University College London
Department Of Electrical and Electronic Engineering
MSc in Telecommunications

Network and Services Management

1997-98 Assignment

Isaac Kokkinidis

April 1998

Contents

1 Introduction

The most important issue for a network operator is the viability of the whole telecommunications system as a business. In order to achieve this, the system's resources should be managed efficiently, increasing the delivered quality of service (QoS) and reducing the operating costs. In today's competitive market, the focus of network management has been shifted from just having the network up and running, towards maximising customer satisfaction from the service provided.

In the following sections of this document, solutions and aspects of managing telecommunications networks will be presented. In section 2, issues concerning the design of a customer reception for a telecommunications operator will be addressed, answering Question **A-1**. Section 3 deals with equipment management issues, which are the focus of Question **B-1**. Finally, a comparison of software architectural frameworks for telecommunication management is presented in section 4, as required in Question **C-1**.

2 Customer Reception Support System

Customer facing corporate sectors are crucial for a telecommunications operator's public image, and therefore for its business success. Customers are satisfied when their requests are handled accurately, quickly, completely, and in a friendly way. With the latter excluded, these requirements necessitate a careful design and a considerable investment to the customer reception department.

For simplicity, the possible residential customer transactions can be categorised to service requests/modifications, problem reports, and general service-related queries. One thing that has to be noted, is that the time-scale can vary widely for different customer requests. The customer should be kept aware of the expected execution time of his request.

A hierarchical scheme will be suggested to model the handling of the transactions by the customer support system. According to this top-down approach, the top two levels will apply for all the transactions, and only in the last level will the actions specific to each of the above request categories be distinguished. The activities, the requirements, and the support systems involved in the customer reception, are all included in the following description.

2.1 Top Level

In the highest abstraction level, the process of dealing with customers' requests can be segmented into three main phases:

- *receiving* the request and creating the transaction records,
- *analysing* the request and determining the actions needed, and
- *handling* the request by taking appropriate actions.

An alternative segmentation could be done according to the Awareness, Decision and Implementation (ADI) management model. Both approaches are shown in Figure 1.

