7th IEEE/IFIP International Symposium on Integrated Network Management (IM 2001), Seattle, Washington USA, IEEE Publishing, May 2001

Towards generic Service Management Concepts A Service Model Based Approach

M. Garschhammer, R. Hauck, H.-G. Hegering, B. Kempter, I. Radisic, H. Rölle, H. Schmidt
Munich Network Management Team
University of Munich
Oettingenstr. 67
D-80538 Munich, Germany
{garschha|hauck|hegering|kempter|radisic|roelle|schmidt}@informatik.uni−muenchen.de

H.-G. Hegering, M. Langer, M. Nerb
Munich Network Management Team
Leibniz Supercomputing Center (LRZ)
Barer Str. 21
D-80333 Munich, Germany
{hegering|langer|nerb}@lrz.de

Abstract

Service management has been a hot topic in the research community for the last couple of years. However, due to the complexity of this research area, no commonly accepted definition of the terms service, service management, and the associated management tasks has evolved yet.

This paper contributes to the ongoing process of defining these terms by proposing a top–down oriented and systematic methodology that is used to analyze and identify the necessary actors and the corresponding inter– and intra–organizational relationships. Then, a generic service model is introduced that defines commonly needed service–related terms, concepts and structuring rules in a general and unambiguous way. Since most of the work that is being presented here is still in flux, the service model is finally used to identify and structure open research questions.

Keywords

Service Management, Service Model, Provider Hierarchy

1 Introduction

The telecommunication sector has mainly been driven by the deregulation and liberalization of PTT monopolies within the last decade. The emerging universal service market exposes all players to strong competition and forces providers to think in terms of services, quality of service (QoS) parameters and service agreements when talking to their customers rather than discussing parameters of network devices or end systems. Increasingly, new requirements such as business process outsourcing and e–commerce extend the range of services from (classical) communication and internet services to complex application and value added services.

Due to the increasing complexity of those services, they are typically not implemented by a single provider. Instead, they are composed of interdependent and layered services of different carriers, Internet Service Providers (ISP), Application Service Providers (ASP) and Business Process Outsourcers (BPO). Generally speaking, all these organisations use terms such as service, QoS parameters and service agreements, but lack a *consistent* and *common* understanding of what these terms mean.

This evolution has serious implications on the management of these IT and telecommunication environments: Despite the substantial work that has been carried out in the area of service management, a common and overall accepted understanding of terms and tasks associated with service management has not been reached yet. One major reason is that most approaches focus on specific scenarios and management environments and thus, use a *different terminology* regarding service management. In our opinion, the development of generic service management solutions is not possible until a *common terminology* is defined.

Moreover, complex service hierarchies cross–sect all involved organizations, span multiple organizational, administrative and jurisdictional boundaries and lead to inter– and intra–organizational dependencies. These dependencies relate to all resources, systems, tools, applications, people, workflows and processes that are necessary to operate, administer, manage and provision services in heterogeneous environments and thus have a great influence on the management processes of all involved organizations. It is one major task of service management to identify and to model these dependencies, for example using roles, interactions and communication relationships.

In order to address some of the problems associated with service management, this paper proposes a *generic service model* that defines commonly needed service–related terms, concepts and structuring rules in a general and unambiguous way. This service model can be applied to all kinds of scenarios and helps to analyze, identify and structure the necessary actors and the corresponding inter– and intra–organizational associations between these actors. Since it also covers the whole service life cycle, it helps to establish, enforce and optimize information flows between organizations or business units. To abstract from all service and organizational details, the service model is the result of a *top–down oriented and systematic methodology*. This methodology ensures, that functional, organizational and life cycle aspects necessary for service management are considered. Finally, the service model can be used to *identify and structure research issues*.

To sum up, the generic service model that is presented in this paper fulfills the following requirements:

- **Generic and abstract service definition:** The model gives an abstract definition of a service and thus provides a common understanding for describing services independent of a particular scenario or environment. This generic service definition ensures, that the model can be applied to all kinds of services, from communication services to complex value added services including ones of distributed nature.
- **Integration of organizational aspects:** The modeling approach defines a service as the association between organizations that provide and use services. This approach allows to model complex real–life scenarios such as supply–chains and provider hierarchies that are typical for the emerging universal service market.
- **Separation of service and service implementation:** The separation of the abstract service description from the corresponding service implementation enables providers to implement services according to their local environment without restricting or implying a particular implementation.
- **Identification of generic building blocks:** Although the model does not prescribe the implementation of a service, it identifies generic building blocks (or tasks) that are necessary to implement a service.
- **Management as an integral part of the service:** Finally, our model considers the management of services as an integral part of the service itself. By using the life cycle to identify roles and interactions we can ensure that this approach covers all functional aspects that are necessary for the management of services.

