMN's maturity, its technology has reached a stage where development of a TMN based management system is less demanding in terms of implementation effort compared to any other technology. This is largely due to built-in generic and standardised features that often are needed when building management systems. A modern TMN development platform offers event handling, distribution, naming, persistency, logging, design tools, debugging, test facilities and a number of implemented standardised information models even before the implementation work is started.

The TeleManagement Forum has, with the TMN/C++ Application Programming Interface (API) product set, defined a standard programming interface for use with these international standards for network and systems management. The objective of the TMN/C++ API is to provide a straightforward mechanism to write portable and interoperable management application programs using C++.

To support this objective the TMN API has been designed to:

- Allow widespread portability, with a minimum amount of effort and code modification.
- Be simple, intuitive, and easy to use for experienced programmers.
- Be sufficient to support anticipated applications.

- Be flexible to allow for different implementation strategies, application designs, and operating environments.
- Be useable by a wide variety of application problem domains (i.e. it should be neutral with respect to the actual GDMO content.)

Much of the use of such gateway technology in Network Management is expected at the element management or lower layers of network management, particularly where integration with legacy CMIP based element managers or agents are required.

3.3.1.1 CORBA / TMN INTERWORKING

The Joint Interdomain Management (JIDM) task force was set up by TM Forum and X/Open to provide specifications of how CORBA and OSI System Management (SM) could co-exist in the same management environment. Their specifications involve translating information exchange between TMN- and CORBA-based components. There are two streams in this work – mapping between GDMO/ASN.1 and CORBA IDL, and translation between CMIP/SNMP and CORBA operations invocations. The former consists of facilitating information models in either domain to be viewed by its counterpart. The latter is concerned with component interactions in each domain – CORBA and TMN – providing the capability for components in one domain to act like components in its counterpart's domain. Many descriptions of the work and results of JIDM are available e.g. [jidm]

It also should be noted that, at the time this version of the Technology Direction Statement was produced, the OMG had partially completed the process of adopting a technology for CORBA/TMN Interworking. It also should be noted that the principal submissions to this process were centred on the JIDM and TMN C++ API specifications outlined above. Hence the outcome of the OMG's selection process could well result in a CORBA/TMN interworking specification which is agreed by TM Forum/Open Group and OMG. This would result in greater clarity in the defined roles for both CORBA and CMIP/GDMO, and would see both of these technologies being more fully exploited in future telecom systems.

An implementation of the JIDM CORBA/CMIP gateway was using the assurance and accounting Trial Business systems in FlowThru (see chapters 5 and 6 respectively) while the TMN C++ API was also use in the assurance system for customer integration of CORBA and CMIP based subsystems.

3.3.2 COMPONENT TECHNOLOGY

The advent of component-based development promises to reduce system and application development to a process of wiring together software “integrated circuit” [cox]. To achieve this a component must be able to offer both structural and promotional interfaces. Promotional interfaces allow system developers and third parties to inspect a component in order to determine its usefulness. Structural interfaces allow application developers, integrators and frameworks the means with which they can establish communication.

One key benefit that component technologies promise is software reuse. But why should this be any more attractive than with existing OO development methodologies? The answer lies on the shift of the software development process from a craft-based activity (i.e. building from scratch) to an industrial process. Here, component-based systems reduce the application and system development process to one more familiar in traditional industrial manufacturing. Here, large cost savings are achieved by using standard procedures to assemble complex products from sets of well designed, pre-fabricated parts without a loss of build quality. This is sometimes referred to as the “industrialisation” of the software development process. Such a shift requires a big change in software developer mentality. Existing component-based development techniques are related to well-known OO development practices (such as, business analysis and use case modelling). These will not disappear, but, the deployment of component built systems will shift the emphasis from fabrication to integration, and to this end a component's self-description, or “introspection”, will become as important as its ability to implement some set task.

A number of emerging heavyweight technologies are addressing component design and development including Microsoft's Active X Controls/DCOM, Javasoft's Enterprise JavaBeans [matena] and the OMG's CORBA Component Model [orbos/99-02-05]. The following sections outline the last two of these technologies.

3.3.2.1 CORBA COMPONENTS

As a component can be viewed from several different standpoints depending on how the component is being used and who is viewing the component the CORBA component specification defines a number of separate models through which a component must be described.

The Abstract Model describes a component from a top-level, design perspective. Also known as the Meta-Model, it contains information on the objects, links, and data values of the component, as well as details on how this information should be viewed using Document Type Definitions (DTDs) based on the XML Metadata Interchange (XMI).

At a lower level, the specification defines five further models that describe how the component will both function and integrate. These are the Component, Persistency, Packaging, Deployment and Container models.

The Component Model expresses the component as a type. The type definition provides both compile-time and run-time information on its external interfaces. This includes new IDL syntax to provide:

- Unique component identification
- Identification of interfaces that the components both provides and uses
- Details on the events that a component both emits and consumes
- Navigation interfaces that allow the above to be examined
- Interfaces that allow runtime attribute and property configuration
- Interfaces for managing multiple component instances

The new syntax is essential to allow an IDL compiler to turn a component definition into an executable implementation in a target language.

The Persistency model defines how a component's state may survive its physical representation in memory, extending a component's lifecycle.

The Packaging Model provides details to enable a component to be assembled with other components within the scope of a distributed application.

The Deployment Model defines how components are configured and readied at run-time. The main aim here is to install the component into a physical computing environment.

Finally, the Container Model deals with a component's runtime environment. In general, components execute in containers and must include details of the system services they require in order to operate effectively. Its container, itself a component, provides access to these services and controls external access to potentially multiple instantiations of the component.

3.3.2.2 ENTERPRISE JAVA BEANS

SUN Microsystems's Enterprise JavaBeans (EJB) is designed to run within the Java Runtime Environment and therefore requires a suitably engineered Java Virtual Machine. In order that an EJB may be queried by frameworks and other EJBs and connected to those that wish to cooperate, it must provide a number of standard, external entry points. The specification defines a set of naming conventions, "design patterns", that enable a component's entry points to be identified and which builds on the introspection inherent in all Java classes. These entry points include: a set of methods or interfaces via which the component can communicate; run-time configurable properties that promote customisation; and events with which a component notifies interested EJBs of changes in state.

Crucially, a JavaBean executes within a container, in itself a JavaBean. A container provides an environment within which one or more beans run and acts as a bean coordinator, controlling access to the bean and creating new instances on request. As the EJB spec, "A container is where an enterprise Bean object lives, just as a record lives in a database, and a file or directory lives in a file system". This enables beans to be managed in a locally scalable way. It can also control bean access to supporting services and scarce resources, and provides a secure environment for each bean to exist within the wider distributed environment.

To facilitate this an EJB exists in one of two different guises, as either a *session* or *entity* bean. Session EJBs are transient and have a limited lifecycle. They are created by clients and usually only exist for the duration of a client/server session. Once the bean is destroyed its data is lost, which, crucially applies in the event of a system crash. The bean must manage any persistency that might exist. On the other hand, an entity EJB is persistent and typically lives as long as the data that describes it. As the data may reside in a database this could be for a much longer time than the components initial representation in system memory. Furthermore, EJBs can survive a server crash and are designed to be transactional.

To complement the EJB distributed component model SUN has defined a comprehensive set of APIs. There are 8 in total and consist of the: Java Naming and Directory Services (JNDI), Remote Method Invocation (RMI), Java Interface Definition Language (Java IDL), Servlets and Java Server Pages (JSP), Java Messaging Service (JMS), Java Transaction API (JTA), Java Transaction Service (JTS) and Java Database Connectivity (JDBC). Together these form the Java Platform for the Enterprise (JPE) and this is intended to enable any Java application to access an enterprise-class infrastructure, often termed middleware.

3.3.2.3 COMPARING CORBA COMPONENTS AND EJBs

CORBA Component and EJB specification are alternative component technologies. The EJB has already presented one of the first working implementations, backed by the Java market. The CORBA Components present a more general modelling framework, advocating the use of CORBA as the underlying technology. The forthcoming CORBA Components standard will constitute an integral part of the third version of the CORBA suite. It will contain both the models and the meta-modelling constructs required in order to deploy a full-scale component technology. Additionally, it will cover the mapping from Enterprise JavaBeans (EJB) components to CORBA components and vice versa. In fact, one of the mandatory requirements set by OMG, in its Request for Proposals for a CORBA Component Model [orbos/97-06-12], was the compatibility with JavaBeans, while the latest proposal has considered compatibility with Enterprise JavaBeans. It is expected that an Enterprise JavaBean may be inserted into a CORBA container, while a CORBA Component, written in Java, may run in an EJB Container. An EJB interface can be mapped to its equivalent CORBA IDL and an EJB container can use either RMI or IIOP as its distributed communication protocol. More specifically, an EJB may be inserted into a CORBA container and a CORBA component, written in Java, may run in an EJB container. Furthermore, certain JPE services have equivalent CORBA service counterparts from which they borrow much of their design. JTS is an implementation of the CORBA Object Transaction Service, and the JNDI extends the CORBA Naming Service to give general directory access support that includes the Lightweight Directory Access Protocol (LDAP). The CORBA Component model is currently being aligned to the EJB Model so that JAVA based components are interoperable with (non JAVA) based CORBA components.

3.3.3 TMN-COMPONENT INTEGRATION

This section outlines the possible directions that could be taken in integrating existing TMN technology based on CMIP and the emerging component technology represented by CORBA Components and Enterprise Java Beans.

There exist certain business drivers that will help the persistence of CMIP-based TMN namely:

- The existing body of NE MIBs, which are well understood GDMO models, that have been through a standardisation process and are in many cases already deployed in commercial systems using CMIP.
- CMIP offer efficient and timely management of large number of MOs, e.g. agents with millions of object instances are not uncommon.

Though perceived as complex and expensive, CMIP solution are now fairly mature, and platforms such as UHC's Q3ADE are priced competitively with ORB.

CORBA also offers several business drivers for its adoption for communication management solutions:

- There is a much wider base of CORBA developers when compared to CMIP, due to its widespread applicability to multiple market sectors. This makes development resources cheaper and more flexible.
- CORBA offer better portability and range of programming languages than current CMIP platforms.
- CORBA has a proven ability to integrate with a wide range of different business systems.

These business drivers have therefore led initially to the development of CORBA-CMIP coexistence solutions, namely the TMF/X-Open's JIDM gateway specifications. An implementation of these from UHC has been applied in FlowThru as described in Annex 1. In addition there is now increasing interest in adopting CORBA as a technology for TMN. A CORBA Q3 specification is near acceptance in T1M1 and is also being considered in ITU-T SG-4.

As has already been discussed however, there is increased interest in the distributed system community in building on technologies such as CORBA, RMI and DCOM to provide a component-based approach to application development. This aims to provide the open platforms needed to allow application developers to move from the writing of distributed application to their assembly from off-the-shelf component that will increasingly be obtained from multiple third parties.

With such technologies application developers can, for example:

- Examine and configure components at runtime
- Graphically "wire" components into applications
- Manage persistence, transaction and security behaviour

Component technologies, such as the OMG's CORBA Component Model and Enterprise Java Beans, typically support three tiered architectures, with different capabilities particularly for persistence and business tier components. The core of these specifications is the openness of the API between a component and the platform software in which it runs, termed a container. It is this openness which should allow the development of components by multiple third parties that can be easily integrated together using component assembly tools. The component-container API addresses how component and their interfaces are located, how the lifecycle of components are managed, how components export information on their capabilities, how component capabilities can be configured by application developers and how the security, persistence and transaction feature of a component can be managed.

The advantage of component technologies build on those of distributed technologies and are therefore equally applicable to communication management. There is therefore a concomitant motivation to implement some mechanisms for co-existence of existing TMN systems with systems implemented using component technology. In addition, such a mechanism needs to support the migration of TMN systems to ones that can be implemented in future using component based mechanisms.

A possible mapping of the three component tiers onto the TMN logical layers is given in Figure 3-3. This provides a starting point for examining some of the issues related to TMN/CMIP-component co-existence and eventual migration. Such an architecture would provide for greater reusability of management component, and introduce more flexibility and reusability in the access to corporate data

in the persistence tier. The issue of integration with TMN/CMIP will mostly be evident in the Element Management Layer, where there is the most existing investment in this technology.

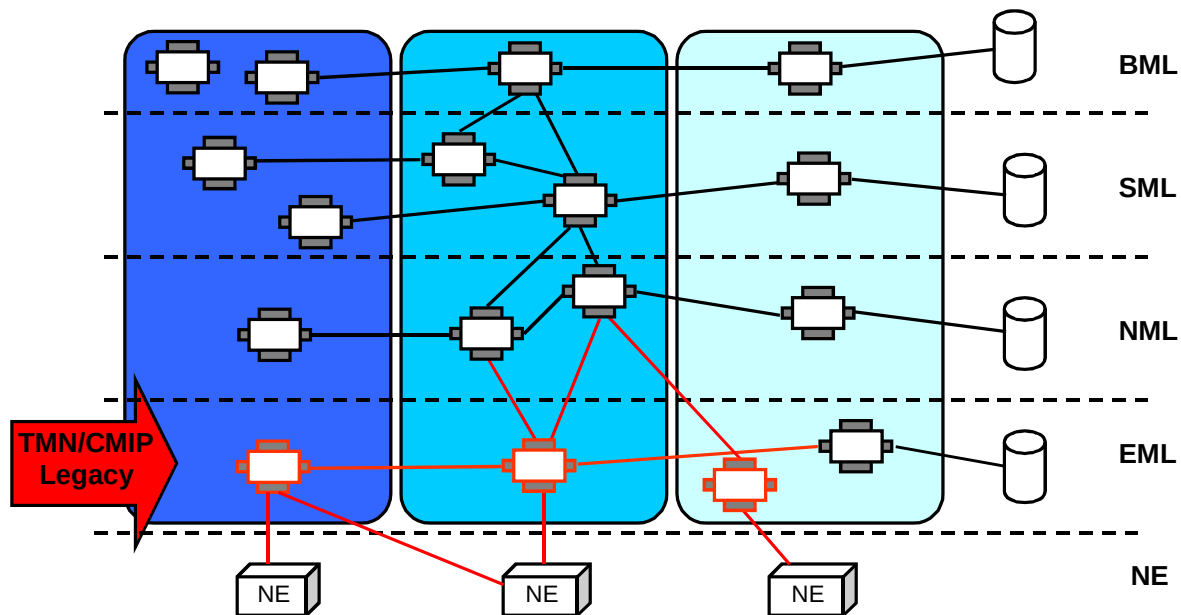


Figure 3-3: Potential TMN-Component Architecture

Requirements co-existence can therefore be summarised as:

- To allow component-based application developers have “component-view” of agents using the same application assembly productivity tools they would use with native component-technology software
- To allow TMN/CMIP agents (typically NEFs) to simultaneously support existing TMN/CMIP managers and component-technology based managers.
- To retain openness in TMN agent procurement
- Introduce openness and better reusability with manager component integration

To support the core requirements for a component-oriented view of existing TMN/CMIP agents, application developers should be able to perform the following on agents using application assembly tools:

- The “wiring” of manager components to TMN/CMIP agent components for both operations and notifications.
- The introspection of TMN/CMIP agent components, i.e. being able to observe the capabilities of a running agent.
- The location and navigation of running TNM/CMIP agent components
- The management of security, transactions and persistence behaviour of TMN/CMIP agents
- The full capabilities of the CMIP protocol implemented in TMN/CMIP agents should be made available to component-technology managers

Several approaches could be taken to achieving these co-existence requirements. One approach would be to provide multi-protocol support in the container, so that each instance could potentially access a CMIP agent. This could use a CMIS-like API such as the course-grained approach proposed in [1m1]. However, this would require major additional software included in each instance of a container running on a process node, as well a presenting scalability problems if other management protocols had to be accommodated. A more flexible approach, that would be more in line with the

architecture of CORBA Components and EJB, would be to provide access to a service that performed a gateway function between the component interfaces other services typically available in the container and TMN/CMIP agent implementations. Such a TMN/CMIP gateway could be based on existing JIDM gateway implementations, but with additional interface for implementing the additional component API function required. Such an approach is outlines in Figure 3-4.

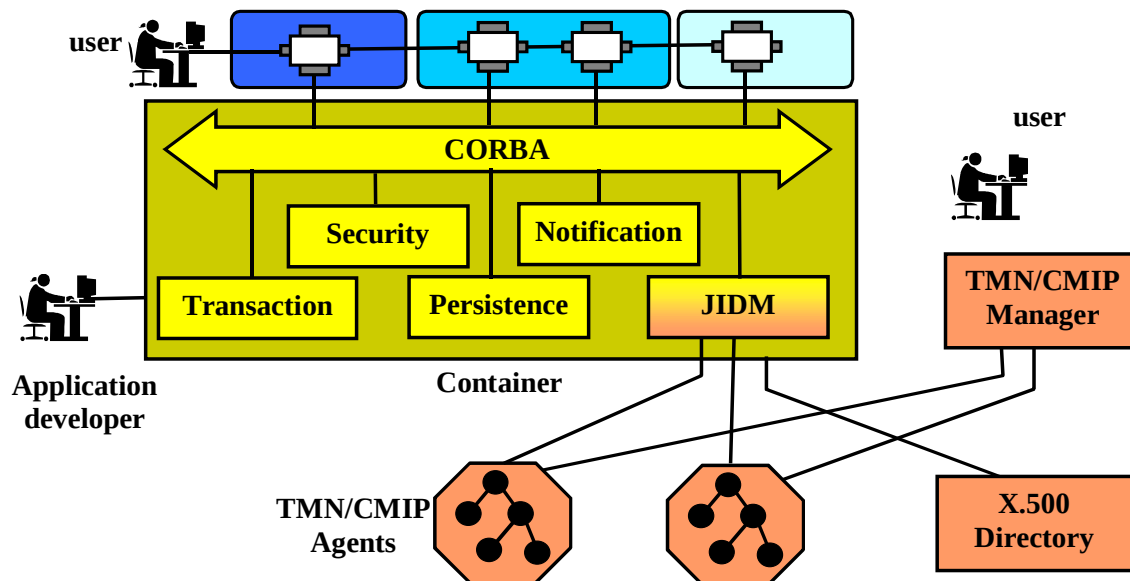


Figure 3-4: TMN Aware Container

The additional function provided by such a gateway would be:

- Integrating naming in container with naming in TMN domain, i.e. X.500 based naming.
- Supporting introspection, this could be performed by exploiting the X.750 standard implemented in agents
- Integrating container security with X.700 security.
- Integrating container transactions mechanisms with TMN transactions
- The granularity of a “TMN/CMIP components” as seen from the application developer point of view. Such component could be at the level of agent clusters, agents, of managed object instances, of managed object class or of management function.

