Service Contracts Based on Workflow Modeling

Holger Schmidt

Munich Network Management Team
University of Munich, Dept. of CS
Oettingenstr. 67, 80538 Munich, Germany
Phone: +49 89 2178 2165, Fax: +49 89 2178 2262
schmidt@informatik.uni-muenchen.de

Abstract. The increasing importance of IT infrastructures and the complexity of IT services in each company often results in endeavors to outsource IT services. This implies meeting the challenge of service quality. Therefore, it is important to sign a service contract for the outsourcing partnership supporting effective usage and management of the service. It must be possible to monitor and manage the service in a constructive and fast manner.

This paper presents a customer-oriented approach for specifying service contracts. The idea is to combine service contracts with workflow concepts. The knowledge about design and management of workflows can be used to specify a service contract constructively supporting operation and usage of complex services. The use of the customer's business processes as a basis for the contract ensures a customer-oriented service view. Workflow concepts allow both, specifying non-ambiguous contracts and constructive instructions for usage and management of services by the customer. Combined with a suitable contract structure a controlled dynamics for the service contract is possible.

Keywords: Service Contract, Service Level Agreements, Workflow, Business Process, Outsourcing.

1 Introduction

The increasing dependence of a companies success on IT infrastructures requires to face the challenges of service quality. The complexity of the needed IT services is growing. Therefore, the design and operation of services is often outsourced to independent internal departments or to external partners.

To specify the requested service customer and provider sign a service contract. Such contracts define the functionality of the service and the required service level. The contract is an important source of information for the provider's service management because the service provider usually does not know much about the scenario the service is used in. Thus, the quality of the contract is one important parameter for the success of outsourcing relationships.

Keeping the service quality on a reasonable level is the challenge of modern services. It is also a task of service contracts to ensure that the customer has sufficient management facilities to fulfill the customer's duties, to customize the service according to the users' needs and to control the provider's service provisioning. Therefore, management interactions need to be specified in addition to service level and functionality.

A. Ambler, S.B. Calo, and G. Kar (Eds.): DSOM 2000, LNCS 1960, pp. 132–145, 2000.

In this paper the term management means the service management crossing the domain boundary from the customer domain to the provider domain. Management of the service implementation done by the provider is considered as operation.

The specification of interactions is difficult if it is based on a relatively informal collection of rules and statements. The temporal aspect of interactions cannot be represented adequately and it is hard to master the complexity with unstructured methods. From such methods many contracts for today's IT services result. Interactions can be modeled as a process. Production processes or business processes are described by workflow concepts. Therefore, this paper proposes the usage of workflow concepts for designing and writing of high quality service contracts for IT services.

For demonstration purposes a scenario is presented in section 2. Sections 3 defines the term service contract, derives requirements and identifies the contract elements. A graphical representation of workflows is introduced in section 4. Section 5 presents our approach starting with the contract structure followed by the process model and a classification of processes. Section 6 shows a simple process for contract design and its application. Finally, our approach is discussed in section 7. The last section draws a conclusion and introduces future work.

2 Scenario

To clarify the presentation of our approach and for demonstrating the potential complexity and investments of today's services a scenario is developed. (see figure 1). The scenario originates from experiences in several projects with industrial partners. It shows the complexity and high investments a custom–designed IT service can require today.

A company selling its high priced customizable products over an international network of dealers wants to connect those with its own central IT infrastructure. This permits the dealers to customize and order products online. Additionally, multimedia information material is available online for product presentations. Furthermore, up to date information on the stocking, production date, date of delivery, etc., is available for sales conversation of the dealers with their customers. Each product is individually traceable during the production and delivery processes. Similar facilities are available for after–sales contacts. The applications supporting this functionality are hosted by the company itself.