The remainder of this paper is organized as follows: Section 2 gives a brief overview of current research work related to service management. Section 3 presents the top–down methodology used to develop the generic service model. Section 4 presents the service model itself and its application in provider hierarchies. It also points out the benefits of the proposed service model and identifies unresolved research issues from the building blocks of the service model. Section 5 concludes the paper and presents further work.

2 Related Work

This section gives a brief overview of the general research area of service management. It outlines the established concepts introduced by TINA-C, TMF and DMTF, and several new approaches that are being discussed in the research community. Although these approaches offer some interesting and useful concepts to define terms in service management, they do not address all the requirements identified in the previous section.

Telecommunication Information Networking Architecture Consortium (TINA-C)

The TINA service architecture [15] introduces a set of concepts, principles, rules and guidelines for constructing, deploying, operating and withdrawing TINA services. A TINA service is defined only in an informal way using plain text. The definition of service management within the Service Architecture is based on the concepts introduced in the Management Architecture [13], i.e. TMN layering concepts, the OSI FCAPS, computational aspects of management needs and life cycle issues. Additionally, new management concepts such as context negotiation and service transactions are introduced. The TINA definition of service management is mainly based on the concepts introduced by network and systems management of TMN/OSI, and most of the extensions towards service management are still unspecified or declared for further study. Most important, the TINA approach lacks a consistent top–down model that defines service and service management. Thus, it is not sufficient to identify and classify common service management issues and the associated tasks. Furthermore, it lacks a methodology that helps to apply all the introduced concepts for a particular scenario.

TeleManagement Forum (TMF)

The Telecom Operations Map (TOM) [16] introduced by TMF focuses on the end–to–end automation of communications operations processes. The core of TOM is a process framework that postulates a set of business processes that are typically necessary for service providers to plan, deploy and operate their services. These processes are organized using the TMN layering concepts and furthermore detailed to a finer granularity. TOM offers valuable concepts and addresses aspects of service management using business processes. However, TOM lacks a methodology that outlines how these processes have been identified.

Distributed Management Task Force (DMTF)

The Common Information Model (CIM) [4] introduces a management information model that claims to integrate the information models of existing management architectures; CIM acts as an umbrella that allows to exchange management information in an unrestricted and loss–free way. The Core Model [5] gives a formal definition of a service and allows hierarchical and modular composition of services consisting of other services. However, the focus is on technical details of the service implementation and does not include a notion of domains, such as customer and provider.

Current Research

Service management has been a hot topic in the research community for a few years now, and a lot of research has been carried out. Due to the complexity of service management, most of the approaches pick out one particular problem only; most approaches focus on a specific scenario and try to develop a suitable solution: For example, [11] deals with the integration of network and service management, but is restricted to switched ATM services. [7] introduces a QoS MIB along with some necessary management functionality, but restricts itself to multimedia services. The management of *Local Number Portability* is covered in [1], but only applicable in *Intelligent Networks (IN)*. [10] reviews various approaches to develop service management systems; however, the focus is on software engineering, rather than on conceptual aspects of service management.

Since all these approaches focus on one particular problem of service management, they do not provide a generic service model that can be used in different scenarios and environments. Even the approaches that try to build a general service model do not meet our requirements: For example, [14] develops a Service Management Architecture, which is limited to the phase of service usage; it does not consider the complete life cycle, one of our primary requirements. Furthermore, it does not address the problems arising from service hierarchies. Although [3] presents a very detailed model for service management, it does not consider service hierarchies, the relationships between customer and provider and the interactions that take place between these roles. According to [12], a service is composed of components. This definition does not meet our requirements, since it lacks a recursive definition of services composed of other (sub–) services and the associated management issues. Finally, [2] presents an architecture that uses contracts based on service level agreements (SLAs) to share selective management information across administrative boundaries. However, this paper focuses on the definition of a language to formalize SLAs and does not cover the full service life cycle.

3 Analysis Methodology

In the following we present our top–down approach for analyzing the service environment. The analysis methodology leads to objects and relations which form our service model. Finally, we use the service model to describe the definition of service–related terms. An overview of our analysis methodology is given in section 3.1, the analysis steps are carried out in sections 3.2, 3.3, and 3.4.

3.1 Top–down Approach

One of our main requirements is to develop a scenario independent service model. For this purpose we use a methodology, which is following an object oriented development style (comparable to UML). This results in the top–down approach depicted in figure 1, which is used to identify the elements of this model.

