A method to ease creation of telecommunications services based on a Generic Component Model*

X. Logean°, *J-P. Hubaux*°, *S. Znaty*◇

°Telecommunication Services Group, TCOM Laboratory
Swiss Federal Institute of Technology, Lausanne, Switzerland

◇Network and Service Department, ENST Bretagne, Rennes, France

Abstract

This paper proposes a method to ease the creation of -distributed- telecommunications services. The method is based on a generic component model which is a common software architecture applicable to the design of every component of a telecommunications service. The concepts, architecture and specification of the Generic Component Model are presented. The method based on the Generic Component Model is then applied to the design of a Connection Management Service on top of an ATM communication platform.

Keywords: TINA, Service Specification, Design and Management.

1 Introduction

Nowadays the telecommunications market is characterized by a fast evolving technology, a multiplicity of providers and the need for value-added services such as Multimedia Services. Telecommunications services have requirements in terms of *reusability* for their quick provision, *interoperability* to alleviate service interactions management, *distribution* -a service consists of a set of components that interact with each other-, and *manageability* since control and management of services should be integrated within this service. To meet these requirements, a high abstraction level of services and systems is necessary.

The work presented in this paper aims at defining a method based on a Generic Component Model (GCM) used to build distributed telecommunication services. Genericity enables the model to be reusable for the design of any component, and therefore speeds up its introduction. Moreover, every component provides the same skeleton or template, therefore providing the basis for an easy treatment of the service interaction management problem.

The IN [1] Service Independent Building Block is a type of generic component; it exists only in one plane of the IN conceptual model and implements a very simple functionality. The INA building block [2], ODP cluster and capsule [3] belong to this category of generic software components, but are too low level to be directly used for designing services. In TINA [4] a generic service component architectures has been proposed but is yet to be enhanced for applicability. This generic component model is called Universal Service Component Model (USCM) [6]. Using this model, all services or service components are described as consisting of a core surrounded by an access layer. The access layer of a service is divided according to three access aspects namely usage, management and substance. These divisions are referred to as sectors.

The *Core* comprises the logic and data that define the intrinsic operation of the service.

Components in the *Usage* sector provide interfaces for the external components or services.

Substance sector components are the internal representation of the external components that the service uses.

Components in the *Management* sector provide the logic and data management to control the initialization, configuration, accounting, and other management functions needed to establish, configure, and characterize the service.

The TINA consortium has already finalized the work on the USCM, but no emphasis has been put on the issue of applying the USCM to service components definition and specification, nor on how to relate these specifications to the computational modelling concepts.

In this paper we present a Generic Component Model which specifies and enhances the TINA USCM by:

- the definition of generic primitives for the access sectors of the component,
- the unification and specification of the internal structure of the component,
- the unification and specification of the organization of the internal data of the component.

Although the GCM is based on the TINA USCM, it is not restricted to the TINA world, and can be used for the design of any distributed application.

Section 2 introduces the method based on the GCM and its underlying concepts to ease the creation of services. Section 3 illustrates the use of the GCM for the several types of TINA service components. Finally, the GCM is applied to the design of connection management service in a TINA-like platform called OAMS (Open Architecture for Multimedia Services over ATM) [8] which is our TINA testbed.

*This work is partially supported by Swiss Telecom PTT in the framework of the Ernes*TINA* project (http://tcomwww.epfl.ch/~ErnesTINA)

2 Generic Component Model

The method presented in this paper consists of applying the Generic Component Model at each stage of the service creation process (Figure 1). The GCM is first developed at

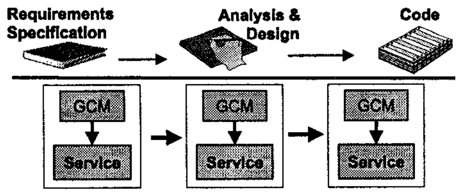

Figure 1: Service creation method

each stage: analysis, design and implementation. Then, the GCM is used at each level as a template. At each stage, the requirements of the service captured with the help of the GCM are used as input for the next phase. For Instance, the GCM is applied to help and capture the requirements of the service at the analysis step and the result of the analysis is used as an input for the design phase.

Management of network resources has become a crucial stage in the creation and use of open, object-oriented, generic and transparent services. Systems management involves managing and managed systems. A Managed system is itself divided into an *Agent* process and *Managed Objects* (MOs) which are stored in a Management Information Base (MIB) [9]. The Agent process is in charge of offering to the Manager a set of operations to be performed on the MOs and therefore hides the implementation of the MOs. The MOs provide a data representation of the resources for the purpose of management. The Agent translates the Manager requests which are CMIS-based or SNMP-based into methods offered by MOs.