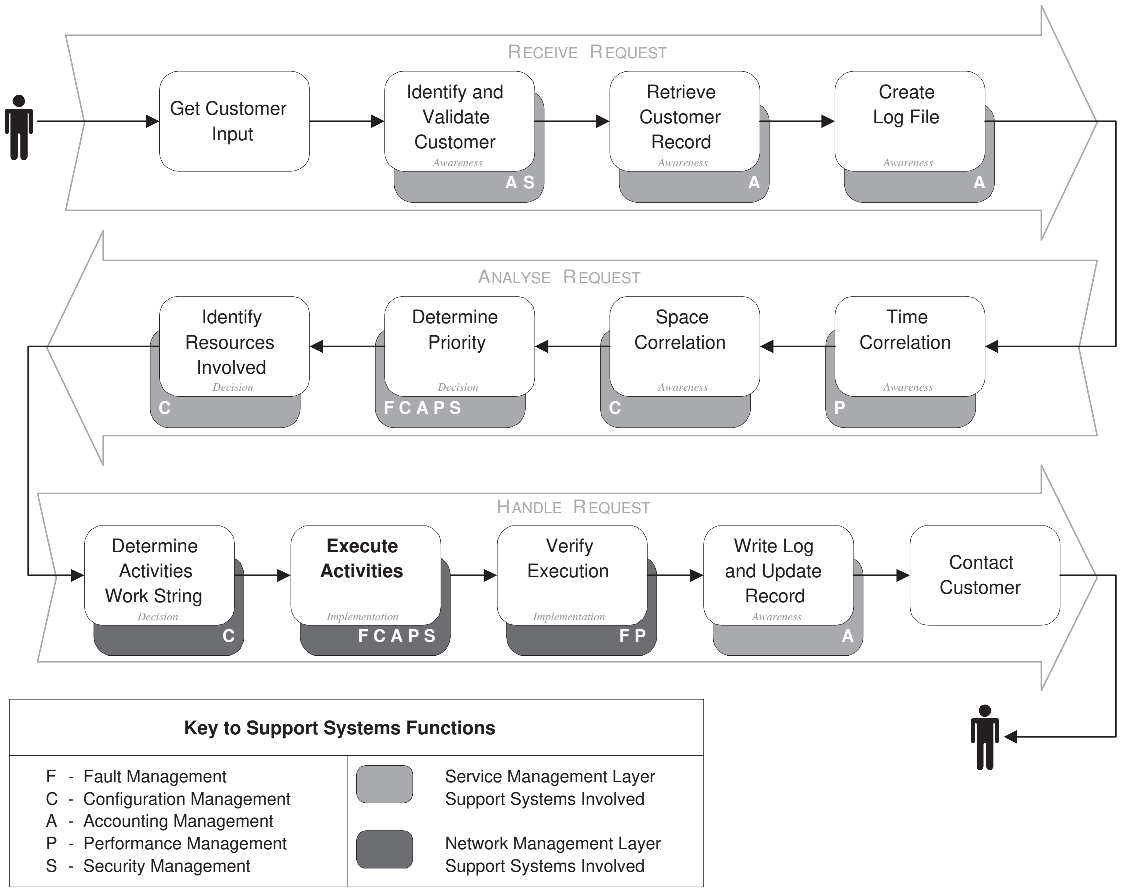

Figure 1: A suggested customer reception work string and its support functions.

2.2 Middle Level

Moving on to the next level, a suggested path for satisfying all the types of customer requests is depicted in Figure 1. As each operator's requirements are unique, the optimum design for these tasks is possibly different than the one suggested. Therefore, some activities could be executed in a different order, others could be executed concurrently, and others off-line, for not urgent requests.

For all the activities, there is a requirement for non-stop processing reliable support systems, possibly through fault-tolerant implementations, as redundant equipment arrays. Transparent interconnection using open interfaces and protocols, and distributed architectures are also crucial for the management tasks, as well as for the communication with other operators' support systems. The performance of the computing facilities should be adequate to deal with several concurrent tasks, possibly in real-time. Moreover, hardware and software scaleability is essential in order to accommodate middle and long term expansion of the network and its customer base [1, 2, 3].

These support functions are depicted in Figure 1, using International Standard Organisation-Open Systems Interconnection (ISO-OSI) terminology for the management functional areas. The Telecommunications Management Network (TMN) layer in use, is indicated by the appropriate colour. Support systems in the TMN Service Management Layer (SML) provide adequate functionality for most of the actions shown. In the request handling stage, however, interaction with

Network Management Layer (NML) support systems is needed. In detail:

Identify and Validate Customer is an essential procedure, in order to avoid time-consuming dealings with irrelevant requests. The SML Accounting and Security functions can cater for this action, as such information is stored in the upper levels of the management hierarchy. The support equipment includes private databases keeping the customer records, secure network infrastructure, and authentication procedures for retrieving such information. Software could be based on client-server architectures, since the information repository is usually centralised.

Retrieve Customer Record is providing the customer service representative with the appropriate information and the contact history of the requesting customer, which enables him to resume a session which from where if was left off. This functionality can be provided by the SML Accounting function, as in the previous action. The equipment is similar as well, including graphical user interface workstations, implementing a Computer-Telephony Integration (CTI) solution.

Create Log File to store the current transaction's details, so as to use in future contacts of the specific customer. This is a powerful marketing tool for operators, as by monitoring all the transactions they can derive the features that are more important for that customer. As above, SML Accounting support systems are catering for this procedure.

Time Correlation is checking the history of similar requests and network events to determine the time-related dependencies of the request. This could be used to prioritise transactions and appropriate activities, as described below. Statistics gathered from SML Performance function, are needed to accomplish this action.

Space Correlation is checking the configuration and equipment dependencies of similar past requests or events. It can be used as an initial estimation of the resources that need to be energised or reconfigured in order to satisfy the request. SML Configuration management provides the appropriate support system for this action.

Determine Priority is filtering the requests, according to the correlation information described above, and, possibly using Artificial Intelligence logic, prioritises the subsequent actions needed. This is a milestone point as the first decision-making action in the string, exploiting all the information from all SML support functions gathered so far. A knowledge management system using technologies as Case Based Reasoning (CBR) could be used. Such a system allows automated decisions based on known cases, even when presented with fuzzy queries or missing data.

Identify Resources Involved is the second and the last processing action in the analysis stage. A high-level procedure, it determines the equipment and human resources that are responsible for the further handling of the request. Therefore, it needs the SML Configuration support function, and uses technologies similar to those used in the previous stage.