An inherent characteristic of every service is that it involves two major players: one offering and one requesting the service. Speaking in technical terms there is a *provider side* and a *customer side*. Both *interact* to accomplish a service. By solely examining these interactions we are able to draw conclusions about the service functionality without the need to take the service implementation into account. Therefore, it is important to identify these interactions for which we use a *service life cycle*.

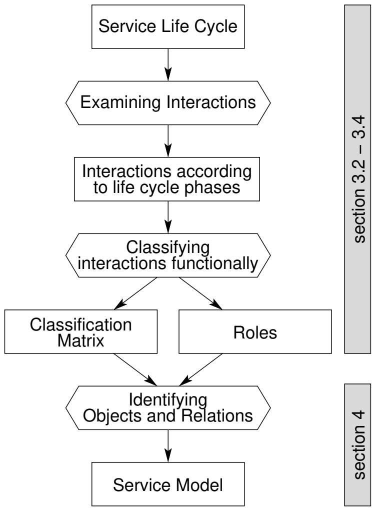

Figure 1: Analysis method

As it is impossible to find every single interaction regarding all possible services, an abstraction of these interactions is needed. Thus, we use *classes* to group the interactions. The life cycle phases lead to a first grouping of the interactions. To refine this rough structure, a *functional* classification (based on TMF's TOM [16] and OSI's Systems Management Functional Areas [8]) is performed in addition. The combination of these two activities leads to a classification *matrix*.

After all, as interactions take place between a pair of *roles* representing e.g., organizational units on both sides, roles are assigned to interaction classes.

By examining the identified interaction classes and roles, we are able to specify interfaces as well as entities participating in service provisioning. This leads to the final step of our analysis method: developing a service model containing objects and relations on basis of the former identified interactions, interfaces and roles. This step also encloses a recursive application of the model to represent provider hierarchies.

The remainder of this section applies the described methodology to examine interactions and to identify interaction classes and roles. In section 4 these entities are used to develop a service model.

3.2 Service Life Cycle and Interactions

Figure 2 shows the separation of the service life cycle into the phases *design, negotiation, provisioning, usage* and *deinstallation*, which is an extension of the life cycle proposed in [6].

The service life cycle starts with the *design* of the service. This includes interactions like the specification of the needed functionality and of possible QoS parameters as well as performing a cost assessment to be able to rate the service in the following phase.

Afterwards, the provider is able to offer the service to the customer which starts the *negotiation* phase. Depending on the complexity of the service this can be a very difficult and time–consuming task. Usually the customer and provider side have to negotiate QoS parameters, tariffs, penalties, discounts, escalation mechanisms, terms of usage (e.g., thresholds for number of transactions per minute) and management of the service (e.g., problem solution time). In fact the agreement contains the description of future interactions in a more or less detailed way that take place between the customer and provider side. The negotiation phase ends with signing a service agreement.

The *provisioning* phase, which follows the negotiation phase, contains interactions needed to properly install the agreed service. This means that the service provider has to implement, configure and test the service and its management. Additionally, the customer delivers needed data to the provider (e.g., user information, network connection points, etc.). Often the service agreement has to be adapted, as several parameters are not known until the implemented service is tested. Usually QoS parameters and accounting units of an individually designed service are not specified to concrete values until the end of a test run. The provisioning phase ends with a statement of acceptance of the service by the customer.

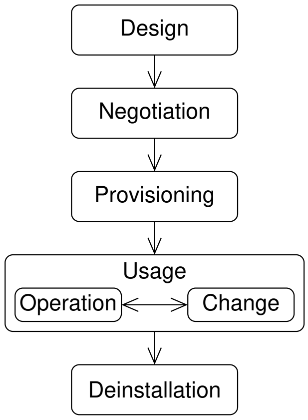

Figure 2: Service life cycle

The actual service usage by the customer takes place in the *usage* phase. During this phase the service is operational. It includes two sub–phases: *operation* phase and *change* phase. The *operation* phase includes all tasks needed to keep the service operational, like support, (QoS) monitoring, fault identification, fault resolution, maintenance, reporting, charging, billing and reviews. However, modifications to the service or its implementation may be required during the usage phase. These interactions are summarized in the *change* phase. Such modifications may change the service functionality, the quality or just the implementation being transparent to the abstract service. In some cases this implies an adaption of the service agreement.

Finally, the service ends with the *deinstallation*. Usually, the complete implementation is removed and involved resources are released.