From this analysis we see that in fact TMN/CMIP support many of the additional functions that make components distinct from their plain distributed object predecessors, and there TMN/CMIP is well positioned to ride the component wave. To achieve this however, a new Technology Integration Point needs to be resolved between CORBA Components and EJBs and TMN/CMIP.

3.4 WORKFLOW FOR MANAGEMENT

This section examines the application of workflow to management, using the FlowThru accounting system as a case study.

There is a growing awareness that workflow management tools and techniques could provide a vital element in the co-ordination of distributed components within different provider domains whilst allowing greater flexibility and the necessary degree of autonomy [rusinkiewicz]. Because of the introduction of new services, new relationships with other service providers or new functionality or equipment, the management components (form service provider and network provider systems) must be capable of adapting rapidly to changes in the way the business process is executed.

A workflow management system is a system that defines, manages and executes workflow processes through the execution of software whose order of execution is driven by a computer representation of

the workflow process logic. Workflow technology incorporates the benefits of co-operative information systems, computer-supported co-operative work, GroupWare systems, and active databases. Workflow management technology addresses the following requirements:

- Improved efficiency, leading to lower costs or higher workload capacity
- Improved control, resulting from standardisation of processes
- Improved ability to manage processes; identification and analysis of problems
- Cost reductions, where cost can be a euphemism for staff
- Increased quality or capacity while controlling costs
- Construction of unique customised business processes to deal with specialised management work practices
- Improved information distribution, and elimination of the delays caused by the need to move hard copy information around the organisation
- Reduced bureaucracy improved the quality of work, decreased cycle times, and acquisition of better management information about business processes.

Thus workflow management can be considered a very attractive technology for integration and interrelation of telecommunication management components. The purpose of applying workflow management technologies in the service management problem domain is to integrate and re-purpose management components that resolve telecommunication business problems, and to automate telecommunication management services. Such enhancements, i.e. the interrelation of service management components, reduce the business process complexity, improve resilience, and improve the overall performance of the network operators' business process.

There are many differences between the architectures, which are used by workflow systems. However most of the workflow systems fall into one of two broad categories [stark]:

- Forms and messages based workflow systems which performs electronic routing of forms to user's e-mail in-boxes.
- Engine based workflow system, which communicates with humans or components via specialised client software.

It is the workflow engine based approach, which we will focus on to achieve management component integration.

3.4.1 WORKFLOW ENGINES

A Workflow Management System (WFMS), as defined by the Workflow Management Coalition, is a system that defines, creates, and manages the execution of workflows through the use of software, running on one or more workflow engines, which is able to interpret the process definition, interact with workflow participants and, where required, invoke applications (or components) [wfmc]. A workflow engine is the basic workflow management control software. This software is often distributed across a number of computer platforms to cope with processes, which operate over a wide geographic basis

The workflow engine controls the flow of work (sequences of management activities which form a management business process) through the system by interpreting the management process rules to determine the scheduling of required activities, and invoking the relevant management components. The engine is responsible for:

- Business process creation, deletion, and management of process execution from instantiation through completion
- Control of the activity scheduling within an operational (business) process.

- Interaction with management components and/or human resources (which execute the required management activities).
- Monitoring and control of the management processes in execution.

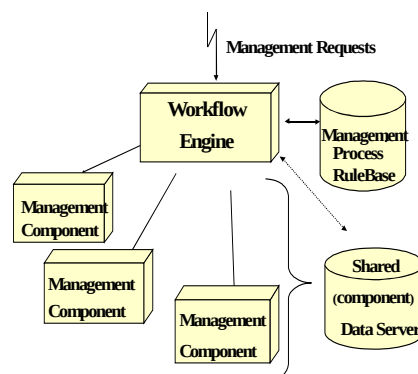


Figure 3-5: General Workflow Architecture

Figure 3-5 illustrates a generalised workflow engine which accepts a (management service) request and based on its process rulebase, invokes the correct sequence of components and store application specific values in a shared data server.

There are a great many products and research projects which offer support for workflow management. In April 1996, 250 products claimed to support workflow features and/or workflow management [sheth96]. This constituted a market size of more than one billion dollars. The number of products has risen since then.

Many of the products simply provided a means of graphically representing a business process using techniques such as dataflow, digraph, flowchart, network, orgcharts, pertcharts etc. e.g. Zippen [castellani]. Others are data management systems, which use e-mail, imaging, databases, electronic forms, engineering drawings etc. to collaboratively process documents or data. Groupware also forms part of this group, Lotus Notes being a good example.

All of these systems have an emphasis on office processes, e.g. imaging, document routing, enhanced mail. However a number of limitations are evident with these types of workflow systems [sheth97].

- Lack of support for heterogeneous computer systems
- Incompatibility between workflow products
- Failure to capture distributed/true nature of infrastructure in business model
- Scalability not achieved
- Very little support given towards fault-tolerance and reliability.

Most of these products were designed for small collaborative projects with small loads. As such, they are unsuitable for large-scale workflow management, potentially involving several thousand users, hundreds of thousands of concurrently running processes and several thousand sites distributed over wide area networks [alonso]. Research projects, however, have confronted many of these issues. Many current research projects are drawing from technologies such as objects, the World Wide Web, CORBA, transaction processing, Java and others in order to help solve some of the problems mentioned above.

3.4.2 CORBA AND WORKFLOW

CORBA is used in varying degrees within many workflow research prototypes. In the simplest case, CORBA is used for database access or as a wrapper around legacy applications. The Mentor project uses CORBA to provide a uniform interface to heterogeneous invocable applications [weissenfels].

The WebFlow project has a CORBA based CLF co-ordinator which communicates with legacy applications, a relational database and a document management system through CORBA [grasso].

The WorkWeb project uses a network of CORBA-based agents, where each agent represents a resource or a participant [tarumi]. These agents collaborate and vie for resources. In OrbWork, again based on METEOR2, transactional concepts are implemented using CORBA to achieve fault tolerance [das]. It includes a layer of CORBA-based system components and failure detection mechanisms, which increase availability and allow recovery. Persistence and scalability are other key requirements, which led to the use of CORBA.

The OrbWork project is a CORBA-based workflow engine which contains a 'Workflow Model Repository', a task manager (combining the duties of the scheduler and dispatcher), a monitor (holding state for the system as a whole) and tasks (wrappers around legacy applications).

3.4.3 WORKFLOW STANDARDISATION

The standardisation of workflow systems has been on-going since 1993 with the formation of the Workflow Management Coalition WfMC (an industrial consortium which set about standardising an architecture for workflow engine based systems, and several interfaces for application invocation, process definition, process management and system interoperability. In 1998, the OMG ratified the definition of a workflow facility, which was based on the WfMC standards.

3.4.4 CASE STUDY – WORKFLOW IN THE ACCOUNTING TRIAL BUSINESS SYSTEM

The ACTS FlowThru project evaluated the workflow based component integration technologies, across a set of TM Forum related Service Accounting business processes. In this area, FlowThru integrated a TINA compliant service management platform with a workflow engine and re-usable accounting service components. The workflow engine is used to integrate a range of accounting components e.g. account manager, Tariff Control, Bill Control, Charge Control, User Metering Management etc. Figure 3-6 outlines the TINA architecture and the workflow engine. The scenario depicted is that for a customer accessing a TINA based Service Management system (Accounting Session Component). The service is being offered by a service operator how is independent of the ATM network operator. The ATM network is managed using TMN technology and ATM usage information is compiled into Call Data Records and emitted as charging notifications. They are received by a JIDM compliant CORBA/CMIP gateway (in the service operators domain) and delivered to the service operators accounting system in CORBA format, where charges for each customer for ATM usage is extracted. The service management system has interfaces into the Subscription Component and Accounting Component. The Accounting Component is actually made up of a WorkflowEntry Agent, the workflow engine and several accounting specific components.

It is the workflow engine that co-ordinates the interactions and sequences of these service accounting specific components to ensure the desired management activities are carried out e.g. generate a bill for the customer, generate tariff information, provide account management controls etc. All component interfaces are specified in CORBA IDL and components execute of a CORBA based infrastructure.

The use of the engine for the accounting component allows the easy introduction of new accounting components without disruption or changes to other accounting components. On introducing a new component into the accounting system, the operator needs to either compose new or amend existing (accounting management) business process(es) and data flow rules. This composition/amendments are required as the engine uses these business rules to decide which accounting components to invoke to satisfy a management request.

In order to execute an accounting business process, an invocation is made to the 'work entry agent' component of the workflow engine. This agent contains a mapping of external accounting service offerings to accounting business processes. On determining the appropriate accounting business process, the work entry agent creates an instance of that business process and invokes the scheduler to start enacting the accounting (business) process instance. The work entry agent also stores any

parameters (which have been given when invoking the accounting service) in the Shared Data Exchange (SDE). The SDE is a distributed information server, which is interrogated on behalf of the accounting components during the execution of accounting activities. Once invoked, the scheduler logs the business process instance (workflow) information into the workflow information server. This workflow information server maintains the state of execution of the business process throughout its execution. However, all accounting (component or application) data is not passed through the engine but is maintained in the Shared Data Exchange. The scheduler then asks the Knowledge Server to determine the next activitie(s) for the process instance. The knowledge server contains several strategy objects which are associated with different business process knowledge. These strategy objects interrogate, at runtime, a knowledge base of process rules to determine the next activitie(s) to schedule. The separation of knowledge base, knowledge server and scheduler allows the knowledge to be potentially represented in different techniques/technologies.

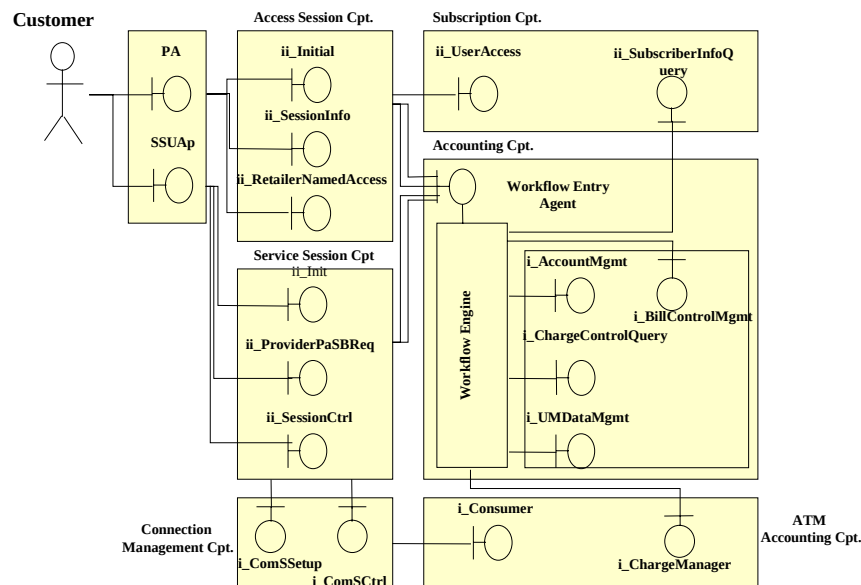


Figure 3-6: TINA based Service architecture using Workflow engine based Accounting Component Integration.

For the trial, a Java Expert System Shell based knowledge-base was used to specify the accounting business processes. Figure 6 illustrates the workflow engine architecture and shows the interactions between the work entry agent, knowledge server, accounting process rulebase and workflow information server. When the scheduler has identified the next activitie(s), it invokes the dispatcher. The dispatcher is responsible for mapping the activitie(s) to component(s). The dispatcher then invokes agent(s) on the component(s) to ensure the activitie(s) are carried out. The agents are responsible for retrieving the necessary input parameters for the accounting components from the Shared Data Exchange. This retrieval is made simple for the agent as the Shared Data Exchange offers a high level query interface and contains meta-data concerning the information flows between activities (and components) required to execute the accounting business processes. The agents invoke the accounting components in whatever means those components support. For simplicity of diagramming Figure 3-7 only depicts three account components. However in the trial seven accounting application components were used. Most of these accounting components were originally developed in other ACTS research project e.g. Prospect, Vital.

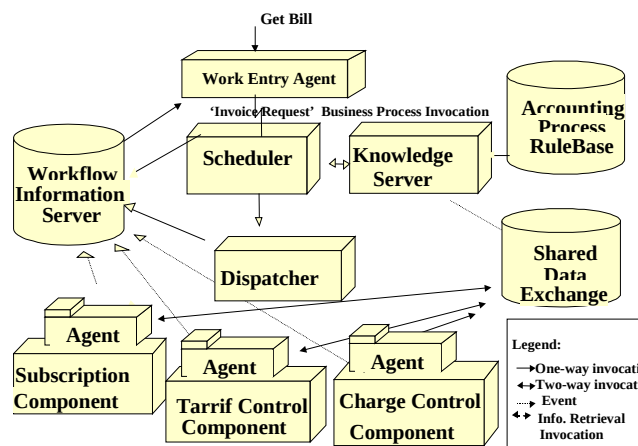


Figure 3-7: Workflow Engine driven Service Accounting

The agents inform the other workflow engine components of the status of the execution of each of the accounting activities by emitting (workflow) typed events. In this way, the workflow information server, and scheduler monitor the state of the business process instance and can schedule further activities in the accounting business processes. The workflow entry agent, dispatcher, scheduler and component agents are multi threaded to allow concurrent execution of business processes. The agent code is 80% pre-written for any application component. The only new code needed is to actually invoke the component itself (i.e. all interactions with the workflow engine, shared data exchange, and use of the CORBA naming service). However even this code could be pre-written if the application component is a CORBA object and a dynamic interface type is used to invoke it.

On completion of the last activity the scheduler sends a 'process instance completed' event. The workflow entry agent retrieves the results of the process instance from the shared data exchange and passes it back to the original accounting service requestor.

In the FlowThru trial, the workflow engine and accounting components were seamlessly integrated with a pre-existing TINA SA compliant platform. All workings of the engine were transparent to the TINA platform.

3.5 CONCLUSION

This chapter has outlined the context in which telecommunication management components would be developed and operated. The irresistible move towards componentisation of the management systems has been discussed. The paper has also discussed key integration technologies integration technologies for management, namely, TMN gateways and APIs, components and workflow.

The similarities and differences between existing TMN recommendations and the new component technologies is noted and as a result as initial proposal for the integration and co-existence of these technologies is proposed..

With regard to the Workflow case study, the use of the engine had several positive effects on the overall design and implementation. Firstly, it was possible to directly map the accounting business processes into the workflow engine rulebase. This provided a means of validating the system behaviour at a high level. The engine also facilitated the ease of introduction of new components as well as the composition of new business processes based on the existing accounting components. Research into the workflow engine is ongoing to provide improved (management) business process management. Research is also ongoing to provide tools for management service creation based on the workflow engine integration.

4 SERVICE FULFILMENT

Mature telecommunication management standards exist largely at the element and network management layers of the TMN logical layer architecture but there is little in the way of standard information models or architectural guidance to those wishing to integrate management software across the network to service management layer boundary. One industrial body that has attempted to address both service and network management in an open and integrated way is the TeleManagement Forum (TM Forum). It has developed a business process model [tmf-910] that, based on surveys of major service providers, gives a detailed breakdown of the set of business processes that typically encompass a service provider's operations management activities. The interactions required between processes in different providers are also identified. This model, summarised in Figure 4-1, is intended for use as the basis for identifying the requirements for the ongoing development of agreements on common interfaces and information models within the TM Forum. The model is partitioned into processes that relate to customer care, to internal service development and maintenance and to the management of the provider's networks and systems. The processes are also grouped vertically into major service management areas, i.e. the fulfillment/delivery of the service, the assurance/maintenance of the service and the billing/accounting for the service. Individual processes are defined in terms of activities within the process and of input and output triggers to the process.

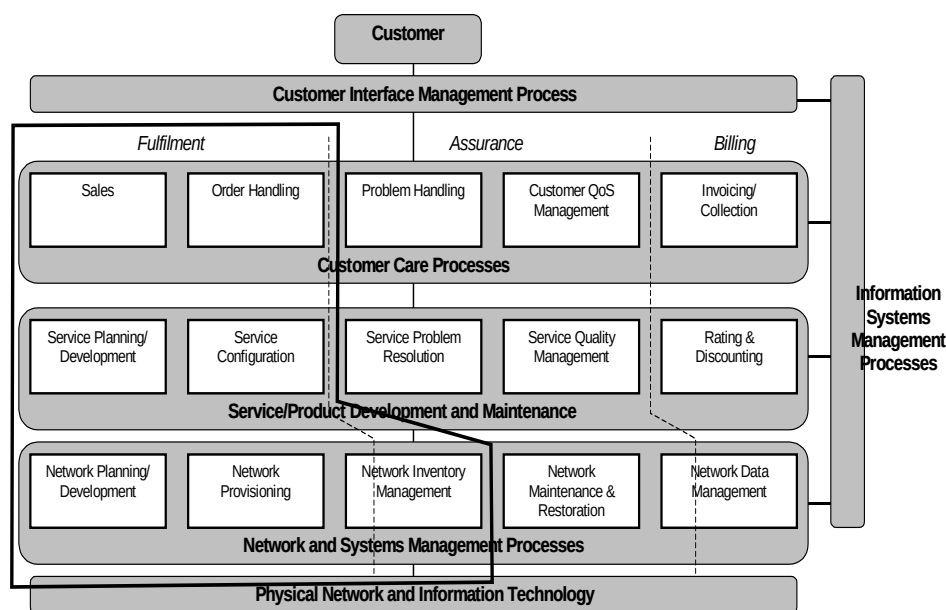


Figure 4-1: TM Forum Business Process Model

Few of the information agreements that have emerged from the TM Forum to date have addressed vertical interactions between service and network level processes. However without guidance on how to develop open interfaces between management systems operating in different logical layers within a TMN there will be a tendency for commercial solutions to be proprietary stovepipes, e.g. between network level accounting and service level billing software.