Determine Activities Work String is the first action that accesses the TMN NML, specifically the Configuration support function. It checks availability of human and equipment resources, distributes and assigns the appropriate tasks, and sets a completion time target. These tasks are very complicated, and not always automated, which means that human intervention is usually necessary.

Execute Activities is where the request specific actions, comprising the bottom level of the suggested scheme, take place. A more detailed description of this level follows in the next paragraph.

Verify Execution involves checking the post-execution state of the network, and ensuring that efficient operation is not hindered by the actions taken previously. NML Fault and Performance support functionality is needed for this task.

Write Log and Update Record is the final administrative action before the request execution is acknowledged to the customer. As in the first work string actions, the SML Accounting support function is used.

2.3 Bottom Level

The work flows between the management functional areas for the three typical customer requests are depicted in Figure 2. At this point, most support systems used are in the NML level. In detail:

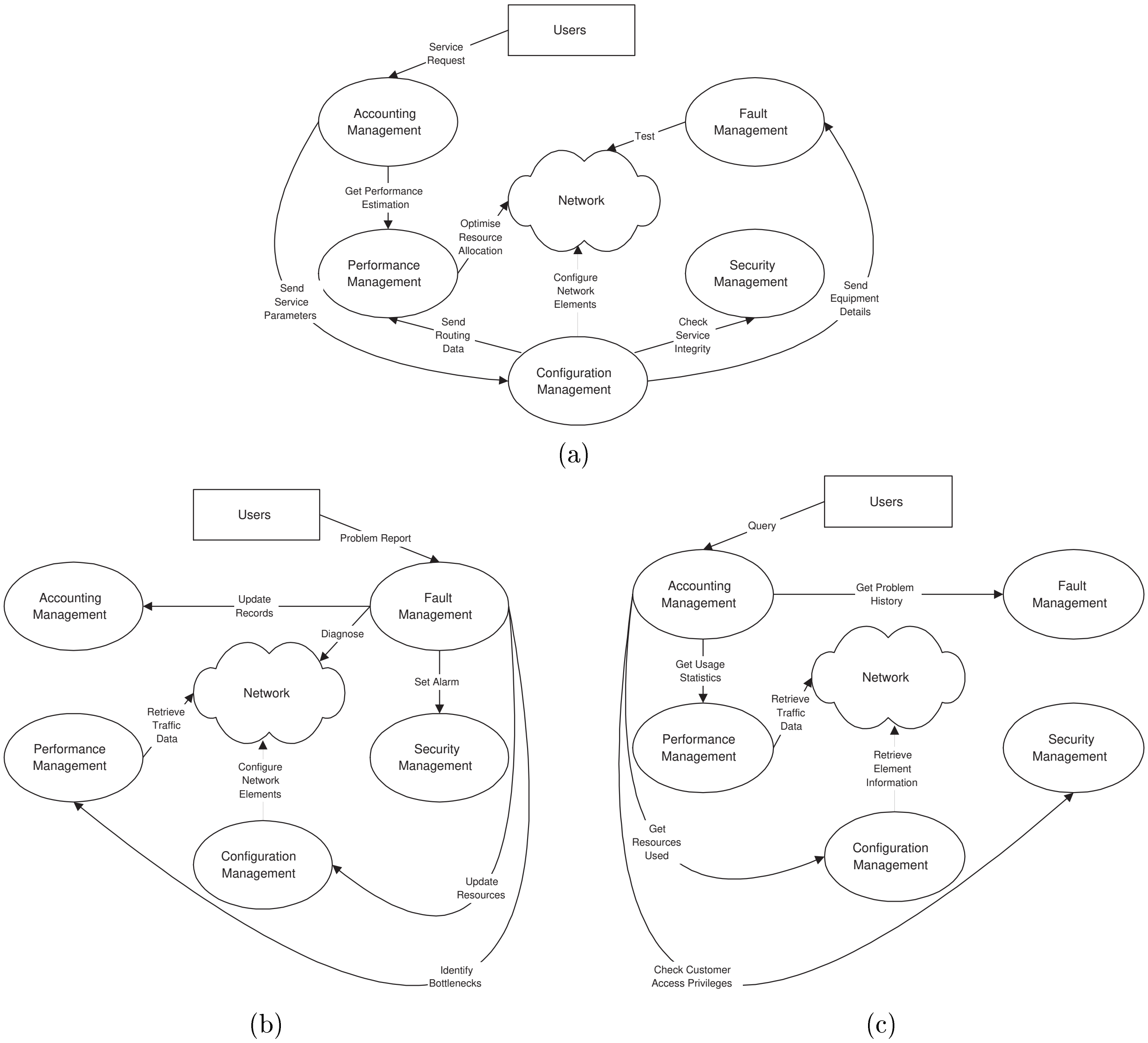

Figure 2: Functional work flows when handling: (a) a service provision/modification request, (b) a problem report, and (c) a customer query.