3.3 Interaction Classes

As shown in the previous section by enumerating some example interactions between customer and provider, it is impossible to investigate every interaction occurring along the service life cycle. Thus, we need an abstraction of these interactions. Besides the life cycle phases that can be used for classifying interactions (every phase corresponds to a class), interaction classes spanning more than one life cycle phase can be useful: Such a classification beyond the life cycle phases helps to identify roles involved in more than one phase. One of our main conclusions after examining interactions along the life cycle is that there exist exactly two major interaction classes: *Usage* and *Management*.

Of course a refinement of the management interaction class according to functional viewpoints is necessary. The matrix shown in figure 3 classifies the interactions. The interaction classes result from an analysis of TMF's TOM [16] and OSI's Systems Management Functional Areas [8]. Combining the two classification approaches results in the interaction classes presented in figure 3, where a single interaction class is represented by a bar indicating the phases of the life cycle it spans. As an exception, OSI's performance management and TMF's QoS management are not shown. These two classes are internal to the provider domain and covered by the service management of the provider side without interacting with the customer side.

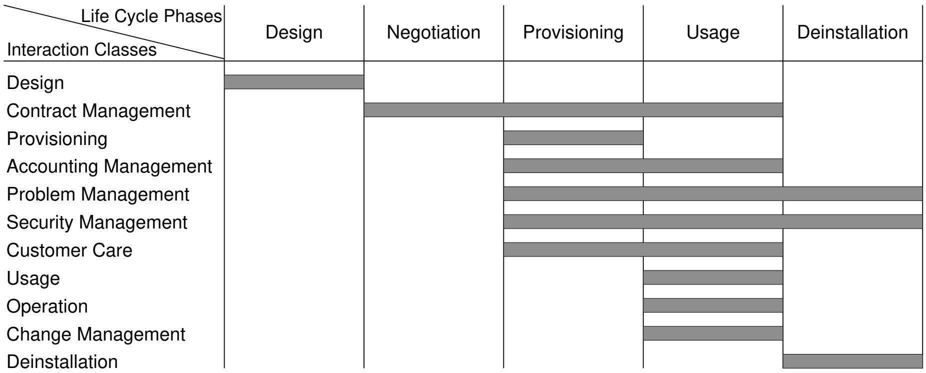

Figure 3: Classification of interactions according to the service life cycle

3.4 Roles

Besides identifying interactions, the main goal of the methodology is to clarify the participating roles. The combination of roles and interactions leads to objects, relations and interfaces involved in service provisioning and therefore needed in a service model.

Exactly one role on the customer side as well as exactly one role on the provider side is participating in every interaction. Such roles are grouped to reach a higher abstraction level. A role can be mapped to organizational units and is associated with certain *rights*, which contributes to the area of security management. In this paper we are concentrating on the *usage phase* at an abstraction level which allows to model a service generically.

As stated in the previous subsection, there are two major interaction classes: usage and management interactions. According to this, we identified two main roles on the customer side: *Users*, who actually use the service and a *customer*, who is interested in maintaining a subscribed service and therefore performs all the management activities on the customer side. On the provider side all necessary activities for enabling service usage as well as service management have to be performed. Since these activities cannot be strictly separated on the provider side's internal processes we only introduce the role *provider* for the provider side at this stage which is sufficient for the scope of our service model presented in the following section. Of course a more detailed analysis leading to more than three roles is possible.

Even though our methodology is more powerful in respect of application in any stage of the service life cycle and the roles involved, for the remainder of the paper, we use the roles user, customer and provider to describe the relevant interfaces, relations and objects in our service model during the usage phase.

4 Service Model

It is difficult to define the term service in a universal way, not restricting it to a small set of scenarios. The current research either leaves the term undefined or the definition is as narrow as needed to match the analyzed scenario.

Our approach is based on the top–down methodology presented in the previous section. As a starting point we define *service* according to our methodology as a set of interactions. But this is by far not enough to determine all aspects of a service. To narrow the definition we define the term service more precisely through the existence of the roles user, customer as well as provider and through their associations to the service. These roles and associations are defined in our service model depicted in figure 5.

The understanding of a service must be the same for customer and provider side. We follow the concept of service orientation which postulates the implementation independent description of the service from the perspective of the customer side. Furthermore, in a customer oriented world, the side independent information shared by both sides must be presented from the point of view of the customer side. The *side independent* aspects can be found in figure 5 between the two domains symbolizing *customer side* and *provider side*. This information is an integral part of service agreements.

The remainder of this section describes the service model in more detail. It is accompanied by a simplified example service, a *virtual store service* to sell products over the Internet (see figure 4). This service is offered by an *e–commerce provider* to dealers to sell their products. The user of such a service is potentially every *Internet user*. He actually does not know that he uses a virtual store of the e–commerce provider.