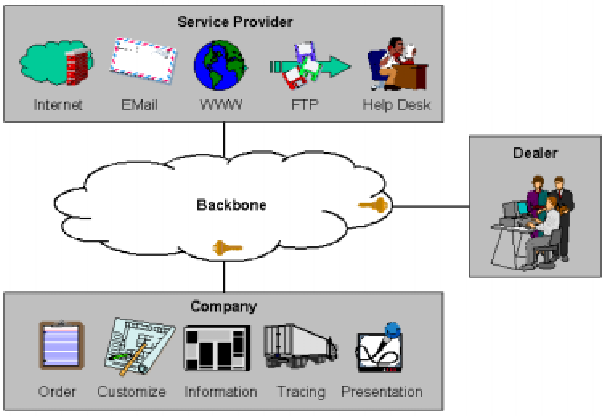

Fig. 1. Scenario

The company has a contract with a service provider for the functionality and necessary management facilities described next. The dealers can use Internet connectivity as additional features with special support for email, WWW and FTP. All communication between dealers and the company is encrypted for security reasons. A firewall secures the Internet gateway. The infrastructure is IP based which requires the operation of all supporting services like management of domain names and IP addresses including the necessary infrastructure. The dealer's connectivity to the provider's backbone is realized by leased lines or ISDN using components at dealers locations owned by the service provider. Finally, the usually quite unexperienced IT users at the dealer locations need a help desk supporting them in day to day connectivity problems. The service provider insists on a long–term contract of several years because the provider wants to amortize the investments in design, hardware, software and staff.

3 Service Contract

This section defines service contract and identifies requirements as well as its elements. A *service contract* is an agreement between customer and provider specifying the service functionality and all management interactions of the customer–provider relationship including the required service level for both.

3.1 Requirements

Common requirements like validity, completeness, consistency or unambiguity must be fulfilled by all types of contracts. But there are additional requirements for service contracts that specify complex IT services in a long–term customer–provider relationship. A service contract must be:

Customer–Centric: The contract must be specified in a terminology originating from the customer's usage scenario. The service implementation should be transparent to the customer. The service specification may not include terms of the service implementation view usually preferred by the provider. But this must not lead to restricted expressiveness. All necessary details must be expressible while allowing the abstraction of well known facts.

Dynamic: Flexibility of a contract is an important factor for a successful long–term relationship because the needs of a customer change relatively fast in the field of IT and therefore it is often necessary to adjust some parameters. Especially service levels need an adaption from time to time to fit customer needs. This must be possible in a controlled way. Not all parameters are tunable because they could require a different implementation. Therefore, the dynamic elements in a contract should be selected.

Constructive: Management facilities for the customer are an essential element in a contract that allows the customer to monitor and influence the provider's service operation to a certain degree, e.g., to customize the service to the users' needs. The management interactions should be used to actively prevent or at least constructively solve problems in cooperation. Operators as well as users should be guided to become active by a service contract if the situation requires intervention. The contract must be understandable for both, users and operators. Therefore, a contract requires concrete and workable instructions which are supportive for usage and management. All other management interactions like ordering, accounting, maintenance, etc. need to be explicitly specified, too. This way the service contract is present during operation and usage not just in a legal proceeding.

Summarizing, to reach these targets a constructive contract is needed which is customer–centric, service–oriented and controlled adaptable. Furthermore, it must be concrete, measurable, realistic, understandable and workable, but without loosing expressiveness. That means all contracting partners know their rights and their duties.

3.2 Elements

The rights and duties are specified by three groups of information: legal, organizational and technical information. To be considered valid a contract must comply with all formalities, e.g., names, addresses and signatures of the contracting parties are needed. This *legal information* is important for a contract, but it is regulated by law. Therefore, this paper focuses on organizational and technical aspects of the customer–provider relationship.

The service is described by *technical information* on functionality, capacity, quality and the interfaces for usage and management. Functionality describes what the service supports to do, not how it is done. Capacity and quality criteria specify the service level that must be fulfilled. The usage interface is the service access point. A management interface is needed for monitoring and influencing the service from the customer's domain.