Service Request/Modification. The centre of the activity execution in this case is the Configuration management function, which will make all the changes to the network elements. Security management will guarantee the service integrity in collaboration with the Fault management tests, and Performance management will optimise for efficiency the network resources involved.

Problem Report. The Fault management function is the one that will accomplish the task of identifying the source of the problem, in collaboration with the Performance management. An alarm condition will then be set through the Security management function, hindering further use of the problematic network resources, and the situation will be restored by actions of the Configuration management function.

Query. Queries involve mainly the gathering of statistics and usage data, and possibly their processing and presentation to the customer. The activity centre is the Accounting function. Before disclosing the results, a check of the customer access privileges should be done through the Security management function.

3 Equipment Management

As the liberalised telecommunications markets proliferate, it is hard to afford ignoring the call for openness. Competition has made network operators to reconsider their traditional devotion to a single equipment vendor, in order to reduce costs. This has increased demand for multivendor systems, that rely on common open architectural principles in order to be interoperable. On the other hand, large equipment manufacturers are providing complete product lines, which include Operations, Administration, Maintenance and Planning (OAM&P) support systems. These are traditionally designed using optimised for the specific product proprietary architectures.

Examples of managed telecommunications equipment include switching equipment generally, such as multiplexers and cross-connects, and for various network technologies, such as ATM, PDH, SDH/SONET, SS7, and wireless networks. Other examples are Intelligent Network (IN) Service Control Points (SCPs), and Global System for Mobile Communications (GSM) Home Location Registers (HLRs).

In the remaining section, it will be assumed that the product will provide end-to-end broadband communications. As such, it will comprise a line of ATM switches, covering a range of speed/bandwidth requirements. This equipment should be provided in an integrated solution framework, with full management support. As a novel product, it will be assumed that no legacy management systems have been developed for the product so far.

A discussion on the adoption of open or proprietary architectures for an operational support system (OSS) follows. It comprises a comparative analysis of the two approaches, from the OSS developer viewpoint. Five phases will be considered in this analysis: specification, implementation, testing, deployment, and maintenance. Finally, the basic points are summarised and conclusions are drawn.

3.1 Specification

Adopting the open approach in the specification stage, will exploit the work made by standards organisations, reducing considerable the time and effort to design the system. The risk to be taken is that the mapping of the standards to the product might not work well, which could lead to a management with limited functionality on the one extreme, or with overloaded functionality on the other.

On the other hand, the proprietary architecture will allow a more tailored and compact solution to the product. This is outweighed, however by the fact that the logical model, distributed architecture and process definitions will have to be defined from scratch, running a risk of a non-functional final design. Another drawback is that this overspecialising approach is usually not scalable or upgradable, as minor configuration changes to adapt the product to the requirements of different customers, might lead to considerable modifications to the system structure.

On the specific product, work has been done by the ATM Forum on MIB (see Section 4 for definition) specification [4]. However, this offers a partial solution, covering mainly Configuration, Performance and Fault management. Therefore, and this is usually the case with recently developed open standards, there is not a turn-key solution offered, but customisations and enhancements have to be made. These will possibly allow for the tailoring of the management support functions to the product.

3.2 Implementation

The software part will be the main focus in this phase, as the hardware is dominated by the open and cost-effective solution provided by general purpose computers. The open approach has the advantage of using third party provided toolkits, usually from the software industry. These packages, allow rapid application development and maximisation of code reuse. Common open platforms and Distributed Processing Environments (DPEs), as SNMP, OSI-SM and CORBA (described in Section 4) are supported, and long term commitment to these architectures is provided.

Domestic-developed applications, offer the opportunity to exploit the experience gained from past products, and do not have the disadvantage of climbing the usually steep learning curves of third-party toolkits and open distribution frameworks. This is not a great advantage, however, since the product line is novel and time-consuming adaptation of developed code and distribution architecture models might be needed.