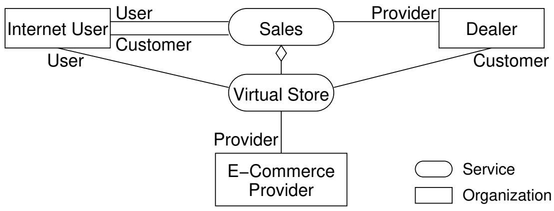

Figure 4: Example service

He thinks he uses the *sales service* of a *dealer* to get the wanted product. For the sales service the Internet user is also the customer. But for the virtual store he is just a user. The dealer is the customer because he outsourced the virtual store, which is a part of his sales service, to the e–commerce provider.

4.1 Side Independent Aspects

According to the main interaction classes, the service consists of usage *and* management functionality. Both types of functionality must satisfy a set of *QoS parameters*. These parameters define the minimum required service quality in order to be useful for the customer side. The QoS parameters define qualitative as well as quantitative values.

The *usage functionality* covers the interactions needed by the user. These interactions represent the actual purpose of the service. Besides these, interactions beyond the service's purpose are needed to fulfill the customer's duties, to customize the service according to user's needs and to control the provider's service provisioning. The *management functionality* comprises these interactions.

In the example the usage functionality of the virtual store service consists of searching the product database, retrieving information on a product and ordering a product. These interactions between user and provider are the purpose of the service. The management functionality for the dealer includes updating the product database, retrieving reports on the queries and orders of the users as well as receiving bills for the virtual store and paying them.

The QoS parameters comprise a maximum number of queries per second, response time for user queries, time until database updates are in effect, payment interval, and so on.

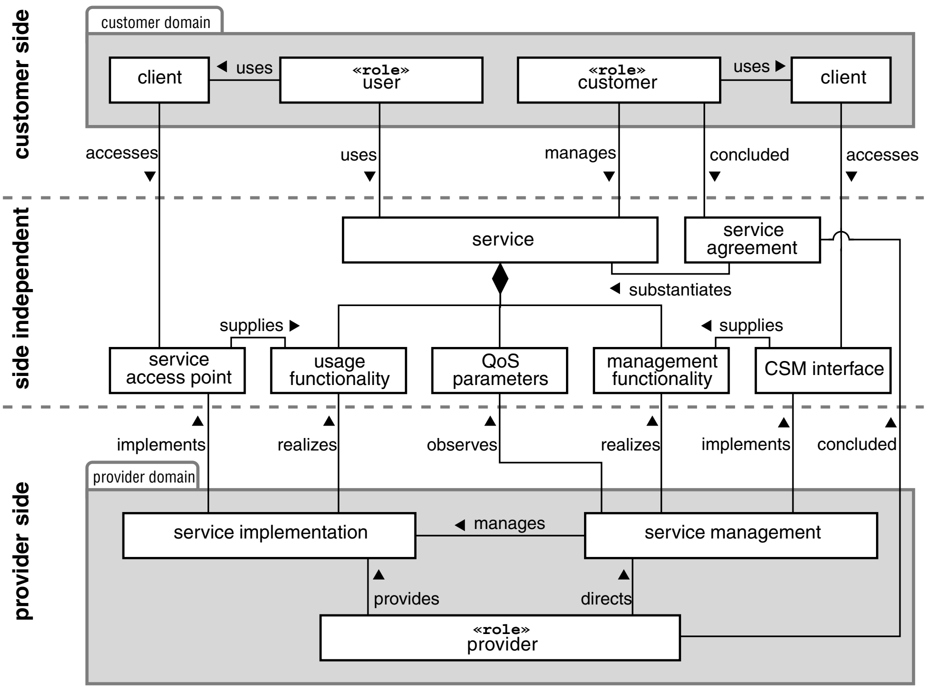

Figure 5: Service model

The *service agreement* substantiates the service by describing the usage and management functionality as well as the QoS parameters.

The information presented up to this point describes the service bought by the customer side and provided by the provider side. To actually be usable, there must exist a service interface between these two sides. Service primitives, protocols and (where necessary) physical connectors are represented by the service interfaces.

The interface definitions must also be included in the service agreement to enable the customer side to access the service functionality. The interface is the point where the responsibility of the provider ends. It must be stressed, that the interfaces are not part of the service. This means, changing or adding interfaces for service access does not result in a different service, though the service agreement has to be changed.

In the same way the functionality was split up in usage functionality and management functionality, the interface is split up in a usage interface, called *Service Access Point (SAP)*, and a management interface, called *Customer Service Management Interface (CSM interface)* [9], where the corresponding management functionality is accessible.