The *organizational information* includes all interactions between customer domain and provider domain for the usage and management of the service. A usage interaction is for example ordering a product. Interactions like adding a new dealer or problem handling are management interactions.

3.3 Related Work

Most service contracts for complex services specify a set of rules and statements which are based particularly on experiences of provider and customer. This very informal procedure does not help in fulfilling the raised requirements. It is very difficult to write high quality contracts, if there is no structure or methodology guiding the design process which also allows to involve the service users.

An approach for specifying contracts in federated environments is presented in [1]. A service consists of several components. For each component quality metrics and interdependencies to other components can be specified. This straightforward approach allows to verify the service quality by evaluating relations of measured values provided by each component. The focus lies on the service implementation. This complicates the use for complex services because customers are not interested in the numerous measured values of components implementing the service. The implementation view is useful for providers but too difficult to understand for customers which are not familiar with the possible service implementations. Furthermore, this approach does not support management from the customer's domain.

Another approach defines a model of electronic services for long–term relationships between customer and provider [7]. Contracts are negotiated between customer and provider to guarantee the availability of a service for repeated use in an open service market where one–time service usage dominates. The approach is based on virtual resources which represent the interface between customer and provider. They are mapped to physical resources on service usage. This approach specifies the interface between customer and provider hiding the service implementation but still focusing solely on the service usage. Interactions not concerned with usage like problem handling are not considered.

There also exists some work of the TeleManagement Forum on processes needed to manage telecommunication services [10]. The main focus in not the creation of contracts but the automation of business processes. After completion this standardization effort can simplify the specification of contracts.

No work for service contract specification regarding the interactions which support usage and management of services from the customer's domain is known. The approach presented in this paper uses workflow concepts for service contracts to support usage and management of services from the customer's domain and to enable a systematic design of contracts.

4 Workflow Modeling

The most important standard body in the area of workflow technology is the Workflow Management Coalition (WfMC). This organization has defined a reference model [11] and terminology for workflow modeling [12]. Based on this work there exists a graphical representation for workflow models [8], [9], [4] which also allow the syntactical verification of the models. Because UML [6] is commonly accepted and the Object Management Group also specified the representation of workflow graphs [5] which is based on the work of the WfMC, UML should be used in this paper for workflow models. This is no restriction because the graphs can easily be mapped to the other representations.

For the representation of a workflow we use activity graphs (see figure 2) suggested in [5] in a simplified version compared to the activity graphs specified in [6].

Workflow graphs are named at the top left corner of the graph. It is possible to group several parts of the graph in so called *swimlanes* which are separated by thin vertical lines. This can be used for example to group activities to organizational units or to actors referenced by the name at the top of a swimlane.

Activities are represented by rectangles with rounded corners marked with the activity name. The model supports recursion by *composite activities*. Such activities are marked with a small symbol at the right side of the rectangle. A composite activity is modeled as a workflow elsewhere which is references by its name.

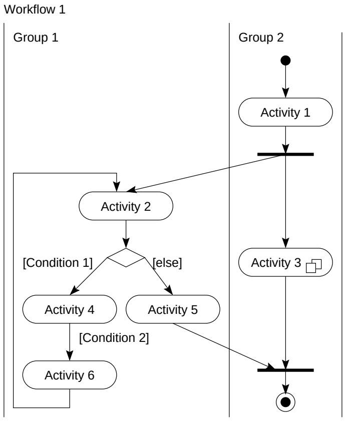

Fig. 2. Activity Diagram for Workflow Models

A workflow starts at a *start symbol*, a small filled circle. It finishes either at an *end symbol*, two circles while the inner one is filled, or at a point where no further action can be executed because all conditions allowing the execution of a path evaluate to false.

Conditions appear in the graph in two ways. They can be explicitly modeled by a condition symbol drawn as a rhombus with conditions in square brackets at each leaving transition arrow. A condition named "[else]" marks the path that should be used if all other conditions evaluate to false. Alternatively, conditions can be placed in square brackets directly at the transition arrow leaving an activity.

