Towards Method Fragments for Service Engineering

Michael Becker[1], Stephan Klingnger[1]

[1]InfAI e.V.

To overcome the challenges resulting from the proliferation of service engineering methods, we present a method engineering approach for service development and provision. Based on method fragments, i.e. coherent pieces of a method, a method can be assembled according to characteristics of a service project. Using a design science approach, we develop a metamodel for service engineering method fragments with the focus on service customisation. In addition, a service customisation information system is conceptualised.

1. Introduction

Today's service industry faces two distinct challenges. One the one hand, companies need to provide services as efficient as possible. On the other hand, the diversity of customer requirements increases and companies need to continuously improve their services (Heiskala, et al., 2005). Service engineering tries to tackle this dichotomy by providing approaches for a systematic service development. In doing so, process models, methods, and tools are developed (Fähnrich & Opitz, 2006). Although these approaches for systematic service development and provision exist, still some 40 percentages of service projects fail (Leimeister, 2012, p. 94). This is caused by the fact that companies are both not aware of service engineering approach and are not able to use provided methods and tools in their specific service project (Fähnrich & Meiren, 2007; Zhou & Tan, 2008).

The existing challenges can be ascribed to the fact that companies observe a need for action concerning the configuration of service engineering approaches (Uhrmann-Nowak, 2010). Due to the large heterogeneity of the service domain (Münkhoff, 2013), it is not feasible to use a one-size-fits-all approach for service development and provision. Instead, the used methods and tools need to be chosen and adapted to specific service project characteristics.

To address the mentioned challenges, we present a method engineering approach for service engineering. Method engineering is founded in information systems development and aims at designing, constructing, and adapting development methods, techniques, and tools (Brinkkemper, 1996, p. 276). It is based on so-called method fragments that are combined according to the individual characteristics of a development project.

The research process of this paper follows a design science approach according to Pfeffers, et al. (2007) and is depicted in Fig. 1 (the current state of the research is marked with an asterisk). The design science research lifecycle starts with identifying and motivating a problem. The motivation for the research at hand was derived from

previous research projects and from indications for research gaps in academic literature. As a second step, the objectives for a solution need to be defined. We follow the design science guidelines established by Hevner, et al. (2004), i.e. artefacts need to be produced, and the research must have practical relevance and provide new contributions for solving the problem. In addition, the research must follow a must follow a rigor and iterative process with continuous improvement of intermediate results. Finally, results of the research must be communicated.

After this preparatory work, artefacts as the central element of design science research projects are defined. According to the design science guidelines, artefacts are produced iteratively which is reflected by the cycle between *Method Fragment Development* and *Evaluation*. During construction of research results, intermediate results are evaluated by demonstrating their usability and by pointing out the way that produced artefacts contribute to the problem. A common approach for demonstrating the usability of theoretically designed artefacts is to develop a real world application using these artefacts. In addition, evaluation can be conducted in the form of expert panels or by assessing productivity gains. Each method fragment that was positively evaluated is formalised and, thus, can be used as part of a service engineering support system.

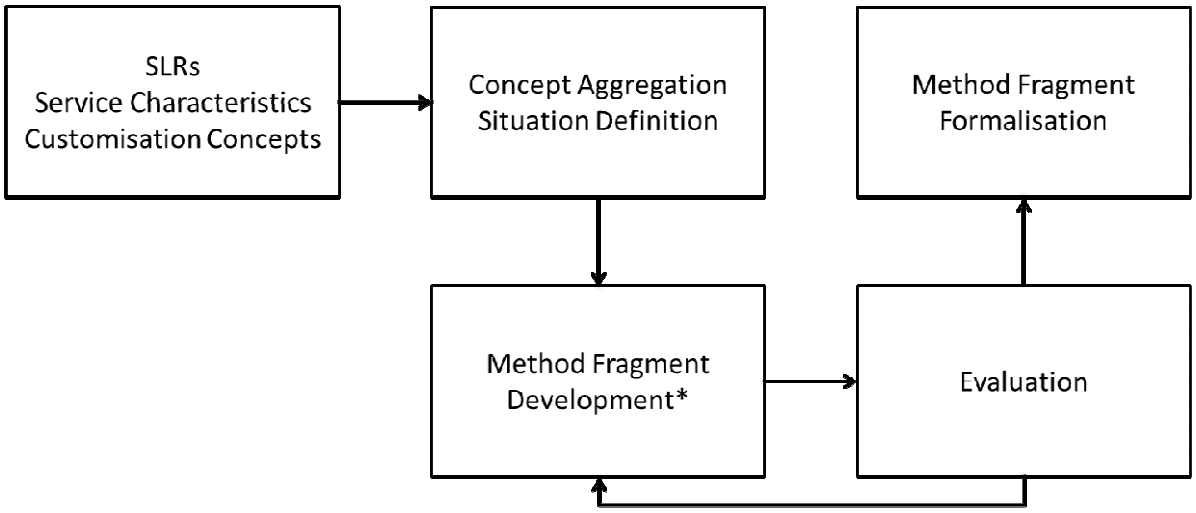

Fig. 1: Research process

For presenting our findings, the paper is structured as follows. In the next section we introduce the theoretical backgrounds concerning method engineering. As preparatory work, structured literature reviews (SLRs) for identifying service characteristics and service customisation concepts were conducted and are presented in section 3. For developing coherent method fragments, section 4 presents a metamodel for service engineering method fragments. Based on this metamodel, we present the usage of method fragments in section 5. Finally, section 6 concludes the paper with an outlook on future research and provides insights about applying service engineering method fragments.

2. Method Engineering

The development of method engineering approaches was triggered by the increased complexity of software engineering projects and the insight that there is no single suitable engineering method for conducting heterogeneous projects (Brooks, 1987). According to Tolvanen (1998), three approaches for selecting an engineering method exist. First, using the textbook approach, a single method is used and performed as described. Second, using a contingency based approach, suitable parts of various methods are combined. However, the methods do not provide techniques to assemble a holistic process. Contrary, the method engineering approach allows for individual configuration of a method according to project characteristics. For doing so, the characteristics of a project are described in terms of a so-called situation and method fragments appropriate for this situation are selected and combined (Tolvanen, 1998).