> The SAP for the virtual store could be a web–server accessible through the Internet. The interface includes the service primitives represented by HTML forms, several protocols like HTTP and IP. A physical connector is not specified in this example.
>
> The CSM interface needed by the dealer consists of a telephone number for a hotline and a leased line to modify the database and access reports. Payment is handled using a bank account.

To sum up, the service agreement specifies the service by defining the functionality for usage and management from the customer's perspective, the service quality and logical as well as physical aspects of the usage and management interfaces between customer and provider side.

4.2 Customer Side

On the customer side in most cases some equipment is needed to access the service functionality. Such *clients* allow user and customer to access the functionality at the SAP and the CSM interface respectively. Such clients can be telephones, computers or applications.

The technical clients must be compatible to the physical and logical aspects of the service interfaces. The sole responsibility for the clients rests on the customer side of the service.

> The user's client for accessing the virtual store is a computer with web–browser and Internet access. The customer also needs a computer which can access the leased line, a telephone and a bank account.

4.3 Provider Side

The main task of the *provider* is to make the service available. This includes all aspects of the service, namely the usage and management functionality of the customer side fulfilling the QoS parameters and the interfaces to enable the usage and management of the service.

For this reason the provider needs a *service implementation* which realizes the usage functionality of the service. To allow the user to access this functionality the service implementation also implements the service access point. The service implementation is the combination of all knowledge, staff, software and hardware needed to realize the service's usage functionality and the SAP.

The provider is also responsible for the service management. That means he directs it in a way which is suitable to keep the service above the agreed quality level but also to optimize the service operation according to other goals like high efficiency and low risk.

The main purpose of the *service management* is to ensure proper service fulfillment according to the service agreement. This means to care for keeping the QoS parameters in the agreed ranges by managing the service implementation. Additionally, it implements the management interface for the customer side allowing access to the service's management functionality that is also realized by the service management.

> The service implementation consists of a host with access to the Internet hosting a web–server, the product database and other supporting applications. The implementation also includes other needed equipment like air conditioning, fall back servers and so on.
>
> The service management monitors and controls the implementation to keep the service in the agreed QoS bounds, generates reports and realizes the management functionality for the customer.

4.4 Recursive Application of the Service Model

Many of the value added services that are now being offered by providers are composed of services that are supplied by various sub–providers. These sub–providers in turn can apply the same principle and can contract other sub–providers. As stated in section 1, the service model must be capable of modeling the resulting service chains and service hierarchies.

A provider contracting services of another provider acts as a customer to the latter. This means that the provider domain embeds the tasks of the user/customer role and the provider role simultaneously. As such, we can reuse the already modeled associations between the customer and the provider domain in order to model the associations regarding the relation of provider and sub–provider. By expanding the provider domain with the entities of the customer domain, we are able to create an enhanced model of the provider domain containing the classes *service implementation* and *service management*, the roles *provider*, *user* and *customer*, and the two *client* classes.

As figure 6 shows there is no provider–internal connection to the newly added classes of the provider domain. Further classes have to be added to the service model to close the gap between the elements of the two stand–alone models of provider and customer side.

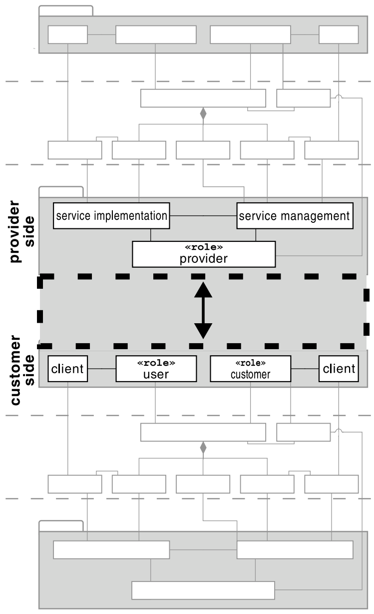

Figure 6: Chaining of services

Figure 7 shows how this gap can be closed. As already explained, all roles of the customer and the provider side of a service can reside within the provider domain. Thus, both, the *customer* role and the *user* role are part of the *provider* role. The *clients* used to access the subsidiary interfaces must be part of the service implementation and the service management respectively to permit an interaction with a sub–provider. As a consequence, the question for further elements within the service implementation and the service management raises.