A path is any valid "way" through a workflow graph along the transition arrows. More than one path can be active at the same time. *Concurrency* is modeled by synchronization bars which are drawn as thick lines. Several transition arrows leaving a synchronization bar model parallel paths which can be executed independent of each other. If several transition arrows point to one synchronization bar these paths are synchronized at this point, i.e., the execution of the following activity is blocked until all paths arrived at the synchronization bar.

The structure of the graph can be mapped to the WfMC control structures [12] as follows: AND-split as well as AND-join are represented by synchronization bars, while an OR-split is modeled by a condition symbol and OR-joins are modeled by several transition arrows pointing to the same activity or condition symbol.

5 Combination of Workflows and Contracts

A service contract should describe the usage and management interactions between customer domain and provider domain. These interactions, e.g., problem management, can be formalized as processes. Processes like production or business processes are modeled with workflow concepts. Therefore, it is reasonable to use these proven concepts to specify service contracts, too. We use workflow concepts in the design phase as well as for the actual contents of the service contract.

Modeled business processes must be understood by people not familiar with workflow concepts. The same is valid for contract design. Users and several experts of customer and provider domain need to be involved in the specification of technical and organizational parts of the contract. Workflow models are easy to understand and therefore a good basis for communication of people with different knowledge. Furthermore, the resulting workflow graphs are a part of the actual contents of service contracts to support the constructive cooperation in the operation phase.

Each process as well as each activity in a process can have quality criteria which must be guaranteed and therefore defined in the contract. But the quality criteria are usually, due to changing needs, much more dynamic than the processes which do not change for a long time. The contract structure needs to reflect the different levels of dynamics because changing quality criteria is common in long–term relationships.

5.1 Contract Structure

This approach divides the contract in three segments which include the elements of the contract identified in section 3.2: legal information, organizational interactions and technical functionality as well as service levels. The top level of the contract structure is depicted in figure 3.

Basic agreements represent the legal information in the contract and define this structure. Therefore, the basic agreement is the basis of the contract.

The *service agreement* specifies the processes needed to use and manage the service from the customer's domain. It defines the observable service functionality and the management facilities, but not including the service level. The processes are service level independent through the use of variable data and virtual resources. The service management of the provider is not specified but it must support the agreed interactions.

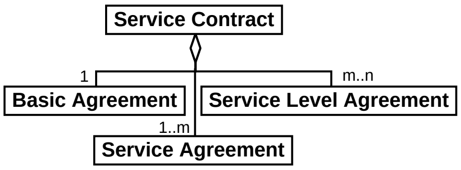

Fig. 3. Structure of a Service Contract

Service level agreements specify the service level by agreeing upon service capacity and quality as well as upon usage and management interfaces for the customer's domain. This is done by specifying concrete values for the variable data and interfaces for the resources of a process. The interfaces must be implemented by the provider, but the way of implementing them is not part of the contract.

5.2 Process Model

The interactions between customer and provider domain are represented by processes which are modeled using workflow concepts. Our process model which is compatible to WfMC terms is depicted in figure 4 includes all relevant process elements and shows their relationships.

Processes consist of *activities*. To enable hierarchical decomposition a process itself is an activity. An activity needs *resources* to be executed on and *data* to work with.

Resources are abstract representatives of *objects* and *roles*. Data are either attributes or variables depending on the point in time of the value assignment. *Attributes* are static data which are defined in the contract. *Variables* are highly dynamic values which are exchanged between the domains at run time.

Activities can generate *events*, e.g., for inter–process communication, synchronization of concurrency or for exception handling. Events are also *triggers* for processes.

The process execution is controlled by the process *logic*. It includes a static part, the *structure*, as well as a dynamic part, the *conditions*. There are three classes of conditions. *Pre-conditions* and *post-conditions* are used to model constraints for the execution of activities. *Selections* are used by the static structure to select a path by a dynamic decision. Events and data of the process as well as system data like current time can be used in all conditions.