According to Brinkkemper (1996), method fragments are defined as “coherent pieces of […] development methods”. To define method fragments, a two-step approach is used: The first step is to identify suitable methods fragments. For doing so, existing methodologies can be used and decomposed into independent parts. In addition, companies might use best practices as a starting point for defining method fragments. The following key questions can be used as a guideline to identify suitable method fragments (Deneckere, et al., 2008):

- *What is the result when using a method fragment?* This answers the question about the contribution of a method fragment to the overall project. It is used to define the fragment’s goal.
- *What are the preconditions to use a method fragment?* Often, it is necessary to conduct preliminary work to execute a part of a method. In these cases, it might be necessary to establish organisational or technical preconditions.
- *In which specific contexts can the method fragment be used?* A very important aspect of method engineering is to describe the situation, in which a method fragment is applicable. In doing so, it is possible to identify relevant fragments. An existing method might be tailored for a specific use case and is, thus, only applicable for a particular project type.
- *What are the constituent parts of the method fragment (activities, models, artefacts etc.)?* To achieve the goal associated with the method fragment, it is necessary to perform several activities, e.g. gather customer requirements. In addition, a method can provide specific modelling notations to represent the results, e.g. the requirements might be represented in terms of use cases.
- *How is the application of the method fragment supported?* Besides describing necessary activities, a method might also provide or reference software tools that support executing the activities. In addition, existing documentation (e.g. standards) can be used.
- *What relations and dependencies with other method fragments exist?* Besides preconditions to perform a method fragment, other dependencies might also exist as well. For example, an existing method might contain decisions that lead to mutually exclusive activities. While identifying method fragments, it is

necessary to keep track of these dependencies for establishing a consistent method repository.

As a second step, the method fragments need to be specified according to a coherent predefined structure. On the one hand, a unified, formally defined structure allows for efficient method fragment search. On the other hand, the structure is necessary to assemble method fragments into a holistic method by using established interface descriptions. To facilitate search and simplify usage, Karlsson & Wistrand (2006) differentiate between an internal and an external view on method fragments. The external view contains a descriptor for identifying the method fragment and the interface description to specify the suitable situation and goals. In terms of the internal view, the content of the method fragment is described in terms of a so-called guideline and product aspects. The guideline describes the activities that are necessary to perform a method fragment and, thus, to achieve a specific goal. In addition, the product aspect describes used tools and formalisations.

The general method engineering process can be depicted in Fig. 2. The *method base* contains the set of the predefined *method fragments* (MF1, MF2, and MF3). Each method fragment is suitable for a specific situation, denoted by service characteristics. In the example, the two binary characteristics *C1* and *C2* exist. The method base serves as a repository for method fragments: method fragments can be added, updated, and removed. For the provision of a service project, it is necessary to define the characteristics of the service project. Therefore, the same characteristics *C1* and *C2* are used. In *Project A*, both characteristics are defined, for *Project B* the value of *C1* is unclear. This might be due to the fact that the project is very innovative and, thus, vague.

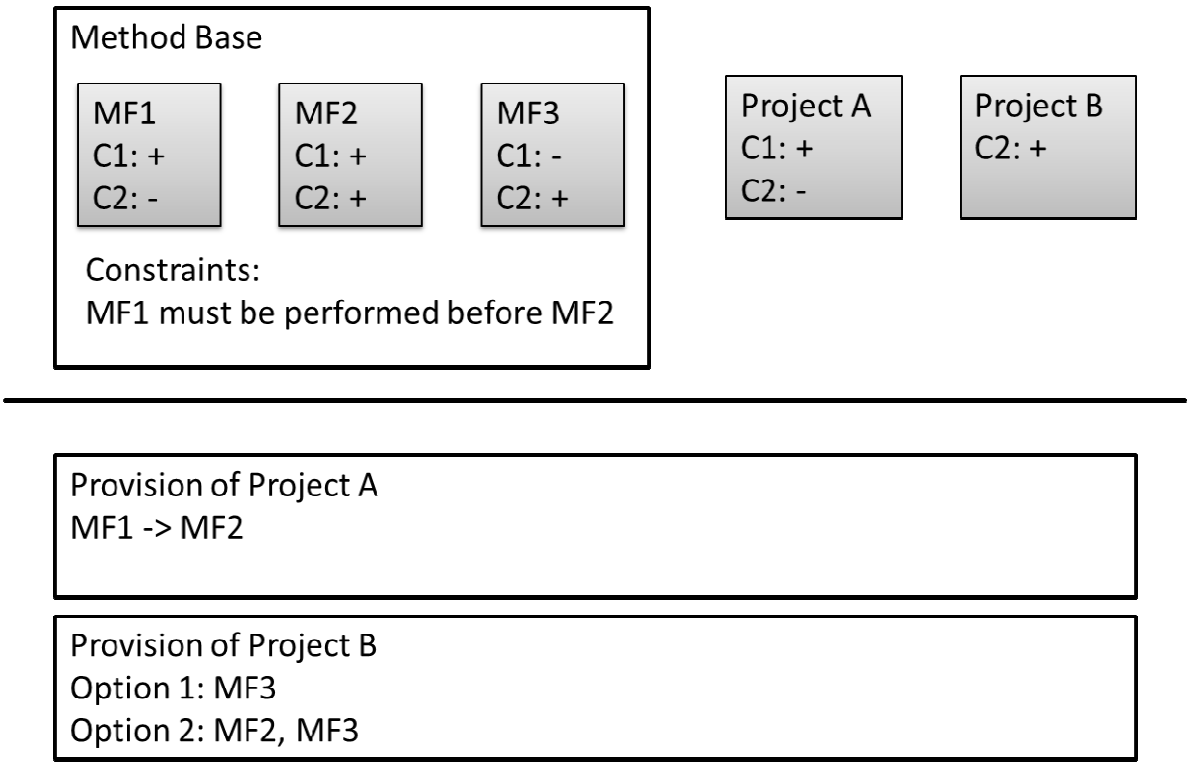

Fig. 2: Method engineering process

A project is matched with suitable method fragments according to its characteristics. Thus, relevant method fragments for *Project A* are *MF1* and *MF2*. To assemble these method fragments into a method, dependencies need to be considered. In the example, there exists a dependency *MF1 must be performed before MF2* resulting in the sequential order of both fragments. In the case of *Project B*, two relevant sets of method fragments exist due to the fact that the value of characteristic *C1* is not known. Relevant sets can either contain only method fragment *MF3* or both method fragments *MF2* and *MF3*. Selection appropriate method fragments is the task of the project manager. Likewise to *Project A*, relevant fragments for *Project B* are assembled. Since no temporal dependencies between method fragments *MF2* and *MF3* is defined, both fragments can be performed in arbitrary order.