The GCM presented in this paper follows the same philosophy, i.e., the clear separation between the way to act on a system and its implementation. Therefore, each aspect (usage, substance, and management) of the GCM has the following structure: an *Agent* which is the interface to the outside world and an *Information Base* (IB) which contains all the data and internal functions of a given aspect of the component. The GCM is therefore compliant with the TINA USCM which introduces the notions of core and access layer. Each of the sectors encapsulates an agent while the core of the component provides one IB per sector. All the data and internal functions of a component are contained in the corresponding IB: Usage IB (UIB), Management IB (MIB) or Substance IB (SIB). Figure 2 presents an OMT [10] meta- information model of the GCM. We can see from this figure that the GCM is made up of three parts (Usage, Management, Substance). A set of GCMs can interact with each other to provide a service (cooperative approach and association relationship), as well as a set of GCMs can be grouped into a global GCM (integrated approach and containment relationship) to provide a service functionality[1]. With such properties, a GCM may represent a large scale, complex service component or service, as well as a small simple one. Whatever size or complexity of the service, its structural organization is consistent with the GCM division, and therefore compliant with other services or service components. Figure 2 illustrates the GCM

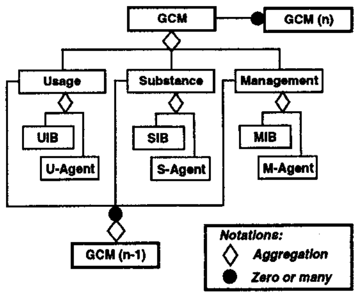

Figure 2: OMT diagram of a Generic Component Model

structure and the two next sections detail the generic aspects of both Agent and IB of the GCM.

2.1 Generic Agent

A GCM Agent is characterized by its own *Architecture* (internal view-core) and the *Service* it provides (outside view-access sector) (Figure 3). The Service Part is subdivided into *Basic Services* and *User Defined Services.*

The *Agent Services* defines the minimal functionality that a component should offer. The associated primitives are compliant with CMIS (Common Management Information Service) requests. These are Set, Get, Create, Delete, Action and EventReport [9] . Any operation can be achieved using one or a set of these primitives. The *User Defined Services* are services that other components can add and use. User Defined Services may be seen as supplementary services that are used to customize a given service. For example, specific security features may be implemented as User Defined Services. Service management tasks may also be delegated to a given service component through its management sector as a User DefinedService, where the User is in fact a manager. The *Architecture* consists of

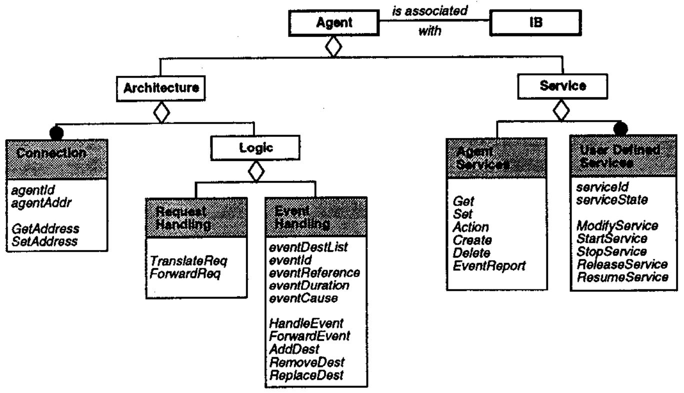

Figure 3: The Generic Agent of the GCM

two entities, namely *Connection* and *Logic*. The role of the entity *Connection* is to maintain the Address of each internal Agent (according to the "composition" property, an agent may be itself a GCM). If a request cannot be handled by the component, the Usage Agent forwards it to the

[1]A Generic Component at level(n) (where letter "n" stands for the level of the component) can be composed of multiple components at level($n-1$). Similarly, a GCM can be associated with other GCMs at level(n).

Substance Agent whose responsibility it is to find a service component that is able to perform the requested functionality. The entity *Logic* handles the Internal (Event) and External Requests and forwards them to the appropriate destination which may be an external component or a subcomponent of the GCM. Some parts of the GCM are optional. For example, the Substance sector is different from the other parts; it is a one-way request access. Through the Substance access sector, requests can only be issued. Therefore, there are no User-Defined Services and Connection entities for a Substance Agent. Each Agent has access to its own IB only, unlike the Management Agent which is able to get information from any IB, in order to maximize the information available to the manager.

2.2 Information Bases

Similar to the generic agent, the structure of the generic IB is subdivided into the entities *Architecture* and *Service*.