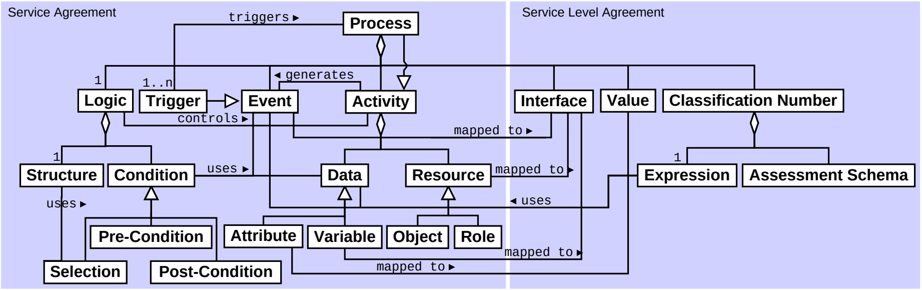

Fig. 4. Process Model

Service agreements contain all this information to model service functionality and management facilities but they do not contain any concrete values for capacity and quality of the service. Service level agreements define this as depicted on the right hand side of figure 4.

Service level agreements map the data needed by activities. Variables are mapped to interfaces to be exchanged at run time, while attributes are mapped to *values* defined in the service level agreement. The resources used by activities are also mapped to interfaces. These *interfaces* hide implementation details. A resource representing a role can be mapped to a technical interface, e.g., defined by a telephone number, or to an individual contact person which is usually done at run time.

An important information in service level agreements are *classification numbers*. In combination with attributes they specify the service level. Classification numbers consist of *expressions* and *assessment schemas* to rate the results. Expressions can refer to variables, attributes and system data as well as events.

5.3 Process Classification

The processes should be classified functionally to deliver an additional level in the structure of service agreements and service level agreements as well as to help identifying all required processes of a service. The process classes shown in table 1 have been identified by combining the TeleManagement Forum's Telecom Operations Map [10] and OSI's Systems Management Functional Areas [3]. The grouping to classes follows a process–oriented contract view mapping processes participating to similar tasks to one class. The class usage completes the classification representing the interactions of the actual purpose of the service.

Some of the processes only exist to allow specification of quality criteria for the service. Classes consisting just of such processes are nevertheless needed in the service contract. The remainder of this section describes the classes in more detail by listing the main processes and typical quality criteria.

Before the service can be used it must be installed. The *provisioning* class includes installation, test, acceptance and if necessary migration processes. Timeliness is an important quality parameter for processes in this class. *Usage* is the most common interaction. Service levels define for example the response time of the service.

Operation processes are invisible to the customer most of the time, but quality parameters are needed for them anyway, for example, the amount of time to detect a malfunction of a network connection is a quality criterion of service operation. As the service quality is usually at least limited during *maintenance* activities the customer wants to limit e.g. the time slot for maintenance.

Table 1. Process Classes

Provisioning	Installation of the service
Usage	Normal usage of the service
Operation	Service management of the provider
Maintenance	Plannable activities needed for service sustainment
Accounting	Collection and processing of accounting data
Security management	Management and observation of service security
Change management	Extension, cancellation and change of service elements
Problem management	Notification, identification and resolution of service malfunction
Contract management	Management of contract contents
Customer care	Service reviews and user support
Deinstallation	Deinstallation of the service

The method used for *accounting* purposes is an important fact because it influences the price of the service. The processes of the *security management* class include mechanisms to observe the service security and to manage the means for authentication of service access.

Minor changes of the service are not uncommon in long–term relationships. This could be the extension of resources or addition of further dealers in the presented scenario. Thus, a *change management* is necessary. A well working *problem management* is a very important factor in outsourcing. Problems can be solved faster if both partners are working together smoothly using well defined processes.