On the specific product, there is support by the ATM Forum for SNMP and OSI-SM (through CMIP) management frameworks [4]. Example tasks accomplished by the support system, covering all the FCAPS areas, include contract definition and user policing, QoS monitoring, performance statistics and traffic characterisation, fault and congestion monitoring, user-controlled resources.

3.3 Testing

For the developers, testing is a very important phase, and it could be done concurrently with the implementation for such complex systems. Sources of complexity are the inherent distributive and autonomous behaviour of the individual support systems and the variety of the managed entities.

In the open approach, the support system can be tested before and during deployment with a wide range of equipment. Thus, overall robustness and reliability is enhanced, and risks are reduced. With proprietary implementation, however, interoperability with test equipment is an issue. The product can only be tested with limited equipment and the risks are much higher.

As the end-to-end connections could span many different networks, testing their seamless interworking is crucial for their operation. Interworking is not defined only in transmission terms, but also in the service context. For example, it is highly desirable that a ATM service, such as a VBR connection, maintains its QoS characteristics when the end users are connected using equipment from different vendors.

3.4 Deployment

The advantage of the open approach in the deployment phase from the end-user viewpoint is definite, as it the only way to meet the demand for privileged use of the network with outsourcing. There is clear advantage for the network operator–the client–as well, since multivendor interoperable architectures are possible, increasing competition and reducing the cost of investment.

From the developer point of view, the great advantage is flexibility and scaleability, as any interoperable equipment can be added or replaced, making it possible to patch the product line with any possible omissions, and provide a complete solution. Another advantage is the easier and seamless integration of special packages for advanced systems such as the decision making expert and AI systems, which are entirely out of the scope of the telecommunications management development team.

3.5 Maintenance

The open approach has a clear technology advantage at the maintenance phase, offering a more future-proof design. This saves a lot of maintenance work from the management support system team, as no special knowledge and equipment is needed to deal with routine maintenance tasks, which are accomplished by the network operator. Moreover, the aforementioned robustness of the open architecture reduces significantly the amount of maintenance-related work load.

The proprietary approach does not exhibit any long term advantage for the equipment manufacturer. A short-term advantage might be a revenue increase from the maintenance department, as only specialists from inside the company might deal effectively with problems, but this is not compatible with today's market requirements, and in the long term it will prove a decision with negative results.

3.6 Conclusion

In the above discussion, it has been demonstrated that the advantages of open systems management outweigh those offered by legacy support equipment. Open standards are usually a result of a long-lasting research trying to accommodate a wide range of applications, and usually provide implementation flexibility by only specifying guidelines. In real world situations, however, they might prove inadequate to satisfy the special needs of a customer. In these cases, the manufacturer has to make a compromise from the dogmatically open approach and provide a customised solution.

Even in such a situation, it is essential to provide an interface to the standards-complying equipment through an adaptation function. Examples of these are the TMN Q-Adaptor Functions (QAF) [5]. Of course, adaptation should only be used in cases where management support with the natively open systems is not providing the desired functionality. If it is extensively used, the adaptation equipment in a network might proliferate. This eliminates the advantages mentioned above, raises maintenance costs, and virtually renders the system not open.

Despite some short term benefits that may be reaped from insisting on legacy systems, the future is on the open, multivendor systems. Towards this direction push customers, network operators and the regulation offices. In the highly competitive telecommunications equipment manufacturing market, successful will be those vendors, who provide interoperable solutions, without ignoring the special needs of their customers.

4 Software Models for Telecommunications Management

Telecommunications management is a challenging field for software technologies, as it is dealing with highly dynamic and distributed resources. Candidate architectures include the *de facto* standard for Internet Management, using the Simple Network Management Protocol (SNMP), the *de jure* OSI Service Management (OSI-SM) standard, and the Open Distributed Processing (ODP) standard, backed by the software development industry through the Common Object Request Broker Architecture (CORBA).

In the following paragraphs, a description of the models used for each architecture and a comparison between them, is followed by a discussion of their application to general frameworks, as TMN or Telecommunication Information Networking Architecture (TINA).

4.1 The Manager-Agent Model

According to the manager-agent model, depicted in Figure 3, real network resources are abstractly represented by *managed objects* (MOs), which are accessed by applications in the manager role only through their agents. The model specifies in detail the management *interface* between the manager and agent applications, in particular, the MOs types or classes, the access service to them, and the supporting protocol stack. The interface between the agent and the MOs is left unspecified, thus customised proprietary implementations are used in it.