3. Service characteristics and customisation approaches

For answering the above presented questions and to establish a sound metamodel founded in the current state of the art concerning service customisation, two systematic literature reviews were conducted. First, service characteristics were identified. Based on these characteristics, it is possible to answer questions about the context in which a customisation method fragment can be applied. With the second literature review, approaches for customising services were identified. Based on these approaches, an aggregated overview about customisation concepts was established.

3.1. Service characteristics for describing the situation

For valid application of method fragments, it is necessary to specify suitable situations. Therefore, a structured literature review was conducted and the results were presented in Becker, et al. (2011). Table 1 presents an extract of the most relevant identified service characteristics.

The characteristics were divided into three classes: *customer interface*, *service process*, and *service outcome*. Characteristics of the customer interface class describe the interaction between service providers and customers. With the help of these characteristics, it is possible to describe the impacts of customer decisions on service provision. Process characteristics are mostly relevant for service providers and influence temporal and local service factors. The characteristics of the service outcome allow for describing services from a customer's point of view.

Characteristics of the customer interface	
Customer contact	Indicates the customer involvement into service provision. It can be distinguished between active contact (customers are directly involved in service provision) and passive contact (customers only need to be on-site).
Relation between customer and provider	The relation can be formally established (e.g. Service Level Agreements) or based on informal agreements (e.g. honorary offices). It can be distinguished between continuous

	service provision and singular provision.
Customer interaction	Customers and providers can interact via a human or a machine interface. Additionally, service provision can either be bound to a specific location (e.g. ATMs) or can be location-independent (e.g. Online Banking).
Characteristics of the service process	
Flexibility	A service process might either be rigid (e.g. due to legal restrictions) or flexible. Flexibility can either be achieved by predefined service variation points or by ad-hoc customisation of employees.
Technology usage and degree of routine work	Several service process steps can be supported or even fully provided by IT. In addition, the complexity of used technologies is relevant. The degree of routine work is closely connected with technology usage, since routinized work can be supported by IT systems more easily.
Decoupling	Decoupling of the service provision impacts the locations where a service can be provided. For example, services requiring heavy machinery are bound to a specific location.
Characteristics of the service outcome	
Materiality	Services with a material result (e.g. ordering customised jewellery) can be distinguished from services with immaterial results (e.g. education). On the one hand, materiality has great impact on the way customers can evaluate the service. On the other hand, different competencies are necessary for providing services with material outcome compared to providing services with immaterial outcome.
Service recipient	Services might either address people (e.g. education changing a person's mental state, person transport changing a person's location), things (e.g. transporting goods changing the location of things), or information (e.g. Big-Data-Services processing data). In addition, services can address time aspects (e.g. maintenance reducing wear and tear) or location aspects (e.g. providing food during a long flight with specific location-based constraints).

Table 1: Relevant service characteristics, source: (Becker, et al., 2011)

3.2. Customisation approaches for describing method fragments

For specifying method fragments, it is first of all necessary to identify methods and parts of methods that can be decomposed into fragments. For identifying relevant methods, a structured literature review about service customisation was conducted (Becker & Klingner, 2016).

Due to the heterogeneity of the service domain, no unified terminology for describing customisation approaches exists. Thus, the following classification is used to build a hierarchic structure for the different customisation approaches:

- *Concept instances* are the specific terms that are used in academic literature to describe a customisation approach. For example, the concept instances *building blocks*, *modules*, and *service configuration items* were identified.
- Concept instances with identical semantics were aggregated into *concepts*. For example, the above mentioned instances are assigned to the concept *components*.
- For additional aggregation of concepts, *generalised concepts* were established. They contain adjacent concepts, e.g. the concepts *portfolio*, *components*, *interfaces*, and *variants* are assigned to the generalised concept *modularisation elements*.
- *Classes* are used to define different views on the service development and provision process. For example, the generalised concepts *modularisation elements*, *characteristics*, and *dependencies* are assigned to the class *representation*.

In addition to the classification, each concept has a specific type constraining the transition from concepts to method fragments. The possible types are depicted in Table 2.

Concept type	**Description and Examples**
Service modelling and description	Concepts of this type are necessary for describing the variability of customisable services. The spectrum comprises simple textual descriptions as well as formal specification of variation points. Concepts like *components*, *interfaces*, and *dependencies* are examples of modelling and description concepts. They are used to model the internal view of a service.
Customisation process	Concepts of this type describe specific activities for customising services. In general, these concepts can be used for defining the guideline of a method fragment. Concepts like *additive customisation*, *product bundling*, and *alternative-based* customisation are examples for customisation process concepts. They are used as guidelines for employees during service provision
Influence factors	Concepts of this type represent decisions during service customisation and as well as external influence factors. They are usually used to further specify the situation a method fragment can be used in. Concepts like *context*, *time pressure*, and *target audience* are examples for influence factors.

Table 2: Concept types

4. Metamodel for Service Engineering Method Fragments

Based on the general method fragment structure presented in the previous section, a metamodel for defining service engineering method fragments was developed. The constituent parts of the metamodel can be seen in Fig. 3. Every method fragment consists of a descriptor, an interface, the guideline, and a product aspect. In the following sections, the development of the metamodel is presented. First, section 4.1 outlines the external view by specifying the structure of a method fragment with respect of specific requirements of the service engineering domain. The internal view of method fragments is presented in section 4.2. A formal definition of the internal views allows for a unified representation of method fragments and, thus, enables simple combination of fragments.