As an example, let us detailed the *Usage Information Base* (UIB) (Figure 4) which contains the data associated to the usage services offered by the component to other components.

The *Management* part of the UIB collects the static data, configured at the initialization of the component. The *Entity* part deals with the dynamic data: time since the initialization and several status of the component (operational, usage and administrative states). The *Entity* class is associated with the *Internal Flow* class which holds the information relative to the internal request of a component (e.g., requests forwarded from the Usage Agent to the Substance Agent). The *Value Added Services* part is concerned with the information specific to the functionality of the component. One component could be designed for authentication, and will therefore manage cryptographic algorithms, password lists, etc. Each Value Added Services class is associated with a *Session* class with data such as associated service, session identifier, session participants, etc.

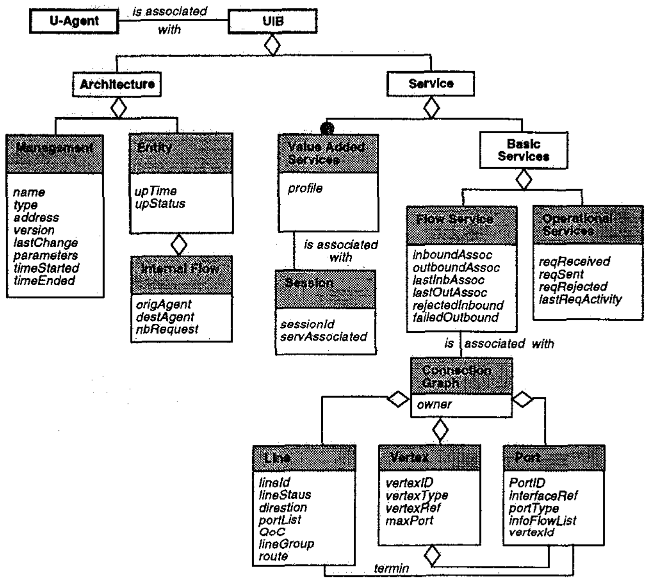

Figure 4: The UIB of GCM

The entity *Basic Services* is subdivided into classes *Flow Service* and *Operational Services*. In a distributed environment a component offers interfaces through which interactions will occur with other components. Depending on the nature of interactions that will occur, the interface is classified as being either an *operational interface* or a *stream interface*. The interactions that occur at an operational interface are structured in terms of invocations of one or more operations and responses to these invocations. When objects interact via stream interfaces, the information exchange occurs in the form of stream flows between the components, where each stream flow is unidirectional and is a bit sequence with a certain frame structure and quality of service parameters. The information concerning the requests, number of requests received/sent, rejected, etc., are found in the *Operational Services* class. The *Flow Service* contains the data specific to the stream flows exchanged between two components. Similar data have been identified in the Internet world in the Request for Comments on Network Services Monitoring MIB [12].

In a distributed environment, each association between two components can be modeled as a *Logical Connection Graph* (LCG), where vertices represent the component, ports represent the stream interfaces and lines represent the streams [13]. Therefore, each *Flow Service* is associated with an LCG specifying all the relative data: ports used, Qos of the line, etc.

3 Use of the GCM

In a distributed environment, a service is based on the interaction and cooperation of many components. The GCM presented in this paper gives the skeleton to design any component in a distributed environment. In this section, we first present the use of the GCM for designing TINA-like components. To help the designer of the service, TINA offers a set of components [14] and specifies their functionality. In Section 3.1 we present the use of the GCM to design those components. Section 3.2 depicts the use of the GCM to design components of a connection management service.

3.1 The GCM in a TINA environment

3.1.1 TINA computational viewpoint

The TINA Computational modelling concepts [11] define the rules of how computational objects interact with one another. Objects interact with each other by sending and receiving information to and from interfaces. An object may provide many interfaces, which may be of different types: operational and stream. The goal of the designer is to focus on the objects required by specifying the interfaces they offer and what interfaces of other objects are required. The notation chosen for computational specification is called *TINA ODL* (Object Definition Language) [15].