Often, requirements of customers change during contract lifetime. If a change is predictable but not the point in time it is useful to specify *contract management* processes for announcing the need, negotiating and changing the contract. Communication on strengths and weaknesses of the service, its quality or future optimizations should be some of the subjects of regular *customer care* contacts. Besides, a help desk or other support offers are usually necessary for complex services.

Deinstallation processes usually need to be specified if equipment of the provider is installed at customer locations or if support of the provider is needed for migration to the following service.

Additionally, it is useful to define basic processes which are needed in many classes. Such processes are e.g., documentation, feedback and escalation mechanisms.

6 Contract Design

The basis of service contract design are the processes of the customer domain which use and manage the service. The operation processes of the provider must fulfill the contract, but their implementation is not an element of the contract. If a service was implemented by the customer before the current management processes focus too much on implementation issues. Therefore, the management processes of the customer need to be redesigned when outsourcing a service for the first time.

6.1 Design Process

To explain the application of our approach on contract design a simple design process is presented. It shows the necessary steps without detailing their execution. In practice it is an iterative process but the order of the steps in each iteration is the same. The process model explained in section 4 is the basis of this process.

First of all, the process classes needed for the specific service must be identified. Then, the design process depicted in figure 5 is executed for each class. As long as a new process is found (1) the outer iteration (2) is repeated. Hierarchical decomposition of processes is possible because each activity can be modeled as a process.

The left hand side concurrent path (3) defines the triggers of the process (4). The second of the parallel paths (5) builds the process logic with the inner iteration (6). As long as an activity can be identified (7) the activity, the necessary control structures and conditions for the dynamic aspects are inserted in the logic (8). Then, the needed data (9) and resources (10) are specified. Thereafter, the loop is repeated (11) to identify the next activity. When execution of both paths of the outer concurrency is finished (12) the iteration is restarted to identify the next process.

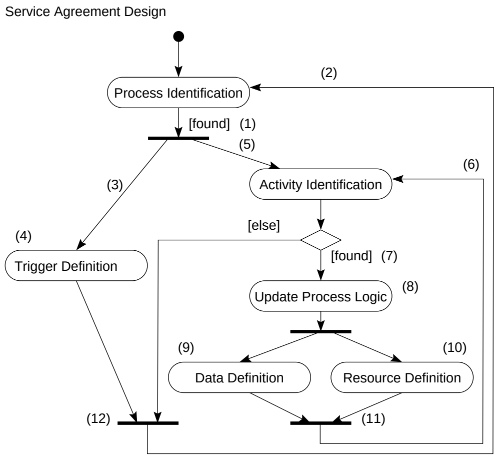

Fig. 5. Design of the Service Agreement

For the specification of the service level agreement the abstract resources, attributes and variables must be mapped. The mapping of resources to specific interfaces can influence the possible service level and depends on the infrastructure of the customer domain. Therefore, the service level is influenced by interfaces, values of attributes and for complexer capacity or quality aspects, which cannot be expressed by a single value, classification numbers must be specified.

Such classification numbers are also needed for quality criteria covering several activities but not the whole process. The expression of the classification number calculates a value for one aspect of the service level. This value is rated by the assessment schema. A schema is just required for guaranteed classification numbers. Some of the classification numbers can be declared dynamic and a process for their change can be agreed upon to enable the controlled adaption of the contract to changing needs or technology during its lifetime.

6.2 Example of a Modeled Process

The design process is demonstrated in a simplified example from the scenario presented in section 2. The example model is a part of the problem management class. It defines the process of fault identification, notification and resolution (see figure 6). The models of the complete process including cooperation during fault identification, documentation, prioritization, exception handling, escalation mechanisms, etc. would go beyond the scope of this paper. During identification of different activities the necessary resources and data must be defined which is also not detailed in this example.

A fault can be detected by both, customer and provider. Therefore, two start symbols exist. Depending on the party that detects the fault the process starts with the *fault detection* on the customer or the provider side. If the customer detects a fault, the next activity is the *notification* of the provider. If the provider detects it the customer is notified by the *customer notification* process. The customer notification is modeled as a composite task because it is a basic process controlled by dynamic values like the message and the receiving role to adapt it to the respective situation.