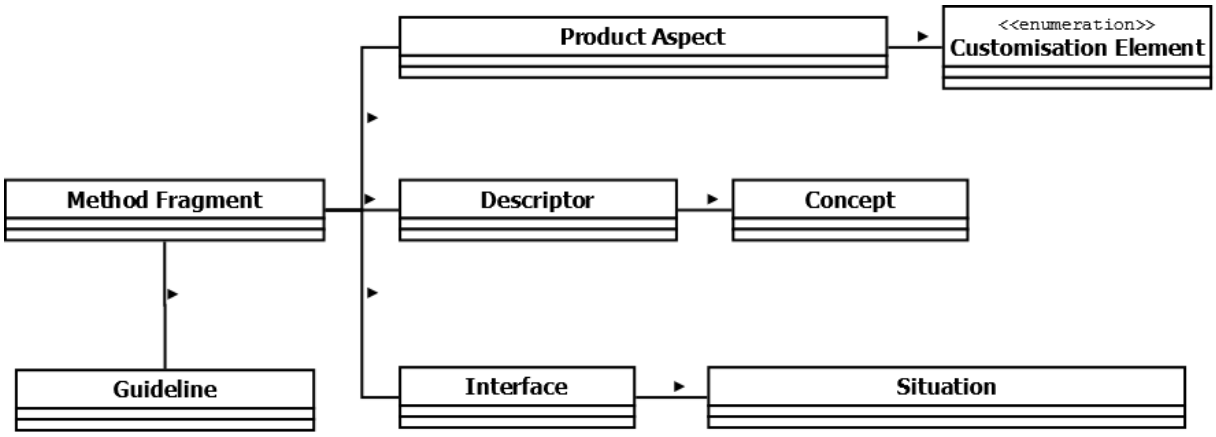

Fig. 3: Metamodel for Service Engineering Method Fragments

4.1. Metamodel of the external view

The external view of a method fragment as shown in Fig. 4**Errore. L'origine riferimento non è stata trovata.** consists of the descriptor and the product aspect. In addition, the interface is used to describe the situation where a method fragment can be applied. For being able to define a method base as a collection of method fragments, dependencies can be specified, too.

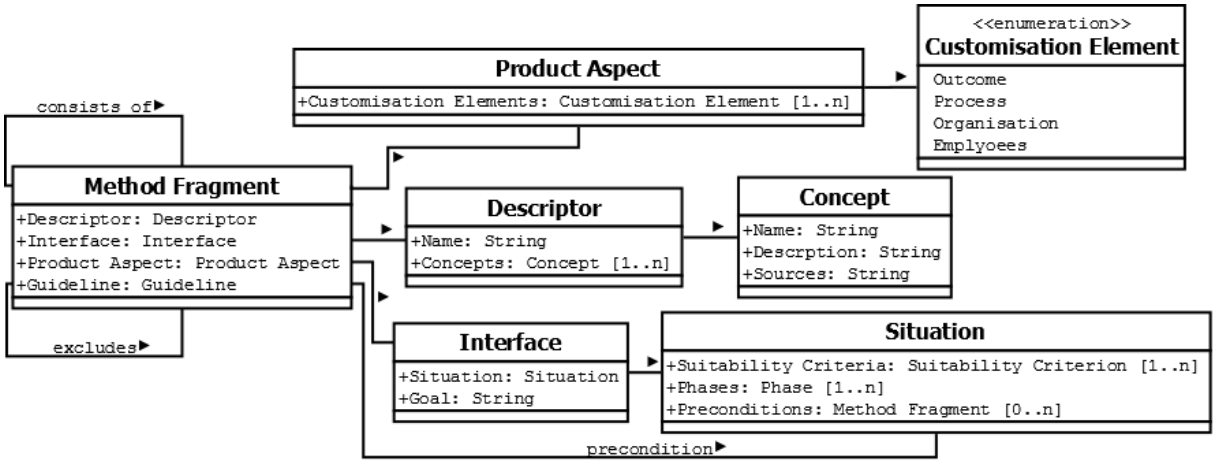

Fig. 4: Metamodel of the external view

4.1.1. Descriptor

Based on the descriptor, a method fragment is uniquely identified as part of a method base. For doing so, every method fragment has a unique name. Besides identification, method fragments are linked with the fundamental concepts from the structured literature review (see section 3). Based on this reference, further information can be obtained, e.g. use cases and evaluations.

As can be seen in Fig. 4, a method fragment can be founded in more than one concept from literature. This is due to the fact that there might exist several ways to achieve the goal of a method fragment. For example, the method fragment *module based customisation* has two variants, *additive customisation* and *subtractive customisation*.

4.1.2. Interface

The interface of a method fragment is used both for describing the specific situation this fragment can be applied as well as describing the goal of the method fragment. For describing the situation the service characteristics presented in section 3 are used. Thus, the situation describes applicability based on the customer interface, the service process, and the service outcome.

Besides suitable service characteristics, the application of a method fragment is further bound to a specific service lifecycle phase. Using the metamodel, the service lifecycle is specified according to DIN (1998) by the six phases *idea generation and assessment, requirements analysis*, *design*, *implementation*, *provision*, and *termination*. The relevant phase of a method fragment is considered for assembling an individual method based on sequential combination of selected method fragments. However, it is necessary to note, that there might also exist fragments that are relevant in different phases and, thus, can be applied at different points of the service lifecycle.

To further specify suitable situations for applying a method fragment, it is necessary to describe preconditions that must be satisfied. A precondition might be either technical or substantial. Using technical preconditions, it is possible to describe preparatory work that is necessary to apply a method fragment. For example, the method fragment *module based customisation* requires applying the method fragment *portfo-*

lio modularisation. Without having defined module, it is not possible to offer modules. Substantial preconditions are used to link two or more method fragments with each other that aim at a similar goal. For example, the method fragment *interpersonal customisation* has the precondition *customer model development* which simplifies applying the fragment by defining relevant tools and models.

The goal of a method fragment is described using textual form. Due to heterogeneity of possible goals, a reasonable formalisation is not possible. However, the description of goals should adhere to a consistent structure, e.g. by describing preconditions. For example, the goal of the method fragment *interpersonal customisation* is *employees adapt their appearance and behaviour to customer requirements*.

4.1.3. Product aspect

The product aspect is defined by the service customisation elements addressed by a method fragment. As stated in section 3, the service outcome, service process, company, and employees are possible elements. By using the product aspect, method fragments can be categorised. For example, companies seeking to modify their organisational structure for customisation can select the appropriate method fragments.

4.1.4. Dependencies

Besides the above described preconditions for applying a method fragment, additional dependencies between method fragments might exist. Using these dependencies it is possible to define compound method fragments. In addition, method fragments can be marked as mutually exclusive. Compound method fragments allow for defining approved combinations. For example, the method fragment *interpersonal customisation* suggests the method fragment *customer model development*. To improve convenience of the method base, it is possible to aggregate both method fragments into a new fragment, *model based interpersonal customisation*. Interaction effects that occur due to aggregation can be described in terms of a new interface and guideline of the aggregated method fragment.

Contrary to compound methods, mutual exclusiveness of method fragments defines two or more method fragments that must not be used together. Usually, this is based on incompatible foundational concepts of the fragments. For example, the method fragments *module based customisation* and *option based customisation* are mutually exclusive. While the former is based on customer-individual combination of modules, the latter is based on providing several service variants that a customer can select.