The TINA Service Architecture gives a set of service components that is divided as follows:

1. Independent Service Objects
2. Dependent Service Objects
3. Specific Service Objects

The first kind of components are identical and mandatory for any service; they are also specified by TINA in ODL. Applying the GCM to such components is therefore not of great interest, except for the validation of the GCM concepts in the TINA world. The second kind of components are needed within each service but are dependent of the service. They contain generic operations but also operations and data specific to the service. The genericity of the GCM is helpful for the design of those components, especially concerning reusability. The third kind of components are specific to the service (e.g., an authentication service, videoconferencing service). Applying the GCM to design such components warrants their compliance to TINA and their interoperability with the other TINA components.
Figure 5 shows how the GCM may be applied to each category of service components. The GCM will be used to design the *TINA Independent Service Objects.* Then these components will be directly instantiated at service run time. Therefore, they will not be reused but supplied within a service creation environment for the design and implementation of a TINA service. The *TINA Dependent Service Objects* will also be designed using the GCM. However in this case, the derived objects will be reused by the service designer. The *TINA Specific Service Objects* will be produced by the service designer directly from the GCM. In this case the GCM is the component to be reused to derive all the specific objects of the telecommunication service.

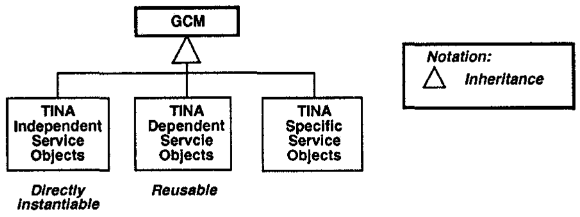
GCM
Notation:
Inheritance
TINA Independent Service Objects
TINA Dependent Servcie Objects
TINA Specific Service Objects
Directly Instantiable
Reusable

Figure 5: Inheritance tree of components

3.1.2 the GCM and the design phase

The mapping of the GCM to TINA ODL is facilitated by the intrinsic features of the GCM: the clear separation between the internal operations and the services offered to the other components. This separation allows a simple identification of the interfaces. Indeed, the *Service* part (e.d., *Agent Services* and *User Defined Service*) of the Agents of the GCM can be directly mapped to operational interfaces. The Architecture part of the Agents as well as the IBs identify the different attributes of the GCM. ODL has been designed to capture the distributed concepts, therefore only the interfaces and attributes interesting from a point of view of distribution are specified.

3.2 Using the GCM in a TINA-like environment for the design of a Connection Management Service

In this section, we show how to apply the Generic Component Model for the design of a Connection Management Service. We consider an underlying architecture called OAMS (Open Architecture for Multimedia Services over ATM) [8] which is a TINA-like management architecture where telecommunications services can be deployed. OAMS consists of four levels, telecommunication service architecture, management architecture, distributed computing environment and ATM communication platform. OAMS embraces TMN [16] within a framework based on Open Distributed Processing (ODP) [3] principles and object orientation. Currently, the management architecture consists of a connection management architecture which enables the establishment, control and release of connections with long holding times, i.e., of semi-permanent or permanent nature. Such an architecture can be used for any service that requires to dynamically add or remove connections between different sites (i.e., VPN, Videoconferencing, etc.) The Connection Management Service consists of a

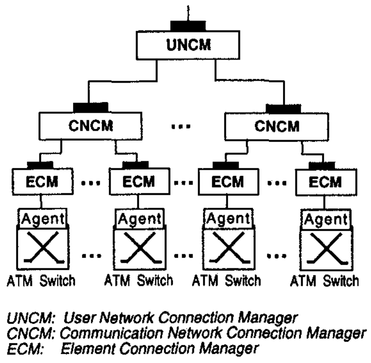
UNCM
CNCM
CNCM
ECM
ECM
ECM
ECM
Agent
Agent
Agent
Agent
ATM Switch
ATM Switch
ATM Switch
ATM Switch
UNCM: User Network Connection Manager
CNCM: Communication Network Connection Manager
ECM: Element Connection Manager

Figure 6: The OAMS connection management architecture

set of computational objects which are depicted in Figure 6. The User Network Connection Manager (UNCM) provides end-to-end connections. These end-to-end connections are then mapped into subnetwork connections established by communication network connection managers (CNCMs); subnetwork connections are finally translated into sets of cross-connections setup within switching elements. The different Components of this architecture are built according to the GCM. Figure 7 gives an example of a UNCM modeled with a GCM.