At this point the two paths join. The next activity is *trouble ticket creation*. Then, the concurrency construct models the fault resolution with regular feedback to the customer on the current state. The feedback path is modeled as a loop monitoring the trouble ticket state. As long as it is open the *customer notification* process is repeated. The feedback interval is realized by a post-condition.

The fault resolution path on the right hand side starts with the *fault identification* activity. This activity is separated from the *fault resolution* because the customer usually wants feedback on the estimated duration and scope of the service malfunction as soon as possible which therefore is modeled explicitly by the *customer notification* base process.

The identification and resolution activities by itself are of no interest to the customer, but he wants to define quality criteria on this activities. Therefore, they must be modeled. After the problem is solved the *trouble ticket* is *closed* which

ends the feedback loop and the customer is notified of the resolution in the final *customer notification* process. The resulting process model including the data and resource definitions is part of the service agreement.

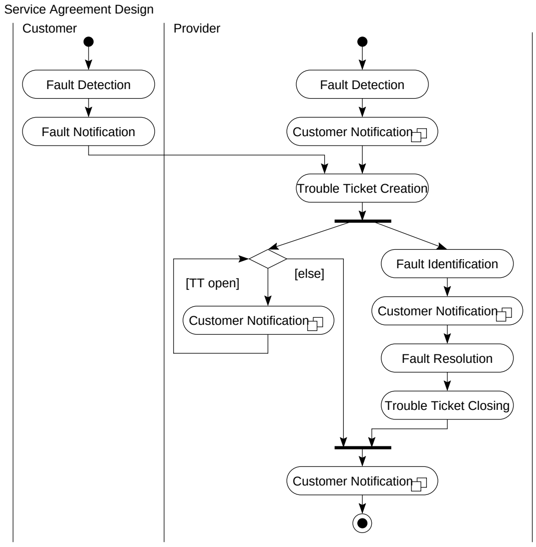

Fig. 6. Fault Notification Process

In the next steps the service level agreements are defined by mapping data and resources to values and interfaces. The necessary interfaces are easy to find because each interaction crossing a domain boundary is visible in the workflow graph when the control flow crosses a swimlane boundary. The only new interface added by this process is the fault notification interface which could be mapped to a telephone interface. The interfaces of the composite tasks are defined in the respective workflow model.

All requirements for the service level not expressed by now must be specified as classification numbers. In the presented example process a classification number on the total time from trouble ticket creation to trouble ticket closing should be covered by a classification number. Additionally, to limit the amount of service failures, e.g., a classification number for the number of trouble tickets open simultaneously and in certain intervals is needed. Several other classification numbers are thinkable, too.

7 Analyzing the Use of Processes in Contracts

The idea to fulfill the raised requirements is to use concepts of workflow modeling during the design phase and to use the resulting workflow graphs as part of the actual service contract. This is a promising approach because workflow modeling is usually done by gathering knowledge of various sources which is similar to the creation of service contracts. There is not a single expert for all service details of networks, servers, applications, help desks, logistics, etc. The concepts of workflows and the resulting models are easy to understand. Thus, it is possible to model a service contract with workflow concepts while keeping the understandability for people that can give input or have to work with the resulting service contract. The workflow graphs support the information exchange and therefore support cooperation in the design of the service.

The resulting models in the service contract serve as a manual for users and operators. It provides concrete and constructive instructions supporting usage and operation of the service, which improves the cooperation between outsourcing partners. This optimizes the handling of critical situations as it tells affected persons how to become active.

Concepts of workflows have been a research subject for many years now and have already been applied successfully by many enterprises to reengineer and optimize their business processes. So workflow modeling is a proven concept which we are adapting to contract design and writing. Describing the service contract in the same way as the processes in service management also simplifies the mapping of the contract to the provider's service management significantly. Furthermore, the explicit statement of resources, interfaces, competence and information flows supports automation efforts and optimization of resource consumption.