4.2. Metamodel of the internal view

The main part of the internal view is characterised by the guideline. Using the guideline, the application of the method fragment is described in detail. In the following, an extensible metamodel of the internal view is developed. Using the metamodel, different types of guidelines can be used. Further, the metamodel unifies the terminology used in the method base. As can be seen in Fig. 5, a guideline consists of two main elements. While *roles* are responsible for developing and providing services, *artefacts* are necessary tools for applying method fragments.

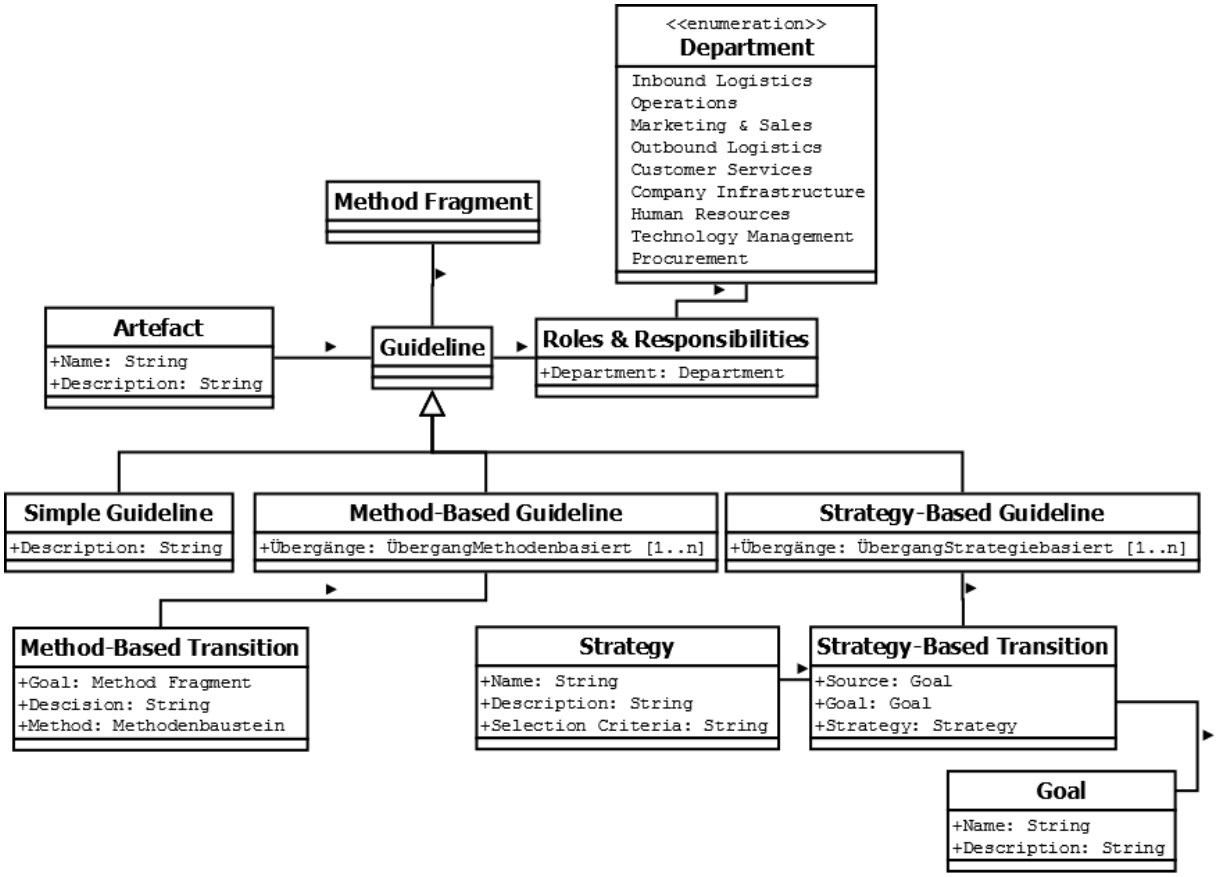

Fig. 5: Metamodel of the internal view

4.2.1. Roles and responsibilities

Using roles and responsibilities, it is possible to describe persons, IT systems, or machines that are responsible for providing several activities of a method fragment. Due to the abstraction level of method fragments and heterogeneity of the service sector, it is not feasible to unify role names for all companies. Instead, a generic role concept is used based on the value chain defined by Porter (2010, p. 64). According to this value chain, companies perform primary and secondary activities. Primary activities are directly connected to producing goods and providing services. They comprise *inbound logistics*, *operations*, *marketing & sales*, *outbound logistics*, and *customer service* (Porter, 2010, p. 66). Secondary activities are of supportive nature and comprise *company infrastructure*, *human resources*, *technology management*, and *procurement* (Porter, 2010, p. 66). Due to the importance of the external factor for services (Meyer, et al., 2000) customers are integrated into the role concept, too.

For assigning segments of the value chain to method fragments, the so-called RACI matrix is used. For every method fragment, it is stated who is responsible for providing activities (*Responsible*) and who approves activities (*Accountable*). In addition, it is stated who has additional information for completing activities (*Consulted*) and who needs to be notified when a method fragment is applied (*Informed*) (Jacka & Keller, 2012, p. 258).

An example for assigning responsibilities using the RACI matrix is given in Table 3. As can be seen, the customer service and operations are responsible for providing the method fragment interpersonal customisation. Both of these divisions have both

direct customer contact and are responsible for providing services. Thus, employees with these roles need to be able to perform the customisation. In case of enquiries, marketing and human resources are consulted. On the one hand, marketing provides customer databases containing information about demands and peculiarities of regular customers. On the other hand, the human resources department is responsible for training and education of employees concerning interpersonal customisation. In addition, marketing is informed about applying customisation activities for keeping track and archiving these activities in customer databases. Finally, marketing is also accountable, i.e. the department approves and defines valid customisation activities.

Department/Responsibility	Responsible	Accountable	Consulted	Informed
Inbound logistics				
Operations	X			
Marketing & Sales		X	X	X
Outbound logistics				
Customer service	X			
Company infrastructure				
Human resources			X	
Technology management				
Procurement				

Table 3: RACI matrix for method fragment interpersonal customisation

Attention should be paid to the fact that the RACI approach cannot always be used in its entire form in every company and for every service. For example, especially small companies do not separate responsibility in different departments. In this case, different responsibilities coincide and might become superfluous. In addition, the RACI matrix differs between different types of services based on the specific characteristics. For example, the service withdraw money can be provided either by an ATM or by a human employee. If the service is provided by an ATM, the operations department is no longer responsible for interpersonal customisation because the user interface is fixed. As a possible solution, a method fragment might have different RACI matrices based on the respective service characteristics.