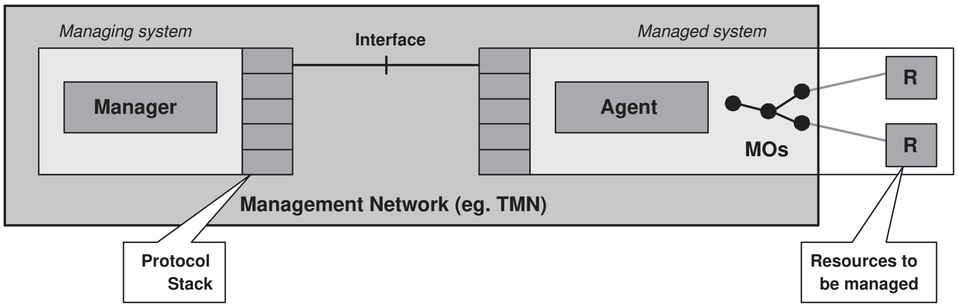

Figure 3: The Manager-Agent model, source: [6].

The manager and agent roles are logical entities rather than autonomous systems, therefore, a network resource could perform both tasks, and cascading structures are possible. Apart from the MOs that are directly associated with physical or logical network resources, there are support MOs, which are used for tasks not explicitly requested by a manager, as network events.

In order to resolve the MO location, dependencies and state, a global database, the Management Information Base (MIB), is used. The MIB has to be implemented as a distributed, dynamic architecture to reflect the current state of the network resources.

4.1.1 Internet Management

Internet hosts, routers, bridges, printers, or any devices capable of communicating their status when requested are called managed nodes, as they have agent functionality. They are managed by management stations, which are general purpose computers running special management software [7]. SNMP MOs are not really objects in the object oriented (OO) sense, as they are just variables and they have no methods, other than reading and writing their values. They are defined using a modification of the OSI Abstract Syntax Notation One (ASN.1) object definition language, called Structure of Management Information (SMI). The MIB is implemented using simple data structures, as two-dimensional arrays.

Management is performed with requests from the management station to the managed nodes, inquiring their current state, and *traps*, which are the notifications that the nodes send to the station when an event occurs. Connectionless transmission using User Datagram Protocol (UDP) is used, so reliability is provided by retransmission and periodic polling by managers. The interface protocol used is SNMP, whose operations are listed in Table 1. Object location resolution is done with a bulk query/transfer of MIB structures to the manager.

From the above description it is clear that the internet management architecture, as a bare-bones approach, is fundamentally open. All new devices can be introduced and managed as long as they fulfil the relatively simple requirements for resuming agent functionality, in the same way that any computer can be connected to the internet as long as it sends and receives IP packets. Agent functionality comprises usually a few entries in a table, and listening in a predefined port for management requests.

Simplicity has its price, though, as the model has not been optimised with respect to performance and transmission efficiency. During its design, only network engineering aspects were taken into account, ignoring service management-related issues. Therefore the model cannot satisfy to-

Message	**Description**
Get-request	Requests the value of one or more variables
Get-next-request	Requests the variable following this one
Get-bulk-request	Fetches a large table
Set-request	Updates one or more variables
Inform-request	Manager-to-manager message describing local MIB
SnmpV2-trap	Agent-to-manager trap report

Table 1: SNMP message types, source: [7].

day's requirements for a variety of novel services, as broadband and wireless services, which are more suited to object oriented management approaches.

4.1.2 OSI Service Management

As a fully compliant to the object oriented architecture principles model, OSI-SM defines its MOs using inheritance from the basic reusable type, the *package*. Classes are formed using at least one mandatory package and possibly, making use of polymorphism, some *conditional* packages, which are run-time specified package instances. The MIB is a set of such classes, defined in the Guidelines for the Definition of Managed Objects (GDMO) language. Global naming is used, allowing for location independent agents.

Management is performed via the Common Management Information Service/Protocol (CMIS/P) primitives: Get, Set, Action, Create, Delete, Event-report, and Cancel-get. There are sophisticated mechanisms as scoping and filtering, which allow the management of multiple objects, with the minimum possible information transmission. Similar fine control and filtering mechanisms are used for events management, eliminating the need for frequent polling from the manager. The reliable connection-oriented protocols used are based on the OSI 7-layer stack.