A similar situation exists in the Telecommunications Information Networking Architecture Consortium (TINA-C). This group has aimed to develop a comprehensive architecture for telecommunications control and management based on Open Distributed Processing principles as defined by ITU-T in [x901]. It generated detailed models for the integrated control and management of multimedia services and broadband networks based on existing concepts from TMN, IN and ATM. Service Management, principally subscription and accounting management, has been addressed in the TINA Service Architecture [kristiansen]. Network Management has been addressed in the Network

Resource Architecture [tina-nra], which draws heavily on existing network management models, e.g. M.3100, in the structure of its Network Resource Information Model (NRIM) [tina-nrim]. Both these architectures have exploited ODP concepts and component-oriented models to integrate management functionality with service and network control. However, the only area where management interaction between the two architectures is currently well defined is connection management, which really is addressing control plane concerns.

A telecommunications management system developer faced with the need to integrate service and network management systems can use the TM Forum business process model to analyse their high level requirements. They can then draw on ITU-T, TM Forum or TINA-C specifications for specific service or network level management interfaces. However, there is little available guidance on the integration of such open solutions across business processes that reside in both the service and network management layers. The development of such guidance is the subject of research in the EU ACTS funded FlowThru project. FlowThru is developing guidelines to aid management system developers to construct systems assembled from reusable components that implement business process interactions. In addition, these guidelines will address how system assembly can be supported by new technologies such as Workflow, Enterprise Java Beans (EJB), CORBA Components and management protocol gateways.

This chapter describes the solution to a management problem spanning both the service and network management layers. It specifically addresses process interactions in the fulfilment area of the TM Forum's business process model, as outlined in bold in Figure 1. This involved the integration of subscription management with network planning and provisioning for a switched network service. The approach taken shows how a CORBA component-based approach has been used to integrate existing software components in a loosely coupled fashion and in an easily extensible manner. This solution involved applying TM Forum industry agreements to existing TINA specified components that were themselves the products of previous research projects.

The next section details the fulfilment problem being addressed and puts it in the context of existing TINA and TM Forum specifications. Section 4.2 reviews the features of the components that were assembled in developing the solution and how they were assembled and modified to satisfy the problem requirements. Conclusions are drawn in section 4.3.

4.1 PROBLEM ANALYSIS

The problem addressed by this work is the integration of management functions in the service and network management layers related to the fulfilment of customer orders. The problem analysis considers orders for a switched ATM service, though the outlined system is applicable to any communications service that involved the dynamic allocation of resources. A customer could range from a single-terminal, domestic users through to corporate users with switched ATM customer premises equipment, supporting a number of users via a single UNI.

In terms of the TM Forum's business process model, the problem domain focused on the fulfilment process and consisted of the following business processes:

- **Customer Care Process:** This involved the management of the interactions between the customer and the provider's management system, in particular translating customer requests and queries into appropriate system operations.
- **Order Handling:** This involved the receipt, processing and tracking of orders for services from a customer and their translation into corresponding requests for network configuration changes.
- **Service Configuration:** This involved the installation and configuration of a service for a specific customer, including any customer premises equipment.
- **Network Provisioning:** This involved the reconfiguration of network resources so that network capacity is ready for the provisioning of services.

- **Network Planning and Development:** This process was responsible for designing the network capability to meet specified service needs at the desired cost and within operational constraints (e.g. QoS requirements), determined principally by service level agreement with customers.
- **Network Inventory Management:** This involved the installation of physical equipment in the network, i.e. trunks, subscriber lines, switches.

Though the Service Planning and Development is included by the TM Forum in this area of the business process model, it did not play a part in this solution. Within the context of TINA, the problem may be expressed in terms of business roles and reference points between these roles as defined in [tina-rp]. A mapping had already been suggested in [lewis99a] between the business roles defined by TINA-C and the business processes in the TM Business Process Model. Based on this, Figure 4-2 shows the overall business context of the problem being addressed in terms of the business roles played by the organisations involved, the business processes undertaken by those roles and the TINA reference points that exist between those roles. This mapping aided system developers understand where existing models from TINA and the TM Forum could be applied, as well as helping them position the resulting solutions for contribution to future standards.

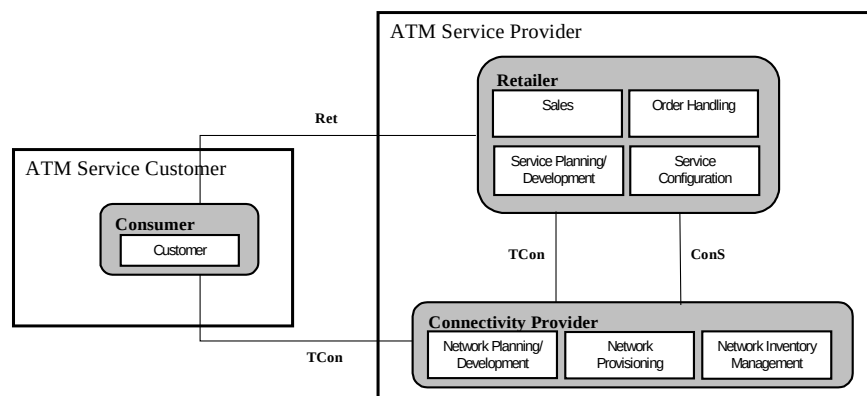


Figure 4-2: Business Process to Business role mapping for ATM Service Fulfilment problem.

As the business problem under examination did not involve multiple providers, the federation reference points between the roles within the ATM Service Provider (available in the full TINA business model) were dropped. As the focus of this work was on the development of functions in the management plane, aspects of network and service control were assumed but not analysed or developed any further. The TCon reference point, which dealt with control of network level interconnection, was not examined on the assumption that existing UNI protocols would be used for connection set-up. Instead, the focus was placed on the management interactions that occurred across the Ret and ConS reference points. The Ret reference point was derived from the TINA service architecture that included a detailed model of how customer subscriptions and user's individual profiles could be managed as described in the section on the subscription management component below. ConS was extended with additional network configuration and planning management features as specified by the REFORM project and described in the sections on Configuration Management and Network Planning [reform].

Existing TINA-specified models provided a core set of concepts that could be used in further analysing the requirements of the ATM service fulfilment business problem and the requirements for interactions between the business processes identified as being relevant to this problem. The key concepts were:

- **Network Access Point (NAP):** This is the representation within the service provider of the point on the network via which the customer uses the switched ATM service. This offers a UNI to the customer and can support multiple simultaneous connections.

- **Class of Service (CoS):** To restrain the complexity of the network dimensioning problem while at the same time offering useful service definitions to the customer, the notion of CoS is introduced to quantise the space made up by ATM QoS types and the parameter ranges they may exhibit.
- **Service Usage Group (SUG):** This is derived from the TINA subscription management model and represents a group of users who may use a particular type of service, defined as a service profile. It also defines the set of network access point from which this group of users can access this service profile.

To analyse the problem further, business roles within the business stakeholders were analysed to determine the responsibilities they have with respect to each other. Based on these responsibilities, use cases were drawn up documenting the functionality of the whole business system from the perspective of the actors who play the business roles. The use cases effectively defined the scope of the scenarios being addressed. They were restricted to the following:

- Adding and withdrawing a class of service to/from those that a customer may use when subscribing to the ATM service.
- Customer ATM service subscription: providing contractual and accounting information on the customer; specifying the physical sites from where the customer will use the service together with lists of users at each site; the classes of service they will use and under which service level agreement parameters they will use them.
- Adding or removing a site from which the customer's users may use the services.
- Authorising or barring a user from being able to use the service.

However, much direct analysis of the internal business processes, intra-component interactions still had to be identified. This was solved by breaking down individual use cases into separate activities that could be represented in a UML activity diagram, see chapter 2, section 2.3. By representing TM Forum business processes as a set of swim lanes and placing each activity in the appropriate lane, activities could be grouped by business process. As each business process could be encapsulated by a component, intra-component interactions were determined by examining the information exchange required to progress from one activity to the next.

The next section presents how this analysis was mapped onto the design of existing management components and the identified interactions with respect to the outlined use-cases.

4.2 INTEGRATED SYSTEM DESIGN

The development of the system analysed in the previous section was driven by the fact that most of the required functionality was already available as existing components, the re-use of which was a key concern. These components were the results of previous research projects that had implemented and extended portions of the TINA specification set, implementing systems focusing in either the service or the network management layer.

The need for reuse and integration of existing software led to a component-based development approach. Typically, components can be defined as independently deliverable packages of software services. It is worth noticing that component-based development can co-exist with OO analysis and development. While the later is focusing on analysing and designing systems under a specific business problem, at the same time it is useful to perform the packaging of the developed system into a set of self-descriptive, application-independent components using the modelling constructs of a component technology.

This section describes how elements of the emerging CORBA component model [orbos/99-02-05] were used for modelling, packaging and seamlessly integrating the components of our system. It shows how the utilisation of a component technology provides easy component assembly and integration in a system designed using the OO development methodology. Three components (cf. Table 2) were identified, modelled and integrated according to a subset of the CORBA component model.

Components	Business Process
Subscription Management	Order Handling Service Configuration
Configuration Management	Network Provisioning Network Inventory Management
Network Planner	Network Planning/Development

Table 2: Building Blocks and their mapping to relevant Business Processes

Before describing how the merits of the component technology were exploited, the functionality of the components is outlined in the next subsection. Based on this high level description of the functionality of the components, Table 2 shows how they mapped onto the TM Forum business processes relevant to the fulfilment problem.

4.2.1 COMPONENTS

4.2.1.1 SUBSCRIPTION MANAGEMENT

The Subscription Management component comprises all the management functions needed in order to define service offerings, administer customers and users, and manage the details of service provisioning. The design is based on the subscription model developed in TINA [kristiansen]. The use cases implemented by the component cover the management of service offerings, of subscribers and of subscriptions. Management of service offering is concerned with the definition and adaptation of offered telecommunications services, from a management point of view. Management of subscribers consists of the creation and deletion of subscribers, the update of subscriber details, and the definition of subscriber's end-user groups and network sites. Finally, subscription management covers subscribing customers to services, update subscription details, cancel subscriptions, and the authorisation of end-users for specific services.

The component is decomposed into Computational Objects (CO). These define separate functional units needed to manage subscriber organisations and end-users, different types of services, and subscriptions to these services. The COs includes Service Template Handler, Subscriber Manager, Subscription Registrar, and Subscription Agent. The Subscriber Manager CO manages information associated to subscribers, e.g., subscriber details and user groups. The Service Template Handler CO is responsible for handling service templates that represent service capabilities provided by the provider. The Subscription Registrar CO acts on behalf of a provider in dealing with subscribers wishing to subscribe to a service and establish a service level agreement. The Subscription Agent CO is closely related to the User Agent CO that manages the access session of the TINA Service Architecture. A separate management application is typically used to allow customer administrators or provider administrators to access the subscription component.

The implementation of the subscription management component used for this problem was developed in the Prospect project, where it was reused in several different services. The definition of the component assumed a predominantly synchronous mode of operation in its interactions with users and other service control and management components. However, the analysis of the fulfilment problem in the previous section revealed that the overall fulfilment process involved unbounded delays due to network installations and reconfigurations. The subscription management component therefore had to offer an interface to users that would enable them to manage prolonged order processing by the provider's systems. A further CO, the Order Handler, was added to deal with the reception, tracking and possible cancellation of orders received from customers. The basic abstractions used in this component were taken from the TM Forum's Order Handling business agreement [tmf504].

4.2.1.2 CONFIGURATION MANAGEMENT

The Configuration Management (CM) component spans the network and element management layers of the TMN hierarchy. Its scope is in general two-fold: to maintain a map of network resources both for inventory purposes and also for showing the topological relationships among resources; and to support the activation, deactivation, reservation and release of resources. The CM component presents a “server” interface to other components e.g. to Subscription Management and Network Planning and Dimensioning in the case of the fulfilment system described in this paper. In addition, it acts as a client to network nodes with management interfaces e.g. to ATM switches.

In TMN systems, configuration information is typically held in one or more hierarchically organised network Operation Systems (OSs), whose model conforms to the Generic Network Element Information Model [m3100] and to the emerging ITU-T Network Information Model. In the case of TINA, configuration information is split in two parts: dynamic information, concerning on-demand or semi-permanent trails, which is held in the Connection Management (CM) subsystem; and static information, which is held in the Resource Configuration Management (RCM) subsystem. In both cases, a client of the CM component may request the activation/deactivation of a resource, e.g. of a NAP for a particular subscriber, or the reservation and release of a resource, e.g. an ATM VP trail for planning and provisioning purposes.

The CM component in FlowThru comes from the REFORM and VITAL projects, which used TINA instead of TMN principles. Given the fact that TINA concentrates mostly on Connection Management which covers the equivalent control aspects, modifications and extensions were necessary. TINA prescribes the Network Resource Information Model [tina-nrim], which is based on equivalent ITU-T models, but the associated information objects are “hidden” behind computational interfaces. The NRM model was extended while a CORBA-based computational object with TMN Q3-like IDL interface was designed to provide access to those objects in a flexible fashion [pavlou97]. An additional important architectural consideration was that the distinction between static and dynamic resources was not considered appropriate.

The CM component comprises two computational objects: the ConfM Network Map (ConfM-NM), which acts in a similar fashion to a TMN configuration OS; and the ConfM Connection Manager (ConfM-CM), which establishes and releases VP trails. The ConfM-NM embodies the functionality of TINA RCM while the ConfM-CM embodies the functionality of TINA CM. The difference is that ConfM-NM holds also dynamic resources, e.g. VP trails, providing a consistent view of all the necessary information for planning and provisioning purposes. An additional interface to the ConfM-NM object is being developed in FlowThru for presenting a customised facility for NAP management to the Subscription Management component.

4.2.1.3 NETWORK PLANNING AND DIMENSIONING

The Network Planner (NP) component is located at the network management level of the TMN hierarchy. From a TM Forum’s viewpoint, it maps to the “Network Planning and Development” process.

More specifically, the Network Planner consists of the following three subsystems:

- The Class Of Service Model (CoSM) subsystem.
- The CoSM maintains a repository of the CoSs supported by the network. Each CoS is defined in terms of its bandwidth characteristics, performance targets and restoration characteristics. It should be noted that the CoSM restricts, yet is based on, the services offered to users through the Subscription Management component.
- The Predicted Usage Model (PUM) subsystem.
- The functionality of the PUM encompasses the maintenance of a valid model of the anticipated traffic to the network. Based on this model, the PUM supplies the VPC_TD (see below) with traffic predictions, required for the design of the VP layer. The VP layer (set of VPCs) and the static aspects of the VC layer (set of admissible routes per source-destination and CoS) are constructed so that to satisfy the predicted traffic demand.

- Anticipated traffic is modelled in terms of the numbers of connection requests per CoS between source-destination pairs. The anticipated traffic model will be acquired from information regarding the subscribers of the network. This information is given by the Subscription Management component and it describes: the number of users that are subscribed to use the network; the specific sources (i.e. the customer's NAPs) that generate traffic; and the specific CoS characteristics of the traffic. The information regarding the CoS being used is extracted by the Subscription Management component by mapping the QoS characteristics of the services (as in SLAs) to network CoSs.
- Additionally, data regarding historical network traffic will be taken into account in order to make predictions as accurate as possible. Normally, the model needs to be validated and modified against actual network usage. However, this is not considered within the context of FlowThru.
- The VPC Topology Design (VPC_TD) subsystem.
- The objective of VPC_TD is to design and redesign, whenever necessary, working VPCs and sets of admissible routes taking into account the existing CoSs. In addition, the existing component also designs suitable protection VPCs and allocates the appropriate amount of protection resources, thus providing for a resilient network design. VPC_TD's task is based on the predicted traffic and the physical constraints (e.g. connectivity, capacity of ATM links) of the network. These input parameters are subject to changes over the lifetime of a network. VPC_TD offers a certain level of flexibility to adapt to such changes.
- The tasks of designing VPCs and routes based on them are not seen in isolation as they are tightly coupled; routes are based on VPCs; and VPCs are defined for routing. Along with these tasks, the tasks of designing appropriate protection VPCs and subsequently allocating protection resources across the network is considered. All these tasks are part of an iterative process involving complex optimisation problems aiming at providing cost-effective design solutions. The optimisation targets are set according to the business policy of the network operator. Part of this process will be to identify all possible routes between source and destinations and to choose a certain subset of these, according to the performance targets of the CoSs to be carried.

4.2.2 COMPONENT INTERACTIONS AND MODELLING

Each component possessed a set of pre-defined IDL interfaces that had to be preserved to maintain access to its core services. Additional component functionality and the information that needed to pass between the components was analysed using activity diagrams detailing each use-case. The identified component interactions are summarised below:

- The Subscription requires information on the available network-level CoSs to: create service records to which a customer can subscribe; define a set of QoS parameter limits to offer different SLAs for each service
- The Configuration Manager must know what NAPs are needed by a customer so that existing NAPs can be activated or new NAPs installed if none currently existed
- The Network Planner needs to be informed as new NAPs are activated for subscribed customers so that the Predicted Usage Model can be recalculated
- The Network Planner requires information on both the number of users at each NAP and the CoS to which they are subscribed, both influence the Predicted Usage Model
- NAP and user updates must be acknowledged by the Network Planner to indicate to what level each request can be accommodated within an existing SLA
- A user application for the service customer was required. This sends various subscription orders to Subscription Management and needs to be informed of changes in the status of these orders as they go through the various stages of fulfillment.