The service implementation is composed of *resources* made available by the provider himself and *services* that are accessible through *sub–service clients*. Hence, we introduce a *service logic* to control both, the usage of services as well as the usage of the provider's resources. Thus, our class diagram shows the class *service implementation* consisting of the classes sub–service client, service logic, and resources. The sub–service client is actually just a refinement of the generic user client added to the provider domain.

The service management will use functionality of the traditional network, system and application management, i.e. the so called *basic management functionality* (BMF), along with the management functionality provided by subsidiary services. In consequence, there has to be a *management logic* controlling the BMF as well as the *sub–service management clients* for the subsidiary service management. The management logic treats the service logic as a managed object which leads to an association between the two classes service logic and service management logic. Corresponding to the service implementation we model the class service management as an aggregation of the three classes BMF, service management logic, and sub–service management client.

As both logics use corresponding clients to access the sub–service and/or their management respectively, they act in the role *user* (service logic) and *customer* (management logic). We model this correlation with two associations, one connecting the service logic and the user role, the other one connecting the service management and the customer role.

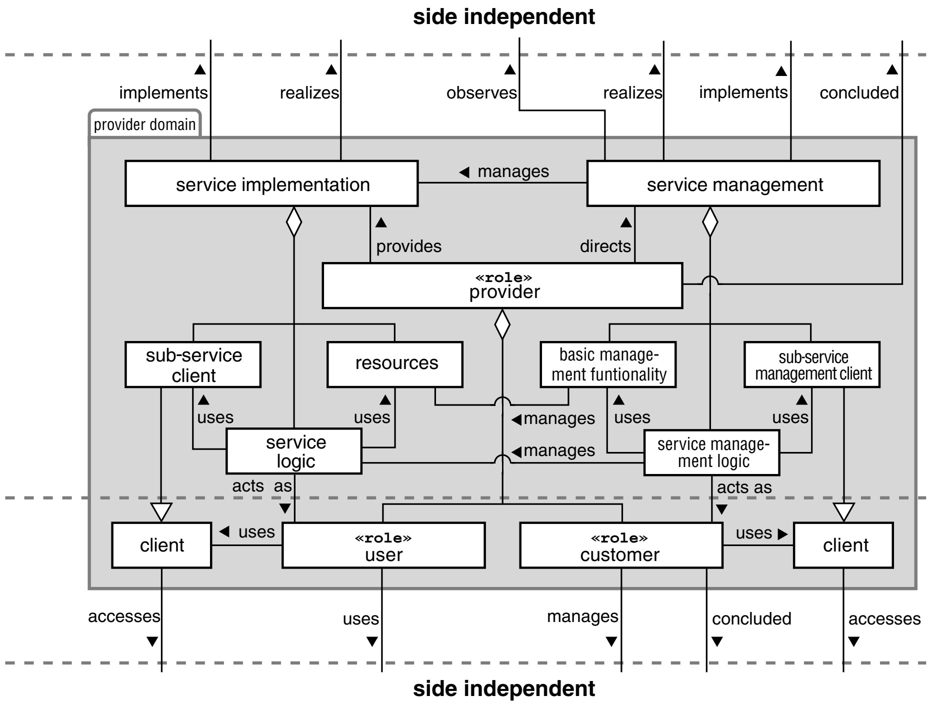

Figure 7: Model of a provider domain in a service hierarchy

4.5 Identification of open Research Questions

As already mentioned in section 1, the service model can be used to derive and classify research questions. The following list points out some of the open research questions that, we feel, need to be addressed:

- The service management is responsible for managing the service implementation. The detailed model of the provider side developed in section 4.4 shows that the management association cannot easily be represented in the detailed provider model. It is realized through various associations between the management and the implementation part of the model. Yet there is no formal description how to map the top level association "manages" to the internal relations between management and implementation. The development of this mapping is one of the most urgent problems in the area of service management today.
- Until now, no common methodology or suitable algorithm exists that helps to map service agreements in service chains or service hierarchies. This problem

is directly represented in our service model. Each agreement is only associated with its own provider domain. The impact of one agreement to each other still cannot be described.

- The service model shows an association between QoS parameters and service management. This association includes a difficult challenge for research: The mapping between service–oriented QoS parameters and the service implementation. It is difficult to operate a service with a given QoS because there is no formal methodology to attune the service implementation to the desired QoS parameters and to keep it within the given quality ranges.
- Finally, no systematic approach has been carried out to identify and formalize the necessary management interactions and management information that has to be exchanged between customer and service provider over the CSM interface. Furthermore, no mature concepts exist that help a provider to implement the CSM interface and integrate it with their service management.