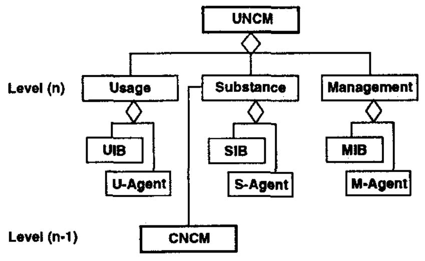

Figure 7: The User Network Connection Manager

In Figure 8 the use of the GCM for the design of the connection management service is illustrated. The UNCM U-Agent enables the establishment of end-to-end connections, their release and modification thanks respectively to CMIS M-CREATE, M-DELETE and M-SET operations. M-GET requests allow to retrieve data information about the connection; problems are handled via the M-EVENT-REPORT primitive. The U-Agent of the GCM already provides all necessary functionalities (Figure 8). The UIB provides the concept of connection graph with lines and ports. The UNCM S-Agent has access to the services provided by the CNCM for the establishment, control and release of subnetwork connections thanks to the appropriate CMIS requests. Therefore the UNCM-Agent is a subset of the generic S-Agent. The Substance IB contains all data concerning the possible operations on the CNCM. It complements the GCM SIB with information specific to the subnetwork connections established by the CNCM. The UNCM M-Agent is able to monitor and control the operation of the UNCM. The Management primitives are directly accessible through the Management Agent. The Management IB contains all data characterizing the statistics of the connection.

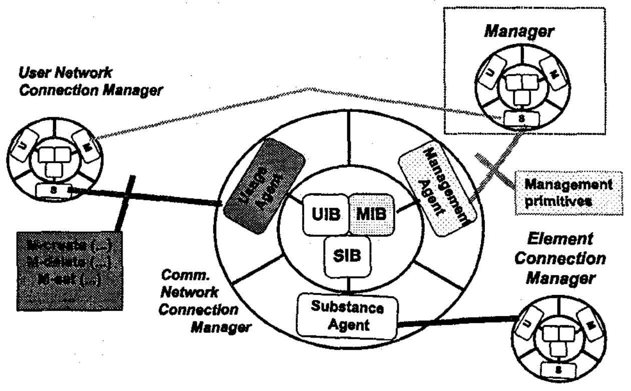

Figure 8: Using the GCM in OAMS

4 Conclusion

In this paper, we introduced a method to ease the creation of telecommunications services by applying the Generic Component Model at each stage of the development of services. The focus has been made on TINA-like services. The TINA Architecture provides a set of components for the design of services; few of them are directly instantiable. Our experience has shown that the GCM is very useful to design those components. The concept of GCM, mainly those of OSI network management with the notions of agent and MIB, goes beyond TINA; it supports reusability, extendibility, maintainability and interoperability criteria expressing current telecommunication software requirements. For example, it may be used for the design of any CORBA-based distributed telecommunications application. Finally, to put our approach on concrete form, we have considered a Connection Management Service and shown how to design such service using the GCM.

5 Acknowledgements

The authors would like to thank S. Koppenhöfer and F. Dietrich for many useful discussions that have significantly improved this paper.

References

[1] ITU-T Recommendation Q120X, Q121X, Q122X: The Intelligent Network ,1993.

[2] H. Rubin, N. Natarajan A Distributed Software Architecture for Telecommunications Networks, IEEE Network, 1994

[3] ISO/IEC JTC1/SC 21/WG7, Draft Recommandation X901, Basic Reference Model of Open Distributed Processing - Part1: Overview and Guide to Use, 1993.

[4] H. Berndt, L.A. de la Fuente, P. Graubmann, Service and Management Architecture in TINA-C, TINA'95, Australia, 81–86, 1995

[5] Chapman, M., Montesi, S., Overall Concepts and Principles of TINA, TINA-C, Version 1.0,1995

[6] D. Brown, The Universal Service Component Model,TINA Consortium, Document No. EN_B1.DKB.003_1.4_93, 1993

[7] J. Siegel, CORBA Fundamentals and Programming, John Wiley, 1996.

[8] Znaty, S., Service and Network Management in the OAMS Open Service Architecture, ACM Computer Communication Review, 1996.

[9] U. Black, Network Management Standards The OSI, SNMP and CMOL Protocols, MCGra-Hill, Inc, 1992.

[10] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, W. Lorensen, Object-Oriented Modeling and Design, Prentice-Hall, 1991.

[11] N. Natarajan, F. Dupuy, N. Singer, H. Christensen, Computational Modelling Concepts, TINA Consortium, Document No. TB_A2.HC.012_1.2_94, 1995.

[12] S. Kille, N. Freed (1994) Network Services Monitoring MIB, IETF, Network Working Group, 1994.

[13] C. A. J. Liccardi, Eurescom Project EU-P103, Network Resource Model, Eurescom, 1994.

[14] H. Berndt, et. al, Service Component Specifications, NA Consortium, Document No. TB_HK.002_1.0_94, 1994.

[15] Kitson, B., Leydekkers, P., Mercouroff, N., Ruano, F., TINA Object Definition Language (TINA-ODL) Manual, TINA Consortium, Version 1.3, 1995.

[16] Principles for a Telecommunications Management Network,ITU Rec. M.3010, 1992.