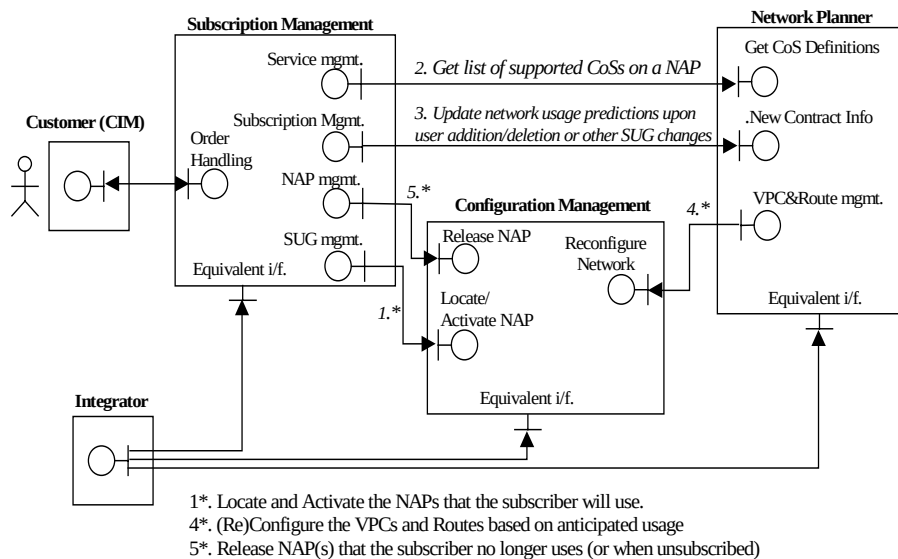


Figure 4-3: Interactions sequence for a subscriber's lifecycle

Figure 4-3 depicts the component interactions and the integration approach followed. The sequence of interactions between the subscription management, the configuration management, and network planning components for a subscription lifecycle are given in Figure 4-3. These interactions and the accompanying IDL definitions could form additional management segments to the TINA ConS reference points. The interactions 3, 4 and 5 can repeatedly happen through the lifetime of a subscription as the customer adds or removes new users and sites.

To promote loose coupling of the components an asynchronous event-based approach to component interaction was taken. An event-based approach was also seen as essential for the Order handler CO in the Subscription Manager which had to mediate the progression of orders between the customer application and the other components.

To better support these interactions, the existing components were modelled by means of a CORBA-based component model. As work towards the specification of a CORBA Component Model is still ongoing (cf. [orbos/99-02-05]), current CORBA platforms do not provide support for building components as of this writing. However, a subset of the forthcoming component model suitable for our purposes was implemented. More specifically, part of the event model, as well as navigation through supported events, as in [orbos/99-02-05], was implemented. The consumer/publisher terminology introduced in the same specification is also used throughout.

The design borrows heavily from the Model-View-Controller classes used to build user interfaces in Smalltalk-80 and described in [gamma] as the Observer design pattern, and more recently deployed in the JDK 1.0+/JavaBeans Delegation Event Model and the CORBA component specification [orbos/99-02-05].

An event is described by a simplified version of the Structured Event, specified in the CORBA Notification Service [telecom/98-06-15], and comprises a fixed-length header that uniquely identifies the event type and a variable-length body consisting of a sequence of name-value pairs, containing the event data depending on the type of the event.

Each component provides the following services

- Navigation through the events that are both published and consume.
- The ability to subscribe an event consumer to an event publisher of a component.
- The ability to send an event to registered components.

The above features are made available through a generic interface (the so-called component equivalent interface) and support a standard set of operations through inheritance from a standard ComponentBase interface.

A specialised “integrator” component was used for configuring the runtime relationships between component instances. Initially, all system components are introduced into system being unconnected (that is, no consumer is registered to receive events from a publisher). By means of the navigation capabilities supported by the component equivalent interfaces the integrator identifies the consumer-publisher communication required and undertakes the appropriate event subscriptions within the system. This requires matching the event types that the components publish and consume.

In our prototype component model implementation, the run-time environment is rather simple, providing for no Containers. However, even with this prototype it is possible to easily plug our components in different systems. The separation of the integrator component enables this role to be played in future system by software configuration applications or possibly workflow based systems with knowledge of event types and component features.

In the above example, design coupling between components is better targeted compared to components with synchronous interfaces by employing standardised event registration and message passing interfaces. An event consumer uses a well-known registration interface to notify a publisher of its desire to receive the latter’s events. Events are passed from event publisher to event consumer via a standardised message-passing interface. This enables components to exchange information with only knowledge of the individual event they are interested in, rather than having to bind to an interface definition that may contain many operations that are not of interest.

New functionality in the form of additional event protocols can be created without impacting existing component design couplings. Also, it is possible to introduce components that may be interested in consuming events for reasons other than their original intention, for instance some of the events from our system could be monitored by additional systems measuring the response times to customer order requests.

Components that receive events for which they have no semantic understanding may simply choose to log them or perform some other generic processing on them. We were able to exploit this final point by creating an event listener capable of demonstrating real-time information exchanges occurring in the system. Because the event listener was registered with all event publishers, each publisher notified the listener whenever it emitted an event. Registration was performed so that there was no need for publisher and consumer to enter into the normal event protocol dialogue, the event listener merely recorded and displayed events. Frameworks can be deployed to expose event interfaces and connect them to consuming components in the same way as our simple integrator component. This allows the flow of information and control between components to be managed more flexibly, being less tightly-coupled to the design of the components themselves. This feature is compatible with the workflow-based approach to OSS integration. Threading ensures that a component is not suspended pending a reply to a given request. This avoids locking problems that may occur in synchronous systems as they are expanded with chains of synchronous requests. The event-based approach still embodies some design coupling between components. In order to process an emitted event a consumer must know how to process a known set of event types. This is a fact from which no component can escape - a consumed event must trigger some predicted action even if this is to ignore it.

4.3 CONCLUSIONS

This chapter shows how problems spanning both the service and network management layer can be analysed using a combination of business processes (as advocated by the TM Forum) and business roles and reference point segments (as advocated by TINA-C).

The chapter also shows, through an example, how such analysis is compatible with the construction of software solutions from existing components. Moreover, it illustrates how the use of UML Activity diagrams mapped to TM Forum advocated business processes can aid the design of those component extensions needed to effect successful inter-connection.

We have shown that the flexible reuse and integration of off-the-shelf software is possible through the exploitation of current component technology, particularly its asynchronous event-messaging. At a practical level, system functionality was extended through the introduction of event logging and monitoring components.

The use of the integrator object demonstrates how dynamic event identification and configuration can also be used to further improve flexibility by reducing component coupling to a runtime configuration issue. This can be enhanced through the development of component and event repositories: a component repository providing a central location within which component interfaces details are stored; an event repository performing a similar task by enabling events to be defined at runtime. Furthermore, event adapters can be deployed between producers and consumers to further customise the event messaging at the component level.

Event coupling is unavoidable, an event is designed to trigger some action in a recipient. However, by using a non-blocking asynchronous event model, coupling can be reduced to a set of generic message passing and registration interfaces. The identification of common event types within a given problem domain would be one approach to reduce event coupling. International bodies such as the TM Forum being instrumental in this through further analysis and development of their business process model to encompass the definition of common event types between processes.

5 ASSURANCE SYSTEM

Fierce competition in open liberalised telecommunications markets and the changing demands of customers using more complex multimedia services are leading to greater emphasis on the provision of customer service as a means of contributing to a service provider's competitive edge. As organisations become increasingly dependent on telecommunications networks to carry out their business, they are requesting guarantees about the quality of the services they are paying for and requiring rapid trouble management from the service provider when network or service performance hinders their ability to conduct business.

Service providers (SPs) are therefore currently investigating opportunities to provide their customers with differentiated service level agreements (SLAs) which state the obligations entered into by both service provider and customer. When a SP offers a service, there is always the possibility that there will be a partial or total failure of that service. Such a problem is known as a 'trouble' and the process of trouble management is concerned with identifying and resolving that trouble. The existence of a trouble can have an adverse effect on the quality of the service as perceived by the customer. Consequently, the agreement of SLAs means that trouble management will be increasingly concerned with the service level where quality of service (QoS) and discount policy competition can be a means for customers to differentiate between service providers.

Because of the QoS dependencies between the service and network levels, an integrated network and service management environment is required. However, the deregulated market makes the process more challenging since the service can span heterogeneous service (and network) provider domains and yet needs to be managed on an end-to-end basis. The provisioning of a multimedia service can therefore necessitate complex configuration and maintenance over a number of administrative domains and involve a variety of actors: service providers, connectivity providers and service brokers. Therefore, the processes of fault detection, localisation and resolution are becoming more difficult, especially as network-related faults have to be associated with specific multimedia end-to-end services, customers, or users.

This paper presents a Service Quality Assurance System that automates interactions between the customer, the service provider and the connectivity provider for the purpose of trouble identification and resolution, SLA fulfilment assessment, and discounting in cases of failure. The system has been developed by integrating TINA and TMN concepts and is based on the TeleManagement Forum business process model. It is shown how MusicShop, a multimedia service that runs over a Premium IP service, can be provided with the kind of support increasingly required by both service providers and customers for SLA and trouble management.

This chapter is organised in seven sections. Section 5.1 outlines major initiatives that have been taken into consideration in developing the Service Quality Assurance System. Section 5.2 describes the business model and the technical approach adopted. Section 5.3 presents details of the MusicShop service, and Section 5.4 discusses the trouble management services. Section 5.5 introduces scenarios for evaluating the system, and Section 5.6 provides some concluding remarks.

5.1 RELATED WORK

The Service Quality Assurance System presented in this paper was developed within the context of the European ACTS project FlowThru. FlowThru is concerned with the overall life-cycle of multimedia service provisioning, including fulfilment [Lewis99d] and accounting [Helleman] in addition to assurance, and is focused on the information "flow through" between customers and multimedia service providers in a multi-domain open service market.

Different consortia have undertaken initial work to handle trouble management processes at either the network or the service level, but have not generally investigated trouble management encompassing both levels. Today's trouble management products are mainly designed to support the internal needs

of a service provider and thus do not support multi-domain trouble management. The FlowThru Service Quality Assurance System integrates network and service level trouble management spanning multiple domains. Its design and implementation is based on concepts from various consortia that have been pragmatically selected and merged into a unified system. The principal concepts influencing the development of this system are presented below.

5.1.1 TELEMANAGEMENT FORUM CONTRIBUTION

TeleManagement Forum (TM Forum) provides the telecom industry with leadership on the most effective ways to streamline the management of communications networks and services. It has recognised the need to support the end-to-end automation of business processes within the service provider environment and has investigated how the processes involved in telecommunications management relate to each other. The current focus is on the integration of all these processes into process “flow through” services built around three high level processes of Fulfilment, Assurance, and Billing of telecommunication services [tmf-910]. The Service Quality Assurance System is implementing a subset of these processes concerned with problem handling.

Within the business process framework, the TM Forum has defined a set of detailed specifications to support important customer-to-business and business-to-business management processes. For multi-domain problem handling the Service Provider to Customer Performance Reporting Business Agreement [tmf-503] and the Performance Reporting Definitions Document [tmf-701] were taken into account. These documents define requirements, concepts, and terms for service level agreements, QoS measurement, and performance reporting and were used in developing the Service Quality Assurance System.

The TM Forum documents Trouble Administration Business Agreement [tmf-501], Customer to Service Provider Trouble Administration Information Agreement [tmf-601a] define requirements, concepts and terms for trouble management between customer and service provider. The documents Customer to Service Provider Trouble Administration Analysis Specification [tmf-601b] and CORBA Interface Specification for Customer to SP Trouble Administration [tmf97] specify the interfaces for CORBA based systems. All these documents have been used for the design of the TINA Trouble Report System (TTRS).

5.1.2 TINA-C CONTRIBUTION

The Telecommunications Information Networking Architecture Consortium has been defining and validating an open software architecture for the provision of telecommunication and information services, known as TINA. TINA defines a set of concepts, principles, rules and guidelines for constructing, deploying, and operating TINA services. The major principles aim at ensuring interoperability, portability and reusability of software components and independence from specific technologies, and at sharing the burden of creating and managing a complex system among different business stakeholders, such as consumers, service providers, and connectivity providers [mulder]. Reference Points are defined to specify conformance requirements for TINA products [farley].

TINA provides a set of specifications such as the Distributed Processing Environment Architecture [gavras], the Service Architecture [kristiansen] and the Network Resource Architecture [steegmans97a]), which have formed the basis for the development of the PLATIN TINA Service Platform (Y.TSP) [eckert] used in the FlowThru Service Quality Assurance System.

Although TINA covers the major market requirements caused by deregulation and globalisation (multi-provider environment, need for flexibility, customisability, etc.) service quality assurance and trouble management issues have not been its primary focus. While trouble management in the context of network resource management is covered by the Network Resource Architecture, the Service Architecture stresses, but does not define, these issues in the service management context. The concepts developed for the Service Quality Assurance System could therefore be used to enhance TINA concepts with trouble management related solutions.

5.1.3 EURESCOM PROJECT P612 CONTRIBUTION

The EURESCOM P612 project was a TMN project focusing on the ITU-T Recommendation X.790 on Trouble Management [x790], which it profiled and validated in the EURESCOM pan-European TMN Laboratory environment [p612][mathan]. It developed a generic, interoperable trouble ticketing (TT) process for the X interfaces involved in trouble management, i.e., the X.user interface between a connectivity provider management domain and a customer network management domain, and the X.coop interface between two peer connectivity provider management domains [p612-d2]. The P612 specifications and GDMO information model have been used to design and implement the network level trouble ticket system (TTS) used in the Service Quality Assurance System.

5.2 MULTI-DOMAIN MANAGEMENT IN THE OPEN SERVICE MARKET

The availability and quality of telecommunication services are of increasing importance as businesses automate and rely heavily on computer-based applications. In the area of trouble management, the objective of all telecommunication actors is the rapid, accurate, and reliable exchange of trouble management information between customers and service providers to minimise trouble resolution time and to optimise customer satisfaction in case of SLA violations.

5.2.1 THE ENVISAGED BUSINESS MODEL

The FlowThru Service Quality Assurance System is being developed for a multi-provider environment where one-stop-shopping requires the customer-facing service provider to handle the various contractual relations necessary to support the multimedia service being offered to the customer. Figure 5-1 shows the various domains involved and identifies the customer domain, and a set of co-operating service provider domains (based on the TINA Business Model) and major roles. The connectivity service being offered is a Premium IP service over an ATM network infrastructure. As this service is capable of being managed, the Premium IP service provider can offer trouble management, accounting, and discounting functionality, which enables the Service Quality Assurance System to encompass management of the connectivity service as well as of the value-added service. The value-added service being offered, MusicShop, is a Web-based public (but secure) file system allowing uploading and downloading of multimedia information for individuals or globally distributed user groups.

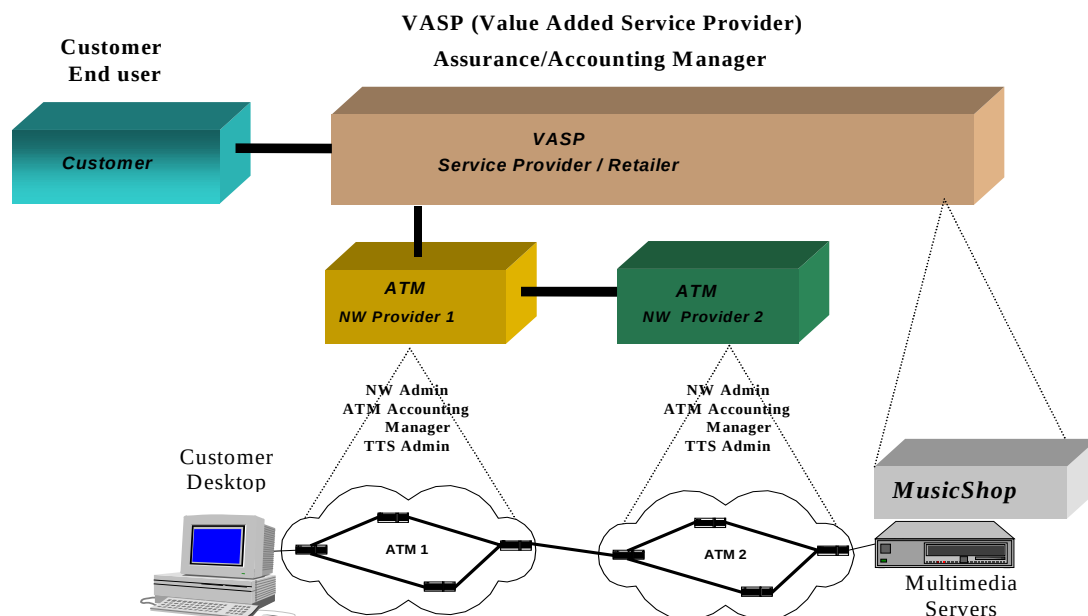


Figure 5-1: The Service Quality Assurance Business Model

Two different connectivity providers have been introduced to reflect the possibility of the connectivity service spanning a number of network infrastructures. To avoid additional complexity it was decided not to differentiate the service retailer from the 3rd party service provider domain so that the same actor performs both business roles. The overall business model addressed by the Service Quality Assurance System therefore contains the following business actors:

- MusicShop Customer plays the business roles of customer and user of the MusicShop service.
- MusicShop Service Provider plays the business roles of Music Shop Service Quality Manager and MusicShop Accounting Manager. As a MusicShop Service Quality Manager, this actor has responsibility for monitoring the service provisioning quality level, detecting faults, correcting problems and handling customer trouble reports. The MusicShop Service Provider has a single contractual relationship with a connectivity provider for provisioning communication services between the customer premises and the MusicShop service provider premises.
- ATM Network Provider 1 plays the role of Network 1 Quality Manager and Network 1 Accounting Manager. The ATM Network Provider 1 is responsible to the Service Provider for all connectivity maintenance and connectivity usage accounting. It is responsible for collecting connectivity and accounting information from itself and any subcontracted third party network provider in order to give a single conceptual view of the connectivity to its customers (in our case the MusicShop). Any connectivity problems are handled by this actor or, if detected internally, reported to its customers. The ATM Network Provider 1 has a third party relationship with another ATM Network Provider in order to provide its customers with a single view of the network, whether or not this geographical area is managed by the network provider itself or by some of its subcontractors. The accounting information presented to its customers takes into account all discounts resulting from network SLA violations.
- ATM Network Provider 2 plays the role of Network 2 Quality Manager and Network 2 Accounting Manager. It is responsible for all connectivity problems and resource usage within its own domain. It only has a contractual relationship with Network Provider 1. Therefore, this actor is only visible to Network Provider 1. All problems detected within its domain are reported to the Network 1 Quality Manager and it manages all troubles forwarded from Network 1 Quality Manager. As Network 2 Accounting Manager, it is responsible for forwarding the accounting information about network 2 resource usage and associated discounting information to Network 1 Quality Manager.