With the scoping and filtering techniques there are vast improvements over the SNMP model in terms of network traffic and query or name resolution latency. Furthermore, the easily reconfigurable object oriented approach followed in all the aspects of the OSI-SM model, can map better to the management requirements of new services than the static internet model. Differences between all the software frameworks discussed here, are presented in Table 2, later in this section. As a disadvantage of the model, it could be noted that there is currently no relation from OSI-SM informal languages and notations to formal legacy software development languages, making the model seem more theoretic than pragmatic.

4.2 The ODP Model

The ODP model was developed to address the problem of *application* interconnection, as opposed to the problem of system interconnection addressed by the OSI specification. This fundamental change to the OSI point of view, was imposed by Information Technology (IT) sector companies, which with the formation of the Object Management Group (OMG) have developed the pragmatic counterpart of ODP, the CORBA specification.

The focus of the model is on the Application Programming Interfaces (API) openness, in addition to the access protocol openness, which is the only supported by manager-agent models. With the former, applications which are accessing objects according to specified guidelines can be used anywhere, in the same way that the latter accommodate any communication resources which comply to their guidelines for management, namely the protocols and the type of objects. The equivalent guidelines for the APIs are modelled as *interfaces*, which are essentially the objects of ODP. The details of these objects are presented in the following paragraph.

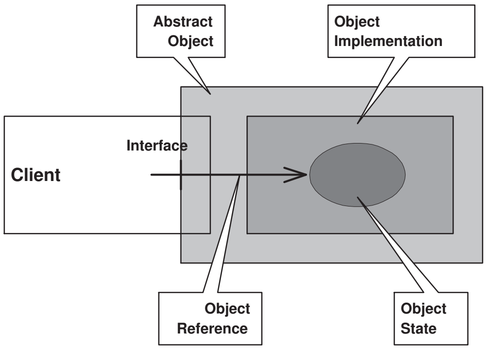

Figure 4: The ODP/CORBA object model, source: [8].

4.2.1 Common Object Request Broker Architecture

In CORBA, objects are specified as their interfaces, as shown in Figure 4. As in OSI-SM, the CORBA model is using inheritance to derive all the interfaces used from the root *object* interface. Unlike OSI-SM, however, in CORBA a common state may be bound to multiple interfaces, in contrast to the one object–one interface OSI model. Interfaces are specified by the Interface Definition Language (IDL), which has less OO functionality than the OSI-SM GDMO/ASN.1, but it is more readable, and it has been mapped directly to legacy OO software development languages, as C++, Smalltalk and Java.

The heart of the system is the Object Request Broker (ORB) which is accessed by client and server applications through the standardised APIs. Apart from hiding the heterogeneous underlying network and DPE, it is providing the communication infrastructure for client and server CORBA objects/interfaces. The latter are distributed as integral autonomous entities, accessed by type, in contrast to the partial attribute-and-methods-distribution, access-by-name scheme adopted by OSI and internet management. They accept get and set operations on their attributes, give replies and throw exceptions. Notifications are modelled as interfaces at the recipient side.

The fact that the access protocol is not the primary issue in CORBA, giving this place to the open API specification, does not mean that openness in the access protocol is not supported. For this task the Remote Procedure Call (RPC) has been specified as the connection-oriented protocol used over the TCP/IP stack, forming the Internet Inter-Operability Protocol (IIOP). RPC is a simple request/response protocol, and the functionality of sophisticated queries, equivalent to the OSI-SM scoping and filtering mechanisms, is provided by special interfaces, called special servers. Such servers include *traders* and name servers, which provide sophisticated query-event handling and enhanced object discovery functionality, respectively.

It has become clear from the above descriptions that the priorities were different when designing the three management frameworks. Internet and OSI management are focused on communications openness, while CORBA on software openness. This is expected, as CORBA is backed by the IT industry, whose interests are mostly on practical issues as software development efficiency, enhanced by extensive code reuse and strict API specifications, which are merits not exhibited by SNMP and OSI-SM.