4.2.2. Artefacts

Artefacts are all kinds of elements that are used to perform customization during service development and provision. Examples for artefacts are customer data that are used to make decisions or employee handbooks containing detailed information about service provision. The artefacts are usually inferred from the product aspect, e.g. method fragments with the product aspect *organisation* require the artefact *organigram*.

4.2.3. Guideline

To describe how to perform a method fragment, the guideline references roles, responsibilities, and artefacts. The guideline can be defined in several ways, which is reflected as subclasses in Fig. 5. The *simple guideline* is the most basic version and describes the content of a method fragment purely textual without formal referencing other method fragments. Contrary, *method-based* and *strategy-based guidelines* use so-called *transitions* for defining how to achieve specific goals and to allow for a more formal definition of the method fragment application.

Using the method-based guideline, a superior goal that should be achieved by performing a specific method fragment is defined. Usually, various approaches for achieving a goal exist. Every approach is more or less suitable based on different influence factors. To enable selection of a suitable method fragment, the decision criteria are defined. The transition of a method-based guideline is, thus, defined as a triple (*goal*, *decision*, *method*).

An example for a method-based guideline is depicted in Fig. 6. It shows the method fragment *module-based customisation* with the alternatives *additive customisation* and *subtractive customisation*. The customer state is used as the decision criterion: for regular customers, the method fragment *subtractive customisation* is used; customisation for new customers is performed using the method fragment *additive customisation*.

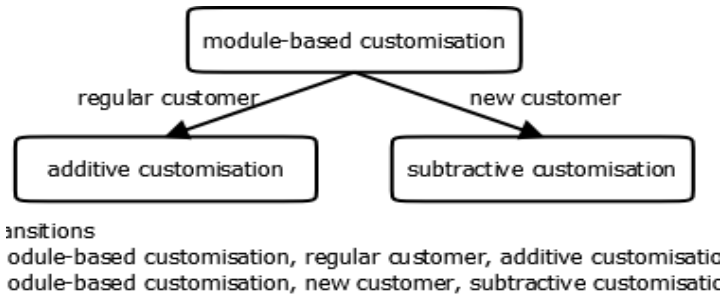

Fig. 6: Method-based guideline for module-based customisation

Based on this definition, Fig. 7 shows the representation of this guideline conforming to the metamodel.

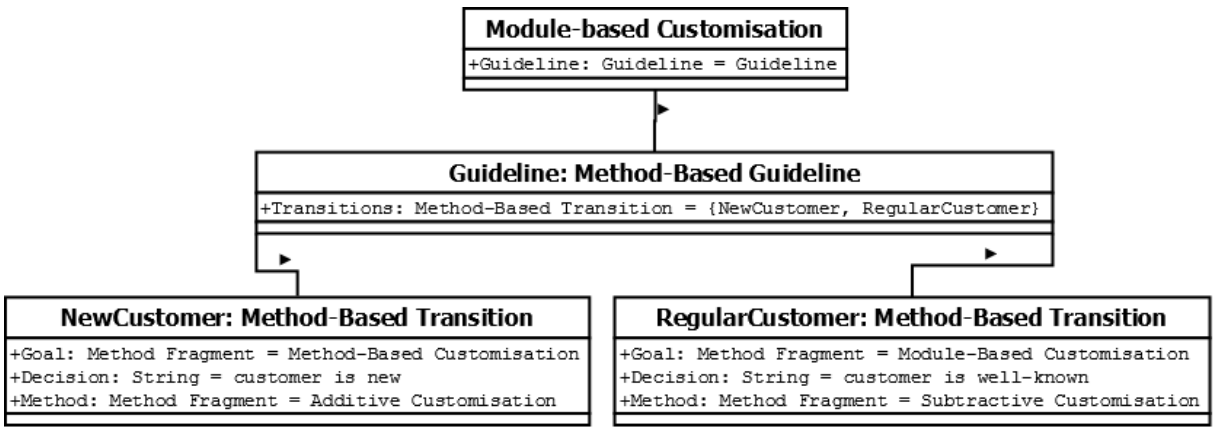

Fig. 7: Representation of the method-based guideline conforming to the metamodel

Contrary to transitions for method-based guidelines, transitions for strategy-based guidelines are defined as a triple *(source, goal, strategy)*. As presented in Fig. 8, the strategy-based guideline of the method fragment *module-based customisation* begins at the source *Start*. The goal *customise customer interaction* can be achieved using two different strategies. First, the *new customer strategy* used subtractive customisation. Second, the *regular customer strategy* uses additive customisation.

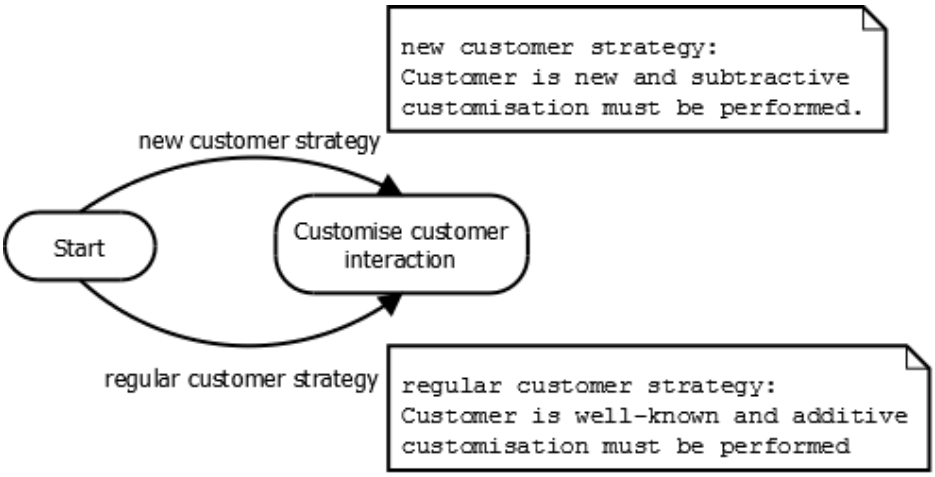

Fig. 8: Strategy-based guideline for module-based customisation

From the metamodel-conform representation of the strategy-based guideline (see Fig. 9), it can be seen that each of the depicted strategies is linked to a specific method fragment. The two strategies *new customer strategy* and *regular customer strategy* are defined as *T1* and *T2* respectively.