5.2.2 THE TECHNICAL APPROACH

All service providers (including the connectivity providers) offer their services as TINA services. Information is exchanged at TINA reference points [school] making use of CORBA (IIOP at the service level) or TMN technology (CMIP at the X.user and X.coop interfaces at the network level). The use of enhanced TINA subscription concepts allows connectivity or value-added service troubles to be associated with affected services, customers, and/or user sessions. An integration of the TINA Trouble Report System (TTRS) and the TINA accounting system enables discounts in case of SLA violations.

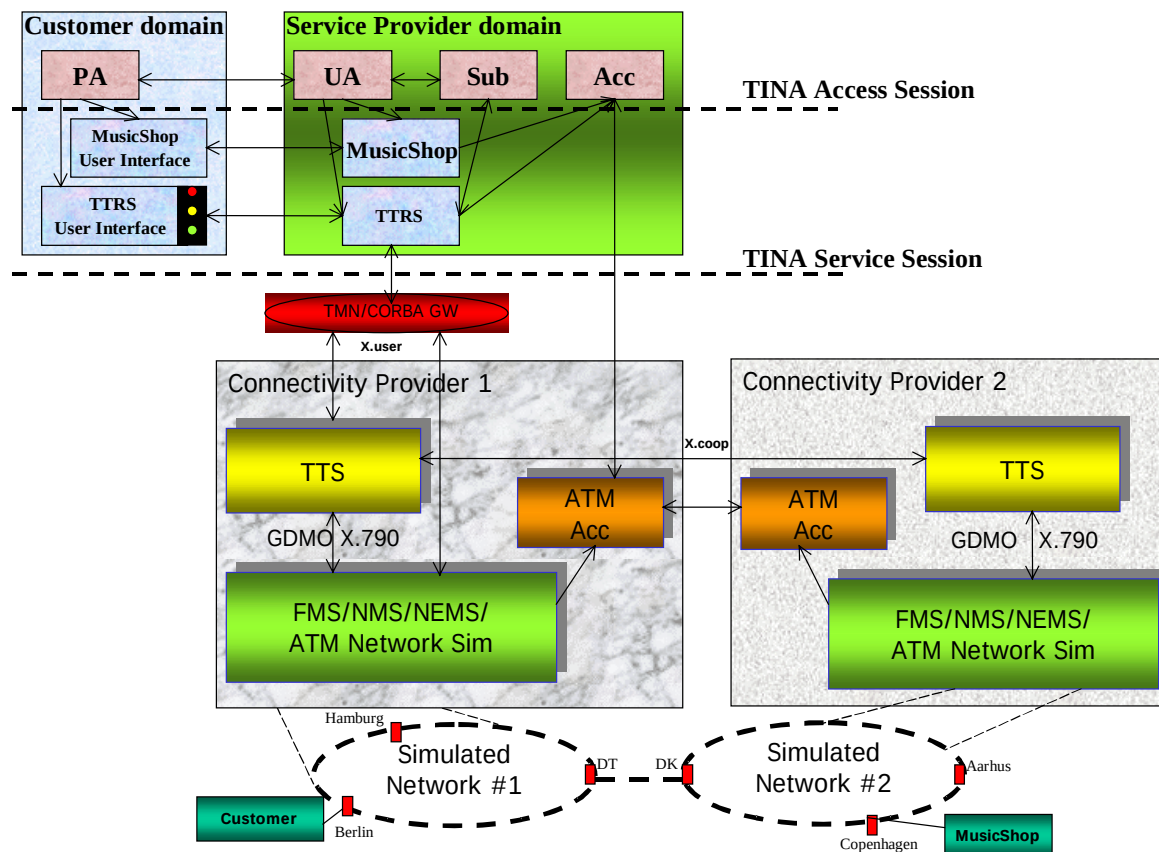


Figure 5-2: Computational Model of the Service Quality Assurance System

Figure 5-2 depicts a simplified computational model of the Service Quality Assurance System configuration at the TINA service level. The TINA service environment provides the infrastructure to run the FlowThru Service Quality Assurance System trial. The MusicShop service and the TTRS are embedded within the PLATIN TINA Platform Y.TSP that provides an implementation of the major TINA service architecture components. They are configured for the service quality assurance trial (service templates, tariffs, customer and user profiles) to enable the services to be offered to the end users.

- The Access Session components (PA: Provider Agent, UA: User Agent) enable service selection and secure service access in a TINA environment. The interface between the customer and service provider domain is based on the TINA Retailer reference point definition [farley].
- The Subscription Component (Sub) contains information related to the registered customers, the existing services, and SLAs between the service provider and customers regarding the use of a particular service and the negotiated QoS. It also contains information regarding the connectivity access points of the customers.
- The Accounting Component (Acc) is responsible for service and network usage accounting. The TINA Accounting System is connected with the ATM Accounting System (ATM Acc) to exchange network usage charge records, and with the MusicShop to exchange service usage charge records. The interface to the TTRS allows discounts to be granted in the case of SLA violations.
- The Assurance Trial Service, i.e. the MusicShop (MusicShop User Interface, MusicShop) is a TINA service that is offered to the customer. Authorised users can access the service from different network access points to download and upload documents. This service will be used to demonstrate trouble management and discounting at the TINA service level.

- The TINA Trouble Report System (TTRS) implements a management process that permits the handling of the quality assurance process at the TINA service level of the FlowThru trial system, including SLA management. The TTRS is considered a specific TINA management service embedded in the TINA environment.
- The Trouble Ticketing System (TTS) implements a trouble ticketing service for connectivity providers. It comprises a manager-agent pair communicating using CMIP, although only the agent component is used in the Service Quality Assurance System.
- The Fault Management System (FMS) is responsible for logging failures on the network level. Network related alarms can be grouped in order to provide other components with pre processed information. It maintains a log of received alarms and updates their status until they are closed. On reception of a new alarm, a trouble ticket object is created, and a notification is emitted. This can be forwarded to a special interface that has been prototyped for the commercial ARS trouble ticketing system.
- TMN-CORBA Gateway allows a CORBA based system (i.e. TTRS) to access the TMN TTS for the purpose of exchanging trouble ticket information.

5.3 THE MUSIC SHOP SERVICE

The MusicShop service has been developed and implemented as a sample e-commerce service that is integrated in the TINA platform. It provides customers with a multimedia store and retrieval system using current service access technology (Web) and middleware platforms (CORBA, TINA; Oracle Application Server (OAS), Oracle 8 RDBMS [oracle]). The OAS delivers a scaleable, reliable and manageable platform for hosting shared network applications, such as data-driven multimedia content and interactive information. The MusicShop service allows users to store and retrieve arbitrary information types, ranging from plain text and pictures to audio and/or video (e.g. biographies of musicians, CD covers, pieces of music or music videos). It makes use of a Web-based interface and one or more database management systems for the storage of both management data as well as multimedia data.

The MusicShop system has been integrated into the Y.TSP platform. This implies that as well as being structured according to TINA principles, the MusicShop service has been integrated to work with the surrounding Y.TSP components, for example, it can be started via TINA access sessions and sends user accounting information to the TINA accounting component when it is used.

5.3.1 MUSICSHOP USER INTERFACES

The interfaces that have been identified within the MusicShop service are linked to the various roles played by external humans and components. Three roles are defined:

- The Service Provider Administrator operates the MusicShop, handles customer administration, and is concerned with contracts, trouble resolution and accounting.
- The Customer Service Administrator is concerned with subscription, trouble handling, accounting, billing, and user administration from the customer point of view.
- The end users use the MusicShop service, e.g. the multimedia data is uploaded by musicians to the MusicShop and downloaded by consumers from it.

To support the various roles, the MusicShop service offers a separate user interface for each role as a TINA user application. This interface provides each role holder with the specific core service and management functionality required by the role. The interfaces are provided either via HTML pages or Java applets and can be used in any Java capable Web browser. The documents are retrieved and uploaded using the HTTP protocol; some management functionality uses the CORBA IIOP protocol. Users are granted different access permissions according to whether they are musicians who can upload, consumers who can download or administrators who maintain the service.

Bilateral authentication between customer and service provider and the provisioning of authentication, access control and encryption mechanisms are essential requirements for a commercial service in the envisaged multi-provider environment. These are being supported by the secure access session provided by the TINA platform and its integration with the HTTP service by the use of cookies and SSL.

MusicShop Provider Administrator Interface: This interface supports the tasks that need to be performed by the service provider administrator. The main tasks of service provider administration for subscription management and service deployment are:

- Management of customer related information concerning customer administration (create, modify and delete customer), customer user administration (create, modify, delete user), and user group administration (create, modify, delete subscriber assignment group (SAG)).
- Service subscription management, which includes contract administration (create, modify, delete), subscription profile definition for a customer, and service profile administration for user groups (create, modify, delete SAG Service Profile). Service type management (deployment and withdrawal of services in the provider's domain) is not a part of the subscription component but is supported as a service provider administration task.
- Accounting, billing and tariff management.
- MusicShop management
- Trouble ticket handling and problem resolution.

MusicShop Customer Service Administrator Interface: This interface supports the customer service administration tasks. These are:

- Management of subscription related information concerning administration of users (create, modify, delete user) and user groups (create, modify, delete SLA: SAG).
- Service subscription management, which includes on-line subscription and service profile administration for user groups (modify SAG Service Profile).
- Accounting, get bill.
- Trouble ticket handling.

MusicShop Service User Interface: The main operations available to the user through this interface are, according to the permissions granted:

- Folder management related functions, i.e., create, delete, rename, and move.
- Multimedia information management, including upload, download, delete, rename, and move.

5.3.2 MUSICSHOP COMPONENT DESCRIPTION

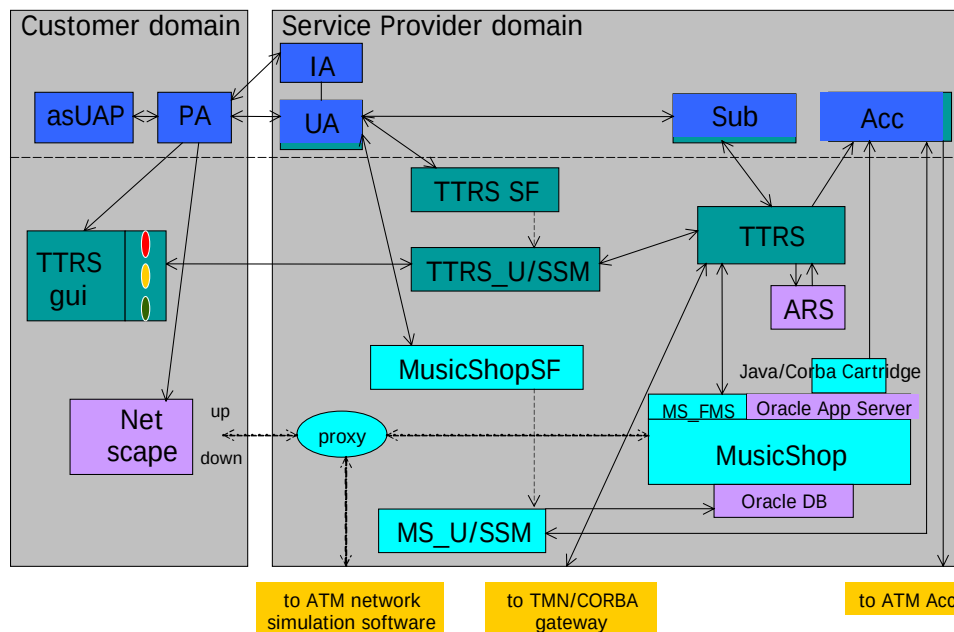


Figure 5-3: The MusicShop Component Model

Figure 5-3 depicts the components of the TINA service platform Y.TSP and of the MusicShop system (the light gray filled boxes) being used in the FlowThru system trial. The platform includes the components for subscription (Sub), accounting (Acc) and the basic functional components responsible for the access session (asUAP, PA, IA, UA) and the start-up and shutdown of services (MusicShop SF). The MusicShop service components comprise proxy, Netscape Browser in which the GUI runs, MS_U/SSM and the MusicShop main component, which runs on the OAS.

- The Proxy server acts as a monitoring tool, which physically stops the connection between client and MusicShop server when there is a simulated communication failure within the network domain.
- The MS FMS handles incoming and outgoing trouble tickets associated with failures within the MusicShop provider domain. It is also responsible for automatic trouble resolution as well as trouble detection where possible.
- The Java/CORBA Cartridge is located in the OAS and delivers accounting information to the TINA accounting system.

During the subscription phase, customers (in the musician and consumer role) are registered for the MusicShop service with a defined SLA including the QoS expected from the MusicShop service provisioning and the tariff to be applied in the case of both normal and degraded situations. The MusicShop therefore supports the service provider in obtaining the necessary information to bill customers as well as to offer the service with a meaningful performance as defined in the SLAs. MusicShop customers can adapt the service parameters to satisfy their requirements (type of access depending on terminal capabilities) as well as to monitor the total cost of service usage. To fulfil those needs, the MusicShop, as integrated in the TINA platform, offers a rich set of management functions, including subscription, accounting, customer administration, service provider administration and service configuration. The integration of service management and core service functionality is transparent to the users, thus emphasising the equal importance of both areas.

5.4 THE MANAGEMENT SERVICES

The MusicShop service is a multimedia service spanning many organisational and technology domains and is thus typical of the type of telecommunications service being introduced into the open service market. In order to provide adequate customer care and service quality assurance for the

MusicShop service, the needs of the various actors in the multi-provider environment have to be considered. A multi-domain trouble management support system should be able to enhance the functionality of established customer care centres and of in-house trouble ticketing systems, extending the current manually operated system with an automated exchange of trouble tickets (TTs) via standardised interfaces. This would require embedding legacy trouble ticketing systems such as the Remedy ARS product, as shown in Figure 5-4. This figure depicts the general principles of multi-domain trouble management, a subset of which has been implemented for the Service Quality Assurance System.

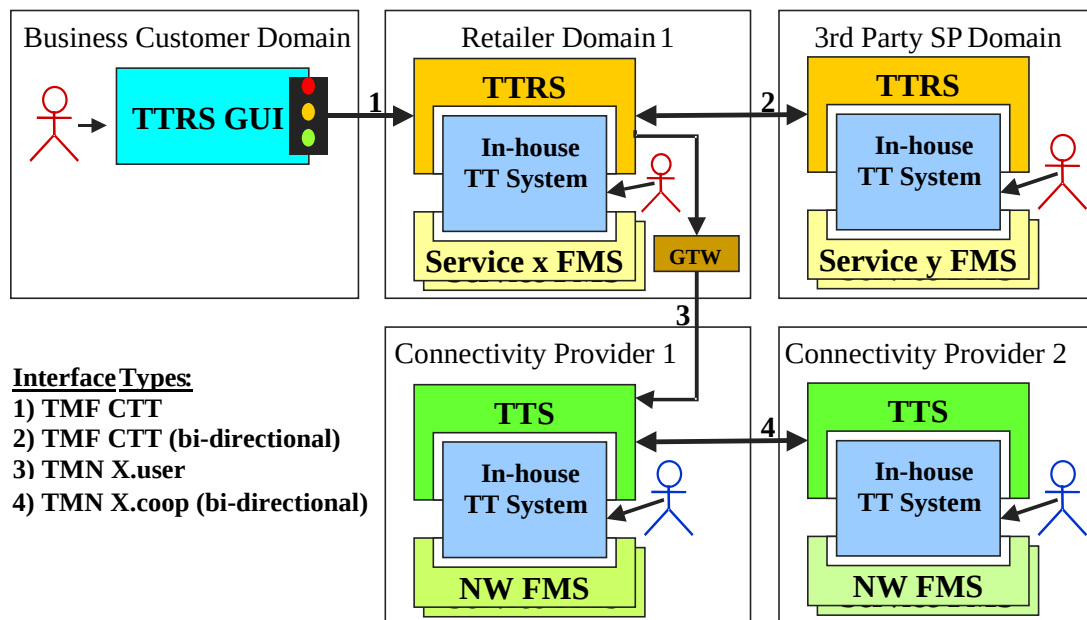


Figure 5-4: Trouble Management in a Multi-Provider Environment

For connectivity providers a TMN based solution has been selected. The same principle of embedding the legacy in-house TT-System can be used here too. To exchange TTs between the customer and the connectivity provider domains (i/f type 3) or between connectivity provider domains, standardised interfaces (i/f type 4) based on GDMO / CMIP specifications of EURESCOM project P612 have been used. The exchange of TTs between connectivity provider and value-added service provider domains requires a mapping between TMN [m3010] and CORBA [corba] technologies. This can be achieved by a gateway which maps CMIS to IDL interfaces based, for example, on the JIDM standards [jldm95], [jldm97].

The rest of this section presents the two trouble management systems developed for the FlowThru Service Quality Assurance System, one based on CORBA and TINA principles for the service level, and the other using TMN for the network level.

5.4.1 THE SERVICE LEVEL TINA TROUBLE REPORT SYSTEM (TTRS)

The service-level trouble management service, implemented by the TTRS, is offered to customers as a TINA service. It makes use of TINA service architecture principles and enhances the current definitions and reference point specifications to support distributed trouble management business processes. The TTRS also makes use of enhanced definitions of reference points between retailers and third-party service providers [school]. The TTRS component may therefore be considered as an additional component of the TINA service architecture for trouble management and for the support of service quality assurance and customer care. It is designed to exchange trouble reports between the customer, retailer, third-party service provider, and connectivity provider domains. It also supports SLA handling and the initiating of discounts in case of SLA violations.