CORBA is a much more general framework than SNMP and OSI-SM with its applications not limited to telecommunications management, and this raises some doubt to whether the architecture

Features	**Internet**	**OSI-SM**	**CORBA**
MO Basic Types	variables (scalar)	packages	interfaces
Object Definition	SMI/ASN.1	GDMO/ASN.1	IDL
Object Binding	N/A	dynamic	static
Object Resolution	bulk queries	conditional queries	trading
Object Naming	unique object IDs	unique for instance	multiple for instance
Object Access	multiple object, access by name	multiple object, access by name	single object, access by type
OO Architecture	no	yes	partial (no polymorphism)
Transport Protocol	connectionless, UDP	connection-oriented, CMISE	connection-oriented, RPC over TCP/IP
Notifications	traps, not configurable or scalable	sophisticated, fine control, filtering	sophisticated, filtering
Strengths	simplicity	performance, efficiency, fully OO compliant, access facilities	readability, maps to legacy programming languages, distrib.
Weaknesses	efficiency, readability, access facilities	agent complexity, no legacy software support	not tested in large scale, not full OO functionality

Table 2: Comparison of software models for telecommunications management.

will provide compact, optimised, and efficient solutions meeting the requirements of today's networks. However, as a pragmatic approach it delivers today with many products, sacrificing some of the object oriented concepts, which are dogmatically adopted by OSI-SM. The comparison of the three management frameworks is summarised in Table 2.

4.3 The TMN and TINA Software Models

The TMN was conceived as a hierarchical management framework enabling the centralised control of the network resources. In such a system, tight control is exercised from upper layers to well defined *groups* of highly dependent MOs. The manager-agent model, with the encapsulation of all dependent MOs in a single network entity, the agent, and the specification of access protocols enabling simple and detailed control from the manager, has been adopted as the suitable model for this approach.

The TINA is a software architecture developed to meet the demand for sophisticated services, as broadband and multimedia services. TINA specifies an information-centric view of resources *and* connections with its Network Resource Information Model (NRIM) [9]. The services mentioned above, require additional management entities which do not map well to the manager-agent model, as they exhibit a higher degree of *autonomy*. That is the reason for the adoption of the ODP paradigm by TINA. This can be better demonstrated by the service example described in the following paragraph.

Multicast multimedia connections are generally point-to-multipoint connections, with dynamically changing information sources, sinks and content, spanning combinations of, for example, ATM

Virtual Paths and Circuits (VP/VC), and SDH/SONET transmission technologies. Modelling the management of such connections is much more straightforward when using a distributed processing computational model, which assigns autonomous objects to each of the manageable entities. For this example, apart from the network elements, such entities include sources and sinks, network and subnetwork connections, VP/VC channel or transmission technology, traffic and content related objects.

TINA is still on the standardisation process, but it is definite that it will adopt, among others, ODP compliant distributed platforms. On the other hand, TMN has already been deployed in several applications, so intermediate approaches, giving TMN distributed processing functionality, and convergence solutions for TMN and TINA, are currently a point of research [10].

References

[1] R. Mathonet, "Integrated Management of Telecommunications Networks and Services," *Alcatel Telecommunications Review*, 3rd Quarter 1997, pp. 200-207.

[2] Online document, "Customer Care Tutorial," *ISM-BC in IEC Web Proforums*, at http://www.webproforum.com/ism-bc/index.html

[3] Online document, "Operational Support System Tutorial," *Manchester International Inc. in IEC Web Proforums*, at http://www.webproforum.com/manchester/index.html

[4] K. Krishnan and W. Fuller, "An Overview of MIBs for ATM Network Management," *53 Bytes, The ATM Forum Newsletter*, Vol. 5, No. 4, pp. 6-8, December 1997.

[5] R. H. Glitho, S. Hayes "Approaches for Introducing TMN in Legacy Networks: A Critical Look," *IEEE Communications Magazine*, Vol. 34, No. 9, pp. 55-60, September 1996.

[6] T. Magedanz and R. Popescu-Zeletin, *Intelligent Networks. Basic Technology, Standards and Evolution*, International Thomson Computer Press, 1996.

[7] A. S. Tanenbaum, *Computer Networks*, Third Edition, Prentice Hall International, 1996.

[8] Z. Yang, K. Duddy, "CORBA: A Platform for Distributed Objects Computing," *ACM Operating Systems Review*, Vol. 30, No. 2, pp. 4-31, April 1996.

[9] M. Lengdell, J. Pavon, M. Wakano, M. Chapman, M. Kawanishi, "The TINA Network Resource Model," *IEEE Communications Magazine*, Vol. 34, No. 3, pp. 74-79, March 1996.

[10] A. Manley, C. Thomas, "Evolution of TMN Network Object Models for Broadband Management," *IEEE Communications Magazine*, Vol. 35, No. 10, pp. 60-65, October 1997.