The systematic identification of tasks along the timeline results in more consistent as well as complete and therefore higher quality service contracts. All technical parameters are easier to identify when analyzing the usage process along the timeline.

Furthermore, the process models are highly reusable. They can be reused in higher level processes, further contracts or for other purposes like optimization efforts. Workflow models are precise enough to specify a contract by arbitrarily refining the tasks, but it is also easy to abstract from well known facts and unimportant details. Another feature of workflow models is the possibility to verify them syntactically [8], [4], i.e., to verify the syntactic consistency of a contract specified by a workflow model.

The use of the customer view in form of the customer's business processes results in a customer–centric contract. Additionally, it is service–oriented because the customer's processes focus on usage and not on implementation. The differentiation of the contract in several segments facilitates the controlled dynamic adaption of the contents during contract lifetime. This enables the revision of quality aspects independent from functionality allowing to shorten the negotiation process substantially and therefore enhances the flexibility.

The shortcoming of the approach is that the creation of a contract according to this approach requires lots of effort. But a relevant amount of effort is acceptable for the difficult task of specifying a good contract for a complex service with a contract lifetime of several years.

8 Conclusion and Future Work

This paper proposes the application of workflow concepts for the specification of contracts concerning IT services. The main points of the approach discussed are that it supports the identified requirements for long–term service contracts which need to be customer–centric, dynamic and constructive. Supplementary, this approach generates some additional advantages, like the systematic design of service contracts. The most attractive benefit of the workflow concepts is the active support of cooperation for usage and operation resulting from the instructive nature of workflows. Additional features like controlled adaption are partly enabled by the presented contract structure.

The approach is currently verified in an industry cooperation with a major provider of telecommunication services. The necessary effort sets the main focus especially to extensive, custom–designed services. But we are researching ways to simplify the reuse of this effort.

Further research is concerned with substantiating the application methodology and extraction of characteristic capacity and quality parameters. The target is to identify a methodology for the systematic derivation of customer–oriented but measurable quality parameters.

Acknowledgment

The author wishes 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. P. Bhoj, S. Singhal, and S. Chutani. SLA Management in Federated Environments. In M. Sloman, S. Mazumdar, and E. Lupo, editors, *Integrated Network Management VI (IM'99)*, Boston, MA, May 1999. IEEE Publishing.
2. 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. 651 p.
3. Information Technology – Open Systems Interconnection – Systems Management Overview. IS 10040, International Organization for Standardization and International Electrotechnical Committee, 1992.
4. A. Karamanolis, D. Giannakopoukou, J. Magee, and S. Weater. Modelling and Analysis of Workflow Processes. Technical Report DTR99-2, Imperial College of Science, 1999.
5. Business Process Modeling with UML. TC Document ad/00-02-04, Object Management Group, February 2000.
6. Unified Modeling Language (UML) 1.3 specification. OMG Specification formal/00-03-01, Object Management Group, March 2000.
7. T. Preuß and H. König. Service Supplier Relations for the Outsourcing of Information Processing Services. In *Proceedings of the IEEE Enterprise Networking and Computing Conference (ENCOM 98)*, Atlanta, GA, USA, 1998.
8. W. Sadiq and M. Orlowska. Modeling and Verification of Workflow Graphs. Technical Report 386, University of Queensland, November 1996.
9. W. Sadiq and M. Orlowska. On Capturing Process Requirements of Workflow Based Business Information Systems. In *Proceedings of the 3rd International Conference on Business Information Systems (BIS99)*, Poznan, Poland, April 1999.
10. SMART TMN Telecom Operations Map. Evaluation Version 1.1 GB910, TeleManagement Forum, April 1999.
11. The Workflow Reference Model. TC Document 00-1003, Workflow Management Coalition, January 1995.
12. Terminology & Glossary. TC Document 1011, Workflow Management Coalition, February 1999.