The TTRS, which incorporates an in-house trouble ticketing system based on the Remedy Action Request System, was developed as part of the Y.TSP and has incorporated TINA principles in its

design. Specifically, the TTRS has been designed with service session user applications, user service session managers and service session managers in mind. However, the TTRS is expected to be a service that exists as part of Y.TSP itself. Hence, the service will be started automatically within the user access session to enable problem notifications to be delivered to end users who have access sessions established. It is being implemented using Java to enable easy installation and porting over various operating systems, such as UNIX and Windows/NT.

The TTRS component is the central component in supporting TINA based trouble management and ensuring service quality of both the TINA based services and the connectivity upon which those services run. Consequently, the TTRS component needs to interwork with numerous components as illustrated in Figure 5-5.

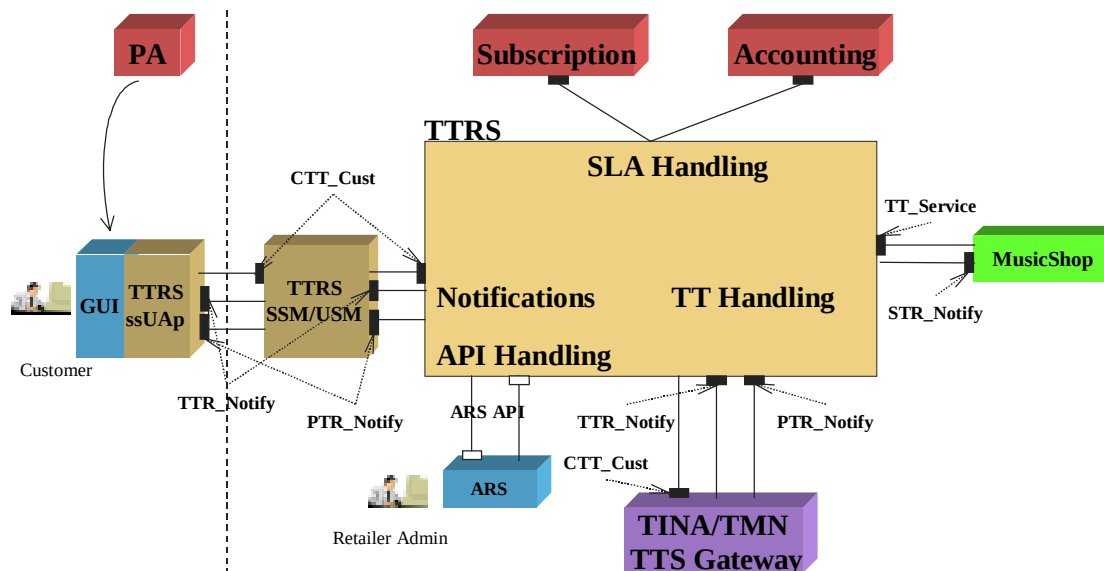


Figure 5-5: TTRS Interfaces in the FlowThru Framework

The following interfaces identified in the context of the TINA Trouble Reporting System reflect the TINA environment as well as the integration between the service level management and the network level management for trouble identification and SLA management.

- The CTT_Cust interface is used by a customer to create, modify, track the status of, view, verify, delete, and cancel trouble tickets; to grant an authorisation for repair activities or to escalate a trouble ticket. The service provider exports this interface so that its customers can manage the lifecycle of trouble tickets effectively.
- The TTR_Notify interface is used by a customer to receive notifications from the service provider about trouble tickets, for example that a trouble ticket has been created due to a problem that another user, the service or connectivity provider or some remote service or connectivity provider has identified. This interface also allows customers to be notified about modifications to existing trouble tickets, status updates of trouble tickets, deletion notifications of existing trouble tickets and cancellation notifications of trouble tickets.
- The PTR_Notify interface is used by customers to receive notifications from service providers about trouble tickets associated with scheduled maintenance.
- The IDL for these interfaces is provided in the TM Forum interface specification document [tmf97]. The TTRS component provides a CORBA based wrapper around the Remedy ARS component so that it can be incorporated into the Service Quality Assurance System using open CORBA interfaces.
- The TT_Service and STR_Notify interfaces between the TTRS and MusicShop components correspond to a subset of the functionality of the CTT_Cust, PTR_Notify and TTR_Notify interfaces. More precisely, the TT_Service interface is used by the MusicShop service to inform

the TTRS of intended maintenance periods for the service and to request that trouble tickets are created for problems identified by the MusicShop service trouble management system (and their states changed when necessary, etc.). The STR_Notify interface is used for receiving notifications from the TTRS component related to trouble tickets, e.g. the TTRS might inform the MusicShop service trouble management system that it has created a trouble ticket when the problem lies with the MusicShop service itself.

The TTRS also possesses interfaces to the TINA accounting and subscription components. The interface to the accounting component is used primarily for issuing discounts to users who have incurred some form of SLA violation for one of the services they use. The interface to the subscription component is used for several purposes. First, it is used for querying whether users who complain about services are actually subscribers to those services. If so, the subscription component returns subscription information about the user and any other customers affected. This includes both the normal subscription information and service properties that they might have as well as the SLA that they agreed to when subscribing to those services. These are subsequently used by the TTRS when deciding whether a SLA violation has taken place and if so, how much the discount should be.

The information model used by the TTRS is summarised in the OMT class diagram shown in Figure 5-6.

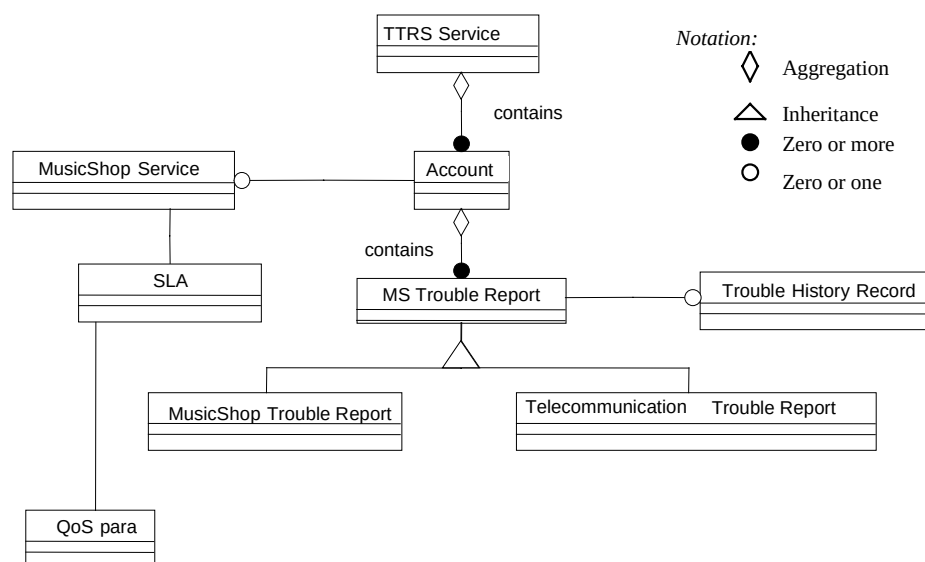


Figure 5-6: OMT Class Diagram for the TTRS Information Model

- **Account** - represents an account for a customer of the MusicShop service. This object captures information about the physical location and the connectivity provider SAP (Service Access Point).
- **TTRS Service** - is an object class representing the TINA Trouble Resolution service. Only one instance of this class will be instantiated.
- **MusicShopTroubleReport** - is inherited from the troubleReport object class. The providerTroubleReport is used to keep information about an existing problem in the MusicShop service for a particular account.
- **telecommunicationTroubleReport** - is inherited from the troubleReport object class. The telecommunicationTroubleReport is used to keep information about a service or resource related trouble report.
- **MusicShop Service** - represents the MusicShop service. This object keeps information about the type of service. It is assumed that other services would exist in the future.
- **SLA** - represents the Service Level Agreement associated with a particular MusicShop Service account. Information such as gold or silver service agreement is represented in this object and it

thus allows the TTRS to verify whether the MusicShop service is being provided at the quality agreed in the contract.

- QoS parameters- represents the set of parameters that constitute the SLA. These parameters depend on the type of contracted service (gold, silver) or can be based on customer-specific requirements (availability, response time, audio quality, disk space, etc).

5.4.2 THE NETWORK LEVEL TROUBLE TICKETING SYSTEM (TTS)

At the connectivity provider level of the multi-domain trouble management system, the FlowThru Service Quality Assurance System has used the EURESCOM project P612 specifications. The TTS is therefore based on X.790 and is TMN compliant [covaci98]. The TTS position in the FlowThru framework is depicted in Figure 5-7.

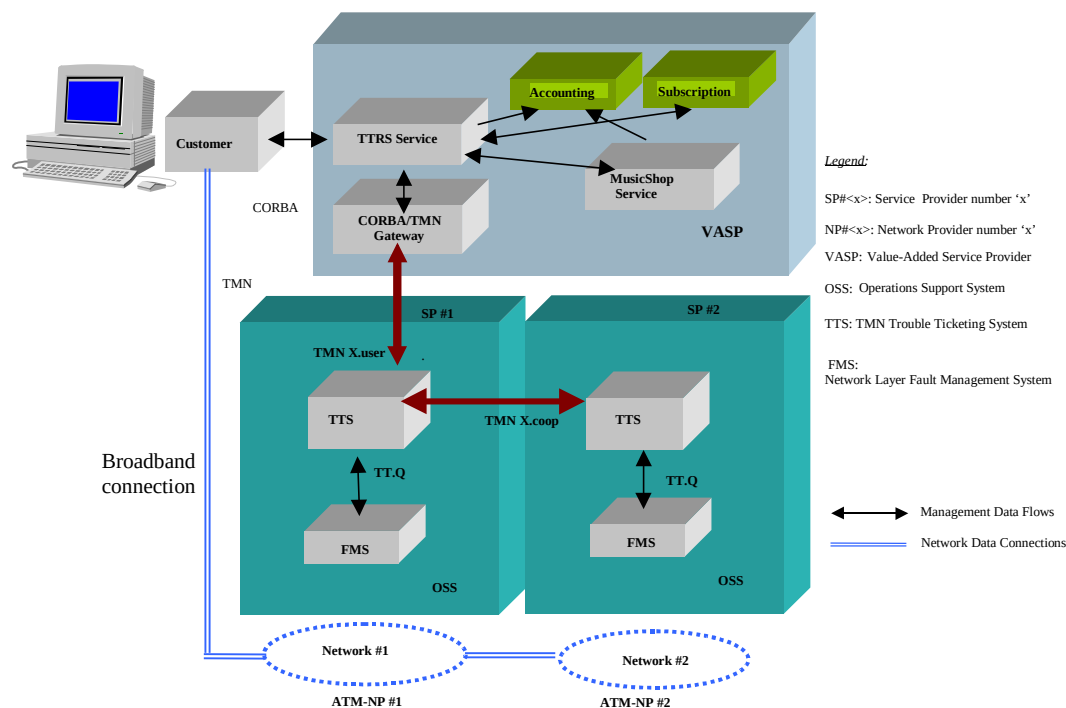


Figure 5-7: The TTS Position in the FlowThru Framework

At the network level, the important issue is interoperability between different management domains according to previously established co-operation policies. Therefore, the TTS system supports the following four interfaces:

- A TMN X.user interface allows the value-added service provider, here the MusicShop provider, to act as a customer to the connectivity provider for the trouble ticketing service requirements (via a CMIP/CORBA gateway) [covaci97].
- A TMN X.coop interface allows two connectivity providers to co-operate in exchanging trouble tickets related to the (ATM) connections.
- A TT.Q interface allows a connectivity provider call centre to dispatch a trouble report to one of its regional areas for resolution. This dispatching is operationally identical to a referral of a trouble report between two co-operating network operators. The only difference is that this dispatching may be executed in a single direction. From the point of view of the TTS agent it is identical with the one used for the X.coop interface – apart from the configuration parameters.
- In addition to these interfaces, a local administrator interface exists for carrying out parts of the trouble resolution process. Depending on the existing SLAs and the complexity of the legacy systems, the ratio between human and automatic activities involved in the trouble resolution process may vary.

Internally the TTS product is based on the Hewlett-Packard Open View Distributed Management platform supporting the OSI protocol stack for CMIP and the HP OV Managed Object Toolkit for the rapid generation/implementation of a TMN agent starting from its GDMO specification.

The managed object classes used by the TTS are defined in the X.790 information model [19]. The internal attributes of the information model have been profiled to meet the requirements for Trouble Ticketing within the FlowThru project. The information model used by the TTS for the maintenance process is described in an OMT class diagram as shown in Figure 5-8.

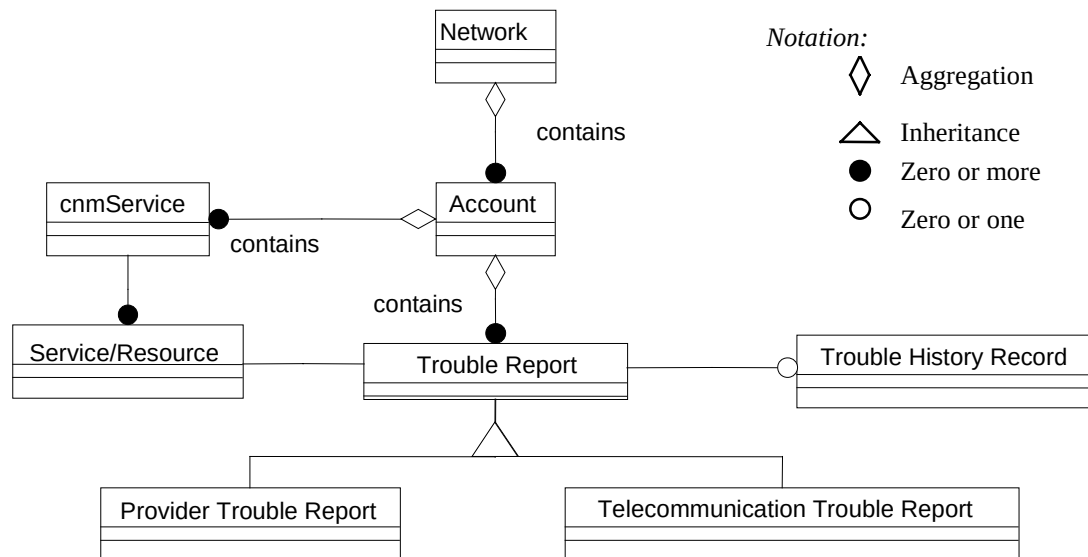


Figure 5-8: OMT Class Diagram for the TTS Information Model

The managed objects are:

- account - represents an account for a customer or a co-operating service provider. It holds all the client-specific data for this service.
- cnmService - is inherited from the service object class. The cnmService (Customer Network Management Service) object class represents the service offered by a service provider that a customer can complain about. Only one instance of this class for each account will be instantiated. This instance contains references to all the actual entities (MOs in trouble) to which the trouble ticket can refer.
- providerTroubleReport - is inherited from the troubleReport object class. The providerTroubleReport is used to inform the customer that planned maintenance affecting a specific service will be carried out.
- telecommunicationTroubleReport - is inherited from the troubleReport object class. The telecommunicationTroubleReport is used to keep information about a service or resource related problem.
- troubleHistoryRecord - contains archived information about a closed trouble report.
- service/resource - represents a set of services and resources concerned by the trouble report.

5.5 SCENARIOS

The Service Quality Assurance System was evaluated within FlowThru for its effectiveness in multi-domain trouble management. This covers the exchange of service level trouble reports and network level trouble tickets, event correlation, and partially automated trouble resolution. The trial scenarios focussed on the information flow between the components of the multi-domain trouble management system distributed among the various autonomous administrative domains of the multi-provider environment.

For the Service Quality Assurance System, customer subscription to the MusicShop and the required premium IP connectivity services and the necessary network and service configuration have already been undertaken in the fulfilment phase and are therefore already in place. The two services, MusicShop and the Premium IP connectivity service, are registered in the TINA subscription component along with the properties associated with them – including their SLAs. When troubles occur and are subsequently resolved, checks on the SLAs of the users affected by the trouble are made, and if violated, result in discounts being issued for the usage of that service.

Various trouble management scenarios are being demonstrated to highlight the integrated interactions between the different trouble management components of the Service Quality Assurance System, for example:

- A musician attempts to upload a piece of music to the MusicShop and identifies a connectivity problem. This problem affects many other users. The problem is resolved without requiring user verification and no SLA is violated for any user.
- A musician attempts to remove a previously stored piece of music from the MusicShop and identifies a service problem. The problem affects only this user. The problem is resolved and the user verifies that the service now works. The SLA is violated for this user.
- A consumer is downloading a piece of music and detects a connectivity problem with the connectivity provider. Simultaneously, connectivity provider 2 identifies the same connectivity problem. The collision is detected inside connectivity provider 1, ensuring that only one TR exists for a given problem in each administrative domain. Connectivity provider 2 resolves the problem and users are not requested to verify. No service level violations take place.

These and other scenarios were used to test and demonstrate the Service Quality Assurance System in a variety of ways. The system was thus evaluated and the results will be applied in further development of the system.

5.6 CONCLUSIONS

We presented an integrated framework for multimedia service quality assurance in an open service market. The Service Quality Assurance System has been developed and demonstrated together with the fulfilment and accounting parts of the FlowThru multi-domain management system to show that advanced multimedia services, such as the MusicShop service, can be supported by an integrated management system spanning a variety of organisational and technology domains.

The system presented in this chapter has been specified using TMN and TINA concepts and structured according to TM Forum business processes. Such a merging of concepts into one system has many advantages. It enables both network and service layer management to be integrated in a system that can span multiple domains and is thus capable of managing end-to-end services. It can provide automated processes to enhance or replace those currently undertaken manually and by linking in seamlessly with subscription and accounting components it can ensure information “flow through” throughout the service management life-cycle. We have shown that it is possible to take “state of the art” developments and provide a solution that can aid both service providers and customers in their pursuit of end-to-end automated trouble management for the multimedia services of today and tomorrow.