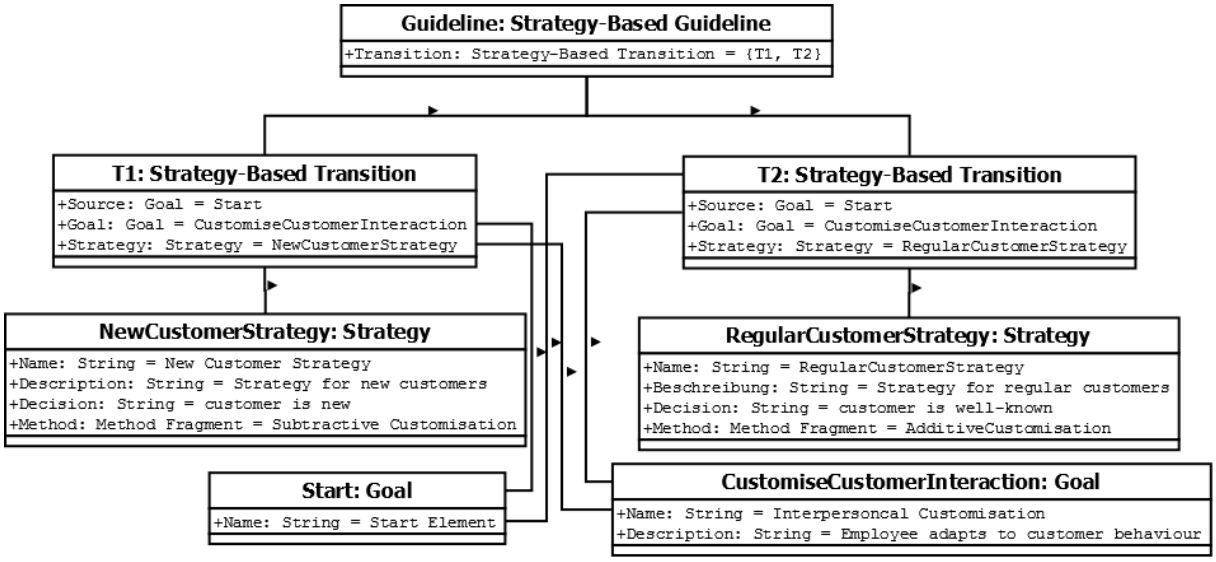

Fig. 9: Representation of the strategy-based guideline conforming to the metamodel

5. Usage of Service Engineering Method Fragments

Based on the metamodel presented in the previous section, it is possible to specify method fragments. These method fragments are stored in a method base acting as a

central repository. To assemble a method using existing fragments, the following approach can be used (Harmsen, 1997):

- *Defining the situation*: As a first step, it is necessary to define the situation, i.e. the specific characteristics of a planned service project. In doing so, only method fragments relevant for the respective situation are presented. The more characterisation factors are used, the more specific the situation can be described. However, it is also possible to describe an incomplete situation to support searching for method fragments in early service lifecycle stages.

- *Selecting method fragments*: Based on the description of the situation, appropriate method fragments are selected. Since it is not always possible to describe a situation in detail, alternative fragments might be found. Kornyshova (2007) argues that users should be supported in selecting suitable alternatives. In addition to explicitly selected method fragments, logical dependencies must be considered, e.g. method fragments requiring other fragments.

- *Integrating method fragments*: Using the selected method fragments, a holistic method is established covering the whole service lifecycle. For doing so, it is necessary to assemble the fragments into a meaningful order. Therefore, rules (e.g. preconditions) can be used.

- *Performing the service project*: The method consisting of selected method fragments is used for monitoring the process. However, dynamic changes must be considered, e.g. changes in the service characteristics resulting in a need for different method fragments. Therefore, it is necessary to select alternative method fragments during project performance.

In the following, a concept for a service customisation information system is presented (see Fig. 10). It consists of a *method base component* containing the predefined method fragments. The *service project characterisation component* is used for defining a specific service project and selecting appropriate method fragments. Based on the selected method fragments, a holistic method is assembled and transferred into a *workflow management component*. Using this component, a service project can be conducted.

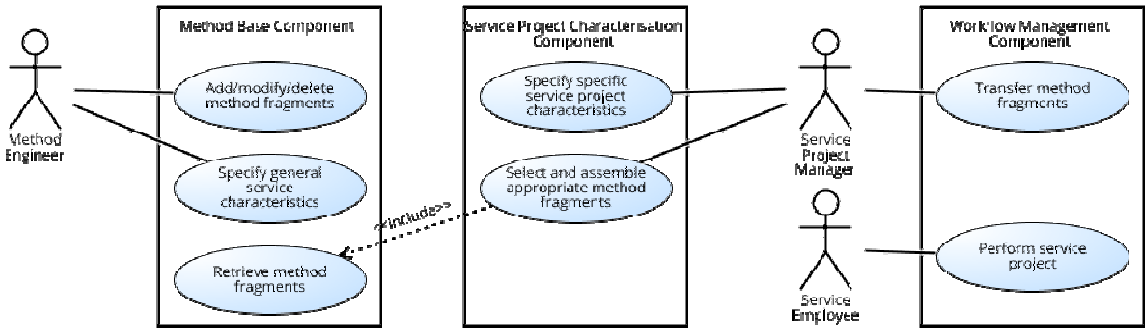

Fig. 10: Service customisation information system architecture

In general, three distinct roles using the service customisation information system can be identified. The *method engineer* is responsible for maintaining the method base. She has to add, modify, and delete method fragments and specify the general

service characteristics that are used to define a situation. The method engineer must ensure consistency of the method base. This is of special importance when new method fragments are added. Ideally, the method base component supports the method engineer by pointing out possible inconsistencies, e.g. contradictory method fragments.

The *service project manager* is responsible for characterising a specific service project. Thus, he selects the values of matching service characteristics. Based on the definition of a service project, appropriate method fragments are presented. The task of the service project manager is to choose between alternative method fragments and to assemble the fragments into a holistic method. Assembling fragments is supported by the system based on definition of dependencies between method fragments, e.g. mutually exclusiveness or requirements. The assembled method is transferred into the workflow management component. Finally, the service project is performed by the *service employee*. The workflow management component presents the method fragment as tasks that must be executed. In addition, the service employee is responsible for giving feedback about changed service characteristics. Based on this feedback, the assembled method is modified: inappropriate method fragments are removed and replaced by appropriate method fragments.