5 Conclusion and Further work

In this paper we used a top–down methodology to analyze the interactions taking place in the environment of services. This resulted in a service model describing entirely the relations between customer and provider side of a service in the usage phase. The model itself determines commonly needed service–related terms by their associations. The methodology can easily be reapplied to analyze and classify the interactions of other phases of the service life cycle besides the usage phase.

The top–down approach used in the methodology was carried on with analyzing the model in recursion resulting in a more detailed description of a service provider. Thus, the service model was developed in a generic, non scenario specific procedure. An application of the model was shown by revealing open research issues.

Our current work, which is co–funded by several enterprises like Deutsche Telekom, Siemens, DeTeSystem and BMW, focuses on an analysis of the interactions in all other phases of the service life cycle and an integration of the results into the service model. Furthermore languages and methods for describing functionality and QoS parameters in a generic way will be developed. A further refinement of the roles, e.g. provider, is necessary to detail their associated functionality and to develop generic solutions for service management.

Acknowledgment

The authors wish to thank the members of the Munich Network Management (MNM) Team for helpful discussions and valuable comments on previous versions of the paper. The MNM Team directed by Prof. Dr. Heinz-Gerd Hegering is a group of researchers of the University of Munich, the Munich University of Technology, and the Leibniz Supercomputing Center of the Bavarian Academy of Sciences. Its web–server is located at `http://wwwmnmteam.informatik.uni-muenchen.de`.

References

[1] S.-K. An, S.-K. Kim, M.-J. Choi, J.-Y. Kim, and J.W. Hong. TMN–based Intelligent Network Number Portability Service Management System Using CORBA. In *Proceedings of the Sixth IFIP/IEEE International Symposium on Integrated Network Management (IM'99), Boston, MA*, 1999.

[2] P. Bhoj, S. Singhal, and S. Chutani. SLA Management in Federated Environments. In *Proceedings of the Sixth IFIP/IEEE International Symposium on Integrated Network Management (IM'99), Boston, MA*, 1999.

[3] G. Chen and Q. Kong. Integrated Management Solution Architecture. In *Proceedings of the NOMS 2000 IEEE/IFIP Network Operations and Managment Symposium — The Networked Planet: Management Beyond 2000*, pages 217–230, April, 10–14 2000.

[4] Common Information Model (CIM) Specification Version 2.2. Specification, June 1999.

[5] Common Information Model (CIM) Core Model. White paper, Desktop Management Task Force, August 1998.

[6] H.-G. Hegering, S. Abeck, and B. Neumair. *Integrated Management of Networked Systems – Concepts, Architectures and their Operational Application*. Morgan Kaufmann Publishers, ISBN 1-55860-571-1, 1999.

[7] James Hong, Jong-Seo Kim, and Jong-Tae Park. A CORBA-based QoS Management Framework for Distributed Multimedia Services and Applications. In *Proceedings of the IFIP/IEEE Ninth International Workshop on Systems: Operations & Management (DSOM'98), Newark, Delaware*, 1998.

[8] Information Technology – Open Systems Interconnection – Systems Management Overview. IS 10040, International Organization for Standardization and International Electrotechnical Committee, 1992.

[9] M. Langer and M. Nerb. An ODP Enterprise Specification of Customer Service Management for connectivity services. In *Proceedings of the 3rd International Enterprise Distributed Object Computing Conference (EDOC '99)*, pages 94–103, Mannheim, Germany, Sept., 27-30 1999.

[10] D. Lewis. A Review of Approaches to Developing Service Management Systems. *Journal of Network and Systems Management*, 8(2):141–156, 2000.

[11] D. Lewis, C. Malbon, G. Pavlou, C. Stathopoulos, and E.J. Villoldo. Integrating Service and Network Management Compoments for Service Fulfilment. In *Proceedings of the IFIP/IEEE Tenth International Workshop on Systems: Operations & Management (DSOM'99), Zürich, Switzerland*, 1999.

[12] Lundy Lewis. *Service Level Management of Enterprise Networks*. Artech House Publishers, 1999.

[13] Management Architecture Version 2.0. TINA Baseline TB_GN.010_2.0_94, TINA Consortium, December 1994.

[14] C. Mayerl, Z. Nochta, M. Müller, M. Schauer, A. Uremovic, and S. Abeck. Specification of a Service Management Architecture to Run Distributed and Networked Systems. In *Proceedings of the 3rd IFIP/GI International Conference on Trends towards a Universal Service Market (USM 2000), Munich, Germany*, 2000.

[15] Service Architecture Version 5.0. TINA Baseline, TINA Consortium, June 1997.

[16] SMART TMN Telecom Operations Map. Evaluation Version 1.1 GB910, TeleManagement Forum, April 1999.