6 ACCOUNTING SYSTEM

Effective telecommunications management relies critically on the integration of different management functions. In a liberalised telecommunications market, this integration has to occur over multiple organisational and technological domains. Existing management applications are often designed according to a specific architecture and implemented to be deployed on a specific technology platform. On the other hand, in any realistic scenario, large-scale management integration must rely instead on the effective reuse of well-understood patterns of design and the reuse of specifications and component implementations.

The trial business scenario selected to support the goals of FlowThru considers the following actors from the TINA business model: the Consumer, the Service Retailer, the 3rd Party Service Provider and the Connectivity Provider. The *3rd Party Service Provider* offers a set TINA-conformant managed multimedia services to the *Consumers*. Both interactive and information retrieval service types are supported. The *Service Retailer* offers secure customer access to these 3rd party services as well as to its own subscription management and accounting management function. The *Connectivity Provider* manages an ATM-based transport network and offers an end-to-end ATM connectivity service. The *3rd Party Service Provider* does not provide wide area connectivity directly, but makes use of whatever ATM services are offered by the various Connectivity Providers that serve areas in which the Retailer's Consumers are located. This is a typical scenario that may evolve in a deregulated European telecommunications market. Workflow management techniques are used to co-ordinate the activities and information flow between the Retailer's internal processes and those of its suppliers. In such a scenario the automation of business processes is essential to the competitiveness of the Service Retailer, and key to the integration with the business process of its customers - the Consumers - and its suppliers - the 3rd Party Service Providers and the Connectivity Providers.

This chapter examines in more detail the design of the accounting Trial Business System.

6.1 THE ACCOUNTING TRIAL – BUSINESS MODEL

6.1.1 BUSINESS SYSTEM

The business environment of the Accounting system is first modelled in a diagram identifying the specific business actors, the business roles and responsibilities they are assigned as well as the relationships between those business roles. An overview figure is presented in Figure 6-1. On the highest level of abstraction the Accounting system addresses three business actors: Service Customer, Service Provider and Network Provider. The *Service Customer* contains the business roles *Customer Administrator* and *User* (or End User). The *Service Provider* contains the business roles *Service Retailer*, *Service Operator*, *Service Subscription Manager* and *Service Accounting Manager*. The Service Subscription Manager role is responsible for contract negotiation between the Customer Administrator and the Service Retailer, and for the mapping of subscriptions onto accounts managed by the Service Accounting Manager role. The Service Retailer role regulates the access to services by Users playing different roles in accordance with the subscription contracts managed by the Service Subscription Manager. The Service Operator role provides a User with the usage facilities of the service and provides the Service Accounting Manager role with service usage statistics on which to base charging information. The Service Accounting Manager role provides the Service Customer with service usage statistics and charges based on these statistics. In addition, it provides billing information to the Service Customer in the Customer Administrator role. Analogously, the *Network Provider* contains the business roles *Network Retailer*, *Network Operator*, *Network Subscription Manager* and *Network Accounting Manager*. These roles are indeed analogous to the roles in the Service Provider domain, but it ought to be noted that the Network Provider provides a pure connectivity service, with the Service Provider acting as the customer of this service.

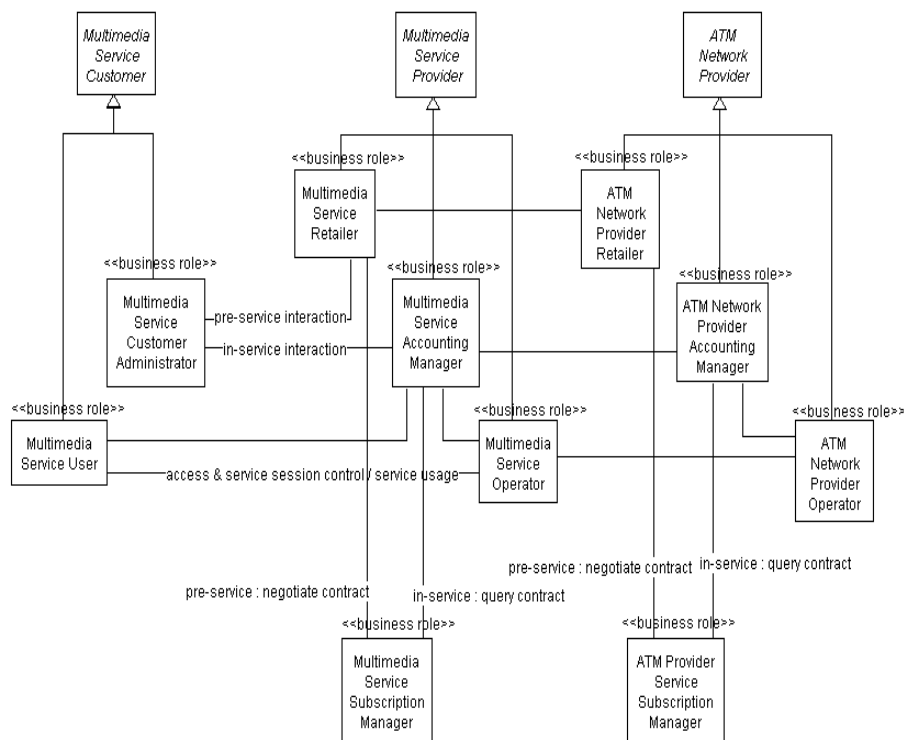


Figure 6-1: Accounting Trial – Business Roles

6.1.2 BUSINESS SCENARIOS

Subsequently, this business model is used as the basis to define the overall functional model, consisting of the system's external boundaries and functional requirements with respect to the business scenarios where it is to be used. These requirements are outlined within the descriptions of the use-cases that are supported by each system – each use-case corresponding to a business scenario. Detailed descriptions of each use-case are provided within the FlowThru deliverable, but this chapter only includes one diagram as a summary in Figure 6-2.

The *getBill* use-case reflects the Customer Administrator requesting the Accounting system to generate a bill for a certain billing period. The bill might also be automatically issued at the end of a billing period as negotiated and defined in the Customer's contract. The Accounting system generates the bill based on charging information from records of accountable events generated by the service, charging information from the Network Provider and subscription information. The *listSessions* use-case represents the Customer Administrator requesting a list of the currently active accounting management service sessions from the Accounting system. These sessions mirror the lifecycle of multimedia service sessions, but exist only to collect and collate usage data for them. The *getSessionCharges* and *getUserCharges* use-cases represent the Service User or Customer Administrator requesting the charges based on the usage data for one particular service session and one particular service user, respectively. When a Service User participates in a particular service session, this generates accountable events for each event defined at subscription. This scenario is captured by the *generateAccountableEvents* use-case. The provisioning of the service itself is covered by the *provideMultimediaService* –by the Service Provider- and *provideNetworkService* –by the Network Provider- use-cases.

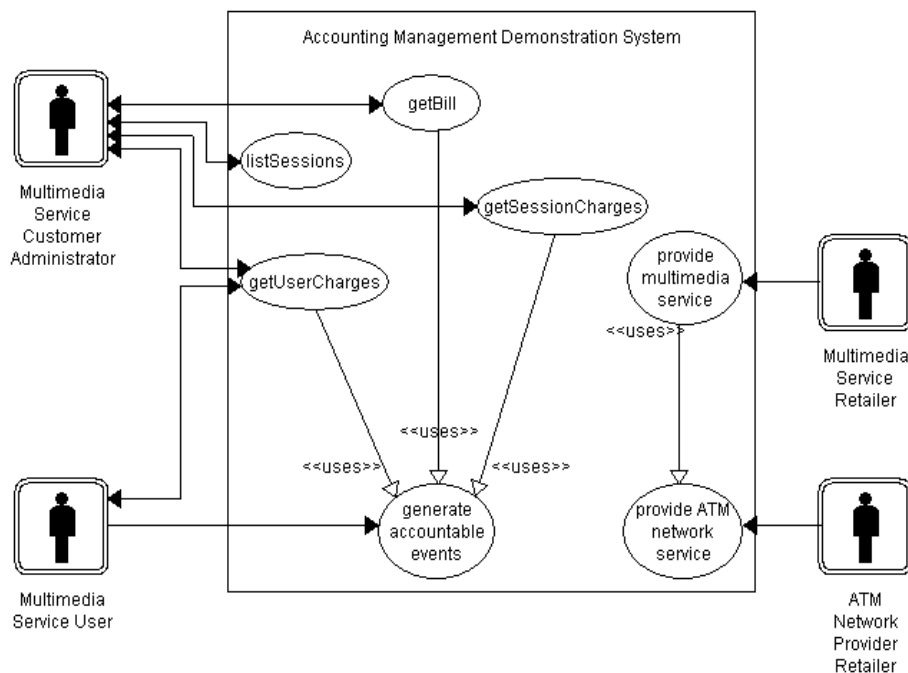


Figure 6-2: Accounting Trial - Use-case Summary

6.1.3 MAPPING OF TM FORUM BUSINESS PROCESSES TO TINA BUSINESS MODEL

The accounting system is a realisation of the TM Forum Billing business process, which is itself comprised of the following processes: *Invoicing/Collection*, *Rating/Discounting* and *Network Data Management*. Figure 6-3 provides an overview of the TM Forum business processes and highlights the processes addressed by the Accounting system. Since the components that FlowThru re-engineers and integrates for accounting were originally built using TINA architectural and modelling concepts, a mapping of the above processes onto TINA business roles was required. Invoicing/Collection were seen as the responsibility of the Retailer, while Rating/Discounting were seen as the responsibility of the Broker, Retailer and Third-Party Service Provider, enabling each to apply its own discounts and rates. Network Data Management could be considered to be the responsibility of the Connectivity provider. However, certain TINA accounting concepts cannot be mapped directly onto the TM Forum model, because of the separation between service level and network level accounting in the TINA model. For this reason, 'Network Data Management' is renamed to 'Usage/Performance Data Collection' to better signify its role in both the service and network layers. This results in mapping the Usage/Performance Data Collection to the responsibility of both the 3rd Party Service Provider and the Connectivity Provider.

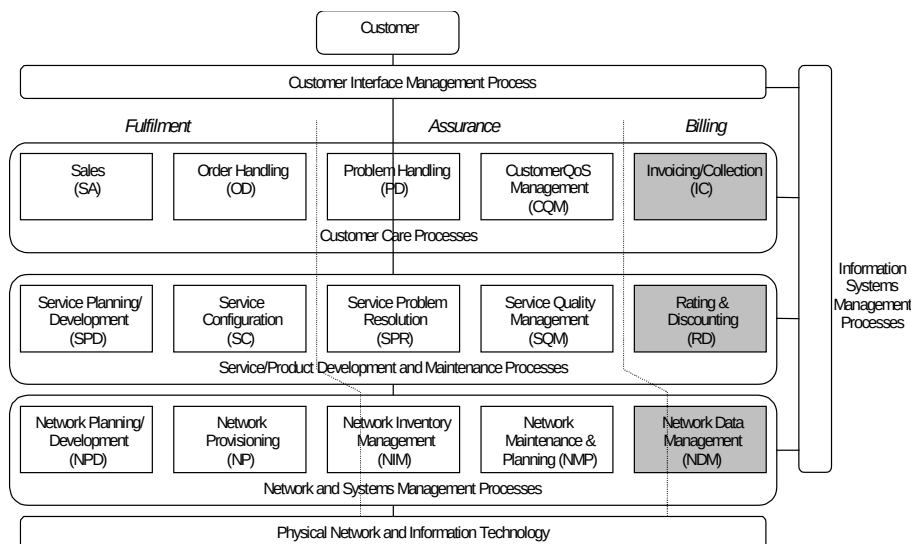


Figure 6-3: Accounting Trial – Business Processes

6.2 THE ACCOUNTING TRIAL - SYSTEM OVERVIEW

Figure 6-4 shows the boundary objects of all the components comprising the Accounting system, their interfaces and their interactions. Components should be interpreted here as the units of reuse. The components can themselves be decomposed into finer grained computational or engineering objects. The overview only highlights the boundary objects within the components, their interfaces and their interactions. The system consists of six separate components, covering different aspects of service and network control and management. The *Access Session* component is responsible for allowing access to subscribed services by establishing a secure context for interactions between the Consumer and the Retailer. The *Service Session* component is responsible for the actual usage of a specific service by a Consumer. This component is specialised to cater for two different service paradigms: a multimedia information retrieval service, the ‘Digital Library’, and an interactive multimedia service, the ‘Desktop Video Audio Conference’.

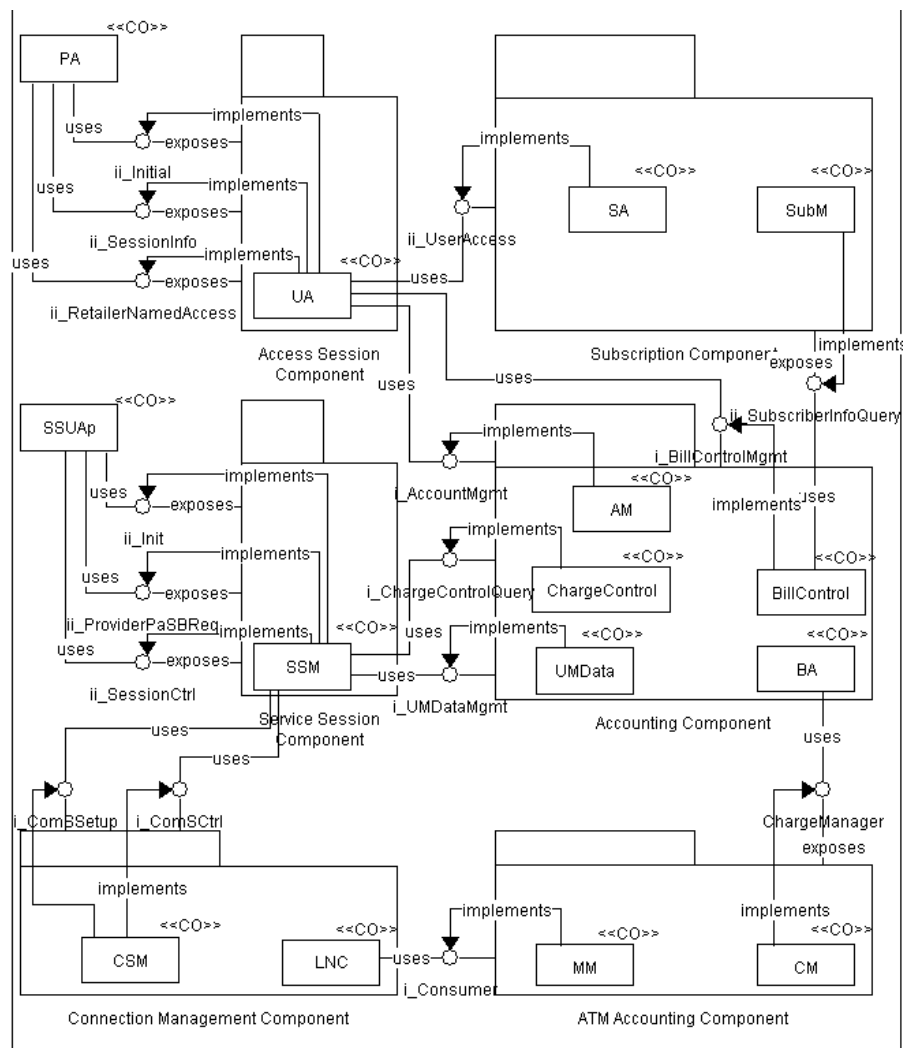


Figure 6-4: Accounting Business System Components and Interactions

The *Subscription* component is responsible for the management of subscription contracts between the Consumer and the Retailer and ensuring access to services is in accordance with these contracts. The *Accounting* component is responsible for collecting and collating service level usage data, generating charges based on this usage data and combining these charges with those collected at network level. Furthermore, this component generates the bill for the Consumer, based on guidelines laid down in the contracts controlled by the Subscription component. The *ATM Accounting* component is responsible for collecting network level usage data, generating charges based on this data and forwarding these charges to the Accounting component. The *Connection Management* component is responsible for providing ATM connectivity between end-user equipment and for generating network level usage data for the ATM Accounting component. It should be noted that the difference between the Accounting and ATM Accounting components is that they operate on different management levels. The Accounting component operates on the service level, while the ATM Accounting operates at the network level. As already stated, these components were identified as a result of the TINA business model separation between service and connectivity provisioning. From a business process perspective, they both map onto the Usage/Performance Data Collection TM Forum business process.

The components outlined above are envisaged to co-operate in the following fashion. In the 'pre-service' phase, a Consumer subscribes to a specific service offered via a service Retailer, in this case either the 'Digital Library' or the 'Desktop Video Audio Conference'. A Consumer who engages in a service subscription is specialised into a Customer Administrator. The Customer Administrator is responsible for subscription administration for one or more End Users, who are themselves another

specialisation of the Consumer role. When a new subscription is created, the SubM (Subscriber Manager – boundary object within the Subscription component) creates a new contract and calls on the AM (Account Manager – boundary object within the Accounting component) to create a new account. During the ‘in-service’ phase, an End User can gain access to his subscribed services by establishing an access session with the Retailer through the UA (User Agent - boundary object within the Access Session component), which represents the End User in the Retailer domain. Subsequently, the End-User can create new service sessions and/or join existing service sessions. The service functionality is implemented by the SSM (Service Session Manager – boundary object in the Service Session component). When the usage of the service implies stream connectivity between different End User equipment, a connection request is generated by the SSM towards the CSM (Communication Session Manager – boundary object within the Connection Management Component), resulting in the creation of communication session context shared between SSM and CSM. During the lifetime of the connection, the MM (Metering Manager – boundary object within the ATM Accounting component) receives, collects and collates accountable events generated by the LNC (Layer Network Co-ordinator – boundary object within the Connection Management Component). Meanwhile, at the service level, the UMDa (Usage Metering Data – boundary object within the Accounting component) receives, collects and collates accountable events (usage data) generated the SSM. The End-User can obtain in-session charges for service usage up to a certain time, resulting in the SSM retrieving this information from the CC (Charge Control – boundary object within the Accounting component). The Customer Administrator can request the charges for any specific service session through the same boundary object of the Accounting component. In the ‘post-service’ phase, or at the end of a pre-defined period stipulated within the subscription contract, the Customer Administrator receives a bill from the BC (Bill Control – boundary object within the Accounting component). Generating this bill requires the generation of charges by the CC object, based on information contained in the contract managed by the SubM object. These charges are to be correlated by the BA (Billing Aggregation – boundary object within the Accounting component) with the charges collected at network level by the CM (Charge Manager – boundary object within the ATM Accounting Component).