6. Conclusion and Future Research

Development and provision of customer-individual services is an ever-increasing challenge for companies, since it is positioned in the dichotomy between the need for efficiency and adapting services to customer requirements. The objective of our research has been the development of strategies for increasing the effectivity and efficiency of service customisation. Therefore, we specified a unified framework consisting of a semi-formally defined method fragment metamodel. In addition, examples for service engineering method fragments were presented. The existing fragments can be used to assemble process models for a service project with specific characteristics. The underlying metamodel allows for specifying new method fragments that adhere to the given structure.

The presented results can be used both in science and in industry. In particular, the metamodel should be part of future discussions about relevant service customisation elements. By using a metamodel-based approach, it is possible to reuse the metamodel and add missing elements or adapt existing elements. In addition, this should foster a general discussion about unifying or standardising service customisation approaches. As was revealed by the literature reviews, the service customisation domain is still very heterogeneous and is in need for a harmonisation of its terminology. Using the results in industry allows for establishing a method base including approved best practices.

Building on the results presented in this paper, our next steps are to continuously extend the method base by specifying new method fragments based on literature[1]. Additionally, existing method fragments are formalised according to the metamodel

[1] A thorough overview about method fragments is presented under http://serviceconfiguration.org

and transferred into a service customisation information system as defined in section 5. Based on this information system, service projects can be set up by defining their service characteristics. Based on these characteristics, relevant method fragments are selected and can be assembled into a holistic process model. In addition, the information system integrates software tools supporting method fragment application, e.g. software for additive service configuration.

7. References

Becker, M., Böttcher, M. & Klingner, S., 2011. *Systemising Service Classifications.* Hamburg, Germany, Fraunhofer.

Becker, M. & Klingner, S., 2016. *Konzepte zur kundenspezifischen Anpassung von Dienstleistungen.* Karlsruhe, to appear.

Brinkkemper, S., 1996. Method engineering: engineering of information systems development methods and tools. *Information and Software Technology ,* 38(4), pp. 275-280.

Brooks, F. P. J., 1987. No Silver Bullet Essence and Accidents of Software Engineering. *Computer,* April, 20(4), pp. 10-19.

Deneckere, R., Iacovelli, A., Kornyshova, E. & Souveyet, C., 2008. *From Method Fragments to Method Services.* Montpellier, EMMSAD .

DIN, 1998. *Service Engineering – Entwicklungsbegleitende Normung (EBN) für Dienstleistungen..* Berlin: Beuth.

Fähnrich, K.-P. & Meiren, T., 2007. Service Engineering: State of the Art and Future Trends. In: D. Spath & K. Fähnrich, eds. *Advances in Services Innovations.* Berlin, Germany: Springer Berlin Heidelberg, pp. 3-16.

Fähnrich, K.-P. & Opitz, M., 2006. Service Engineering: Entwicklung und Gestaltung innovativer Dienstleistungen. In: H. a. S. A. Bullinger, ed. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 85-112.

Harmsen, A. F., 1997. *Situational Method Engineering,* Utrecht: Moret Ernst & Young Management Consultants.

Heiskala, M., Paloheimo, K.-S. & Tiihonen, J., 2005. *Mass Customization of Services: Benefits and Challenges of Configurable Services.* Tampere, Finland, Tampere University of Technology, pp. 206-221.

Hevner, A. R., March, S. T., Park, J. & Ram, S., 2004. Design Science in Information Systems Research. *MIS Q.,* 28(1), pp. 75-105.

Jacka, J. M. & Keller, P. J., 2012. RACI Matrices. In: J. M. Jacka & P. J. Keller, eds. *Business Process Mapping.* Hoboken, NJ: John Wiley & Sons, Inc., pp. 255-275.

Karlsson, F. & Wistrand, K., 2006. Combining Method Engineering with Activity Theory: Theoretical Grounding of the Method Component Concept. *Eur. J. Inf. Syst.,* 15(1), pp. 82-90.

Kornyshova, E. a. D. R. a. S. C., 2007. Situational Method Engineering: Fundamentals and Experiences: Proceedings of the IFIP WG 8.1 Working Conference, 12--14 September 2007, Geneva, Switzerland. In: J. a. B. S. a. H. B. Ralytè, ed. Boston, MA: Springer US, pp. 64-78.

Leimeister, J. M., 2012. Dienstleistungsengineering und -management. In: Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 91-124.

Meyer, A., Blümelhuber, C. & Pfeiffer, M., 2000. Dienstleistungsqualität: Konzepte - Methoden - Erfahrungen. In: M. Bruhn & B. Stauss, eds. Wiesbaden: Gabler Verlag, pp. 49-70.

Münkhoff, E., 2013. Umsatz- und Profitabilitätsauswirkungen industrieller Dienstleistungen: Eine latente Wachstumskurvenanalyse. In: Wiesbaden: Springer Fachmedien Wiesbaden, pp. 11-63.

Pfeffers, K., Tuunanen, T., Rothenberger, M. A. & Chatterjee, S., 2007. A Design Science Research Methodology for Information Systems Research.. *Journal of Management Information Systems,* 24(3), pp. 45-77.

Porter, M. E., 2010. *Wettbewerbsvorteile: Spitzenleistungen erreichen und behaupten.* Frankfurt/Main: Campus.

Tolvanen, J.-P., 1998. *Incremental Method Engineering with Modeling Tools: Theoretical Principles and Empirical Evidence,* Jyväskylä: University of Jyväskylä.

Uhrmann-Nowak, R., 2010. Service goes Europe: Lösungen im mittelständischen Maschinenbau. In: *Service Engineering internationaler Dienstleistungen.* s.l.:Fraunhofer Verlag, pp. 209-223.

Zhou, Q. & Tan, K. C., 2008. *A bibliographic analysis of the literature on new service development.* Bangkok, IEEE, pp. 872-877.

8. Author address

Author(s):

Michael Becker
InfAI e.V.
Service Modelling and Engineering
Hainstr. 11, 04109/Leipzig
becker@infai.org

Stephan Klingner
InfAI e.V.

Service Modelling and Engineering
Hainstr. 11, 04109/Leipzig
klingner@infai.org