Notice that the FlowThru accounting trial realisation supports all business roles identified in Figure 6-1, except for the Retailer and Subscription Manager roles within the Network Provider domain.

6.3 THE ACCOUNTING SYSTEM – COMPONENTS

This section lists the major functionalities provided by each of the components and points to the ACTS project from which the component originates. It also lists the adjustments required to comply with the identified trial use-cases. For extensive analysis models and use-case descriptions of the individual components, the reader is referred to [stathopoulos].

The *Accounting* component is based on the TINA accounting management specifications as defined in [kristiansen]. The component was originally developed within the Prospect project. Some extensions of the Prospect system were deemed necessary by the presence of network level accounting in the Accounting trial. The accounting component contains functionality that covers all areas of the TM Forum “Billing” high-level business process. It maps certain TM Forum processes onto TINA business roles, and in some cases concentrates on specific low-level activities within the processes. The component implements the invoicing part (i.e. production of the final bill) of TM Forum “Invoicing and Collection” process, but not the collection part of it (i.e. payment of the bill). It also offers on-line billing functionality, enabling users and administrators to see the charges accrued so far in a particular service session.

The *Access Session* component is aligned with the latest TINA Ret RP [farley] and Service Component Specifications [abarca]. The component was originally developed within the VITAL project. The component supports the creation of a secure context between Consumer and Retailer, and the access from the consumer to the subscribed services. A Consumer needs to have an access session established before he can engage in any service. The functionality of the component is extended

within FlowThru to allow the Consumer to retrieve his bill related to his service usage from the Retailer.

The *Service Session* component is also aligned with the latest TINA Ret RP and Service Component Specifications and was also developed within the VITAL project. This component supports the participation of one or more users in a service session. The component aims to support service common feature-sets, which can be used in most of the services. These feature sets contain various functions such as: joining a service session, inviting parties in the service session, establishing stream connectivity requirements, issuing voting, etc. These functions are functionally grouped into the feature sets, such as 'basic', 'multiparty', 'stream-binding', 'voting', etc. In the FlowThru project two services will be used to test the service session component: the "Digital Library" and the "Digital Video Audio Conference". Each service inherits from the service common feature sets it requires, and adds its own service specific features to it, thus defining the new service. The component is extended within FlowThru to generate accountable events for the Accounting component and to allow the Consumer to retrieve on-line billing information about an ongoing service session.

The *Subscription* component is based on the TINA subscription management specifications as defined in [kristiansen]. The component is a slimmed-down version of the Subscription management component developed by Prospect, but comprises all management functions needed in order to define service offerings, administer customers and users, and manage the details of service provisioning. For instance, the component allows for authorisation and barring of users' access to specific services. The component implementation is tuned specifically with respect to the actors and use-cases identified within the Accounting trial scenarios.

The *Connection Management* component is based on the TINA Network Resource Architecture [steegmans97a] and TINA Network Resource Information Model [steegmans97b] specifications. The component was originally developed within the ReTINA project. The component provides procedures to set-up, control and release connections in the underlying telecommunications network. Although it offers a technology independent connectivity service to the *Service Session* component, it is internally specialised to control ATM-based networks. The component is extended within FlowThru to generate accountable events for the ATM Accounting component.

The *ATM Accounting* component is designed to capture ATM based charging schemes and apply these schemes to usage data gathered from ATM connection metering. This component is used in both the Assurance and Accounting business systems. The system is capable of producing individual charges, or amalgamated charges (bills) based on this usage data, which are stored in a relational database, or produced in simple report form. The component is designed and implemented specifically for the FlowThru project; it is not based on an existing component implementation.

6.4 CONCLUSIONS

This chapter provides an overview of the FlowThru trial system, by presenting the modelling at business system level as well as at component level. It exhibits the application of the management system development methodology guidelines as outlined by the project in deliverable [wade98]. In essence, this chapter provides an example of how solutions to telecommunication management business problems can be analysed in a way that facilitates their construction from reusable components. The process results in a real-world example of the application of FlowThru's guidelines for systems development, component re-use and integration. It triggers the assessment of the effectiveness of such guidelines in addressing the requirements of management system development. In addition to re-usability and integration of existing components, the ability of the assembled business systems to support trial scenarios that satisfy real telecommunication management requirements is emphasised. The chapter also emphasises the application of the guidelines to realise the accounting and billing business system trial. The mapping of relevant TM Forum business process onto TINA business and computational models turned out to be relatively straightforward, although the separation between service and connectivity provider within the TINA model forced some adaptations to the TM Forum model. The modelling, analysis and specifications produced both at system and component level were passed onto the project's system implementation workgroup, where

the work was undertaken with respect to the detailed interface specification, adaptation and integration of the identified components which resulted in the realisation of the Trial Business System.

7 ACKNOWLEDGEMENTS

The work presented in the paper was conducted with partial funding from the European Commission under the FlowThru project (AC335). The views expressed here are not necessarily those of the FlowThru consortium. The work presented here would not have been possible without the hard work of and excellent co-operation between all the people involved the FlowThru project.

8 REFERENCES

- [abarca] C. Abarca et al, TINA Service Component Specification 1.0b, TINA-C, <http://www.tinac.com>, 1997.
- [ad/97-08-03] UML Summary, v1.1, ad/97-08-03, OMG, Aug 1997
- [agoulmine] N. Agoulmine, D. Dragan, T. Gringel, J. Hall, E. Rosa, and M. Tschichholz, Trouble Management for Multimedia Services in Multi-Provider Environments
- [alonso] G Alonso, D Agrawal, A El Abbadi, C Mohan, "Functionality and Limitations of Current Workflow Management Systems", IEEE Expert, Vol 12, No. 5, Spet/Oct 1997
- [bjerring] L Bjerring, C Grabowski, S Dittman, R Lund, L Bo Sorensen, S Rasmussen, "Experience in developing multi-technology TMN Systems", NOMS '98, New Orleans, USA
- [booch] G. Booch, J. Rumbaugh and I. Jacobson, Unified Modelling Language for Object-Oriented Development, Rational Software Corporation, 1997
- [castellani] Zippin Project Homepages, Stefania Castellani, Xerox Grenoble, <http://www.rxrc.xerox.com/research/ct/prototypes/aippin/home.html>
- [corba] The Common Object Request Broker: Architecture and Specification. Object Management Group, Framingham, MA, Version 2.2, February 1998.
- [covaci97] S. Covaci and D. Dragan, Customer Care Solutions: Interoperable Trouble Ticketing Management Service. In A. Mullery, M. Besson, M. Campolargo, R. Gobbi, and R. Reed (eds.), Intelligence in Services and Networks. Technology for Co-operative Competition. Proceedings 4th International Conference on Intelligence in Services and Networks, IS&N'97, Cernobbio, Italy, May 27-29, 1997, Springer, Berlin, Lecture Notes in Computer Science 1238, pp. 255-262, 1997.
- [covaci98] S. Covaci, L. Marchisio, and D.J. Milham, Trouble Ticketing X Interfaces for International Private Leased Data Circuits and International Freephone Service. In Proceedings of the 6th NOMS '98 - Management for the New Millennium, IEEE Operations Center, Piscataway, NJ, pp. 342-353, 1998.
- [cox] Cox, B., Novobilski, A., Object-Oriented Programming : An Evolutionary Approach, Addison-Wesley, May 1991
- [das] S Das, K Kochut, J Miller, A Sheth, D Worah, "ORBWork: A Reliable Distributed CORBA-Based Workflow Enactment System for METEOR2", Technical Report #UGA-CA-TR-97-001, Department of Computer Science, University of Georgia, Feb. 1997

- [eckert] K.-P. Eckert, E. Moeller, P. Schoo, G. Schürmann, and M. Tschichholz, , Verteilte Objekt-technologie in der Telekommunikation, Praxis der Informationsverarbeitung und Kommunikation, Vol. 21, No. 3, pp. 149-154, 1998. Further information on Y.TSP is available at: <http://www.fokus.gmd.de/research/cc/platin/>
- [farley] P. Farley and R. Minetti (eds.), TINA-C Ret Reference Point Specifications, Version 1.0, TINA-C, January 1998.
- [gamma] Gamma, E., Helm, R., Johnson, R., Vlissides, J., Design Patterns : Elements of Reusable Object-Oriented Software, Addison-Wesley, October 1995
- [gavras] A. Gavras and W. Takita (eds), DPE Architecture, Version 2.0b, TINA-C, November 1997.
- [grasso] Antonietta Grasso, Jean-Luc Meunier, Xerox, WebFlow Homepage, <http://www.xrce.com/research/ct/prototypes/webflow/home.html>
- [hellemans] P. Hellemans, C. Redmond, K. Daenen, and D. Lewis, Accounting Management in a TINA-Based Service and Network Environment. In H. Zuidweg, M. Campolargo, J. Delgado, and A. Mullery (eds.), Intelligence in Services and Networks. Paving the Way for an Open Service Market. Proceedings 6th International Conference on Intelligence and Services in Networks, IS&N'99, Lecture Notes in Computer Science 1597, pp. 1324-1339, Barcelona, Spain, 1999.
- [jacobsen97] Software Reuse - Architecture, Process and Organisation for Business Success, Jacobsen, I., Griss, M., Jonsson, P., 0-201-92476-5, Addison-Wesley, 1997
- [jldm] <http://WWW.opengroup.org>
- [jldm95] Inter-domain Management: Preliminary CORBA/CMISE Interaction Translation Architecture, Joint Inter Domain Working Group, X/Open and Network Management Forum, 1995.
- [jldm97] Inter-domain Management: Specification Translation, Joint Inter Domain Working Group, X/Open and Network Management Forum, 1997.
- [jnsm97] Journal of Network and Systems Management, Special Issue on TINA, Vol.5, N°.4, December 1997.
- [kristiansen] L. Kristiansen (ed.), Service Architecture, Version 5.0, TINA-C, June 1997.
- [lewis95] Experiences in Multi-Domain Management Service Development, Lewis, D., Tiropanis, T., Bjerring, L.H., Hall, J., in [ISN95], pp174-184, Springer-Verlag, Oct 1995
- [lewis99a] A Development Framework for Open Management Systems, Lewis, D., Journal of Interoperable Communication Networks, vol. 2/1, pp11-30, Baltzer Science Publishers, Mar 1999
- [lewis99b] Modelling Management Components for Reuse Using UML, Lewis, D., Malbon, C., DaCruz, A., Proceedings of the 6th International Conference on Intelligence in Services and Networks, Barcelona, Spain, pp210-222, Springer-Verlag, Apr 1999
- [lewis99c] The Development of Integrated Inter and Intra Domain Management Services, Lewis, D., Wade, V., Bracht, R., Integrated Network Management VI: Proceedings of

- the Sixth IFIP/IEEE International Symposium on Integrated Network Management, Boston, USA, pp279-292, Addison-Wesley, May 1999
- [lewis99d] D. Lewis, C. Malbon, G. Pavlou, C. Stathopoulos, and E. J. Villoldo, Integrating Service and Network Management Components for Service Fulfilment. Proceedings 10th IFIP/IEEE International Workshop on Distributed Systems: Operations & Management DSOM'99, Zürich, Switzerland.
- [lewis99e] A Software Development Methodology for Service Management, D.Lewis, Proceedings of the Telecom'99 Forum, Geneva, December 1999.
- [m3010] Principles for a Telecommunications Management Network, ITU-T Recommendation M.3010, Geneva, May 1996.
- [m3100] Generic Network Information Model, ITU-T Recommendation M.3100, 1992.
- [matena] Matena, V., Hapner, M., Sun Microsystems: Enterprise JavaBeans, Version 1.0, 21 March 1998
- [mathan] Luc Mathan, "The TMN Program in Eurescom", Journal of Network and Systems Management", Vol.7, N°.2, pp.249-256, June 1999.
- [mulder] H. Mulder (ed.), TINA Business Model and Reference Points, Version 4.0, TINA-C, May 1997.
- [oracle] <http://www.oracle.com>
- [orbos/99-02-05] CORBA Components: Joint Revised Submission. OMG TC Document orbos/99-02-05, Draft, March 1999.
- [orbos/97-06-12] RFP for a CORBA Component Model. OMG TC Document orbos/97-06-12, July 1997.
- [p612] <http://www.eurescom.de/Public/Projects/p600-series/P612/P612.HTM>
- [p612-d2] Project P612 Deliverable 2, TMN X interface specifications for trouble ticketing, EURESCOM, June 1998.
<http://www.eurescom.de/Public/Publications/Projectresults/p600-series/612d2.htm>
- [pavlou97] Pavlou, G., Griffin, D., Realizing TMN-like Management Services in TINA, Journal of Network and System Management (JNSM), Special Issue on TINA, Vol. 5, No. 4, pp. 437-457, Plenum Publishing, 1997.
- [reform] Final REFORM System Specifications and Architecture, The REFORM Consortium, July 1998.
- [rusinkiewicz] M Rusinkiewicz, A Helal "Workflow Management Systems", published in Journal of Intelligent Information Systems, Vol 10, Number 2, March/April 1998
- [schoo] P. Schoo, C. Egelhaaf, T. Eckardt, N. Agoulmine, and M. Tschichholz, Modularization of TINA Reference Points for Information Networking, In H. Zuidweg, M. Campolargo, J. Delgado, and A. Mullery (eds.), Intelligence in Services and Networks. Paving the Way for an Open Service Market. Proceedings 6th International Conference on Intelligence and Services in Networks, IS&N'99, Lecture Notes in Computer Science 1597, pp. 446-458, Barcelona, Spain, 1999.

- [sheth96] Amit Sheth, "State of the Art of Commercial Technology and Research in Workflow Management" DARPA/ISO Workshop on collective Action Tools - April 10, 1996,
- [sheth97] Amit Sheth, "From Contemporary Workflow Process Automation to Adaptive and Dynamic Work Activity Coordination and Collaboration", University of Georgia SIGGROUP Bulletin, Vol. 18, No 3 (December 1997)
- [stark]"Ovum Evaluates: Workflow" - Heather Stark, Laurent Lachal, ISBN 1898972605, Ovum, 95.
- [stathopoulos] C. Stathopoulos et al, System Specifications and Models, FlowThru Deliverable D3, 1998.
- [steegmans97a] F. Steegmans (ed.), Network Resource Architecture, Version 3.0, TINA-C, February 1997.
- [steegmans97b] F. Steegmans et al, TINA Network Resource Information Model, TINA-C, 1997.
- [tarumi] H Tarumi, K Kida, Y Ishiguro, K Yoshifu, T Asakura, "WorkWeb Systems - Multi Workflow Management in a Multi Agent System", SIGGROUP 1997, Pheonix, Arixona, USA
- [telecom/98-06-15] Notification Service: Joint Revised Submission. OMG TC Document telecom/98-06-15, June 15 1998.
- [tmf-910] Telecom Operations Map, GB910, Evaluation Version 1.1, TeleManagement Forum, Morristown, New Jersey, USA, April 1999.
- [tmf-501] Customer to Service Provider Trouble Administration Requirements Specification, NMF 501, Issue 1.0, TeleManagement Forum, Morristown, New Jersey, August 1996.
- [tmf-503] Service Provider to Customer Performance Reporting Business Agreement, NMF 503, Issue 1.0, TeleManagement Forum, Morristown, New Jersey, March 1997.
- [tmf-504] SMART Ordering - SP to SP Interface Business Agreement, NMF 504, Issue 1.0, TMF, Sep 1997
- [tmf-601a] Customer to Service Provider Trouble Administration Information Agreement, NMF 601, Issue 1.0, TeleManagement Forum, Morristown, New Jersey, March 1997.
- [tmf-601b] Customer to Service Provider Trouble Administration Analysis Specification, Version 7.0, NMF 601, TeleManagement Forum, Morristown, New Jersey, March 1997.
- [tmf-701] Performance Reporting Definitions Document, NMF 701, Issue 1.0, TeleManagement Forum, Morristown, New Jersey, April 1997.
- [tmf-910] NMF Technology Map, Draft NMF GB909, July 1998
- [tmf97] CORBA Interface Specification for Customer to Service Provider Trouble Administration, Revision 0.2, TeleManagement Forum, Morristown, New Jersey, November 1997.
- [vincent] Modeling/Design Methodology and Template, Draft 4, Vincent, A, Hall, C., TMF, Oct 1997

-
- [wade98] V. Wade et al, Initial Guidelines for System Design, Component Integration and Re-Use, FlowThru, 1998.
- [wade99] Approaches to Integrating Telecoms Management Systems, Wade, V., Lewis, D., Malbon, C., Richardson, T., Sorensen, L., Stathopoulos, C., Proceedings of the 7th International Conference on Intelligence in Services and Networks, Athens, Greece, Springer-Verlag, Apr 2000
- [weissenfels] J Weissenfels, D Wodtke, G Weikum, A Kotz Dittrich, http://paris.cs.uni-sb.de/public_html/papers/mentor.html
- [wfmc] Workflow Management Coalition Homepage <http://www.wfmc.org/>
- [x790] Trouble Management Function for ITU-T Applications, ITU-T Recommendation X.790, Geneva, 1995.
- [x901] Information Technology- Open Distributed Processing- Reference Model- part 1: Overview, ITU-T Draft Recommendation X.901/ ISO/IEC Draft International Standard 10746-1, 1995.