Efficient Business Service Consumption by Customization with Variability Modelling

Michael Stollberg and Marcel Muth
SAP Research, Chemnitzer Str. 48, 01187 Dresden, Germany
(michael.stollberg,marcel.muth)@sap.com

Abstract: The establishment of service orientation in industry determines the need for efficient engineering technologies that properly support the whole life cycle of service provision and consumption. A central challenge is adequate support for the efficient employment of complex services in their individual application context. This becomes particularly important for large-scale enterprise technologies where generic services are designed for reuse in several business scenarios. In this article we complement our work regarding Service Variability Modelling presented in a previous publication. There we presented an approach for the customization of services for individual application contexts by creating simplified variants, based on model-driven variability management. That work presents our revised service variability metamodel, new features of the variability tools and an applicability study, which reveals that substantial improvements on the efficiency of standard business service consumption under both usability and economic aspects can be achieved.

Key words: Business Services, Consumption, Efficiency, Customization, Model-driven, Variability Modelling

1. Introduction

In the last years, service orientation has become the dominating design principle for modern ICT technologies in industry as well as in the public sector. The aim is to exploit the enormous potential of services for enhancing the interoperability among systems and enabling the reuse of existing implementations. In consequence, a steadily growing number of available business services and service-based business applications can be observed. The design, development, usage, and management of such solutions requires sophisticated engineering technologies that support the life cycles of service provision and service consumption in an efficient and integrated manner.

This is subject to the emerging discipline of service engineering. Numerous efforts in academia as well as in industry have developed a wealth of techniques, methodologies, and tool support for this. However, most existing solutions focus mainly on support for service provisioning, i.e. for the design, development, publication, and management of services by the provider. The consumption side – i.e. the support for service consumers for finding suitable services and integrating them into the specific target application – is often neglected. Existing technology support for this is mostly limited to low-level technical details, leaving the major part of the analysis and integration task for actually consuming services to manual inspection.

The limitations become obvious when considering the usability of complex services that can commonly be found in real-world business technology environments. For example, consider the Enterprise Services which form the basis of SAP’s modern service-based enterprise technology. These are designed in a generic manner, deal with the complex data structures of the related business standards, and support several usage options. With this, the services become reusable in various business scenarios. On the other hand, their interfaces and usage conditions are considerably complex. Typically, a customer merely requires a subset or a specific flavour of the provided features for a particular business solution. Hence, the Enterprise Services need to be configured and integrated in order to properly fit the customer’s needs. This is a non-trivial task that requires both technical knowledge and business expertise. Due to the limited tool support, the customization requires massive human involvement and thus becomes a highly cost-intensive and error-prone task.

To overcome this, we present an approach for service customization by the creation of simplified variants which only expose the minimal set of features that is relevant for an individual usage scenario. These are defined on the basis of variability specifications that explicitly describe the usage conditions and constraints of the service. Exploiting model-driven engineering, we define a metamodel for describing the variable aspects of services, specify an engineering process for service customization, and provide tool support for both service variability modelling and the creation of customized variants with partial automation and intuitive user interfaces. The customization tools
ensure that the personalized variants comply with the usage conditions of the original service, and generate a conventional service description for each variant so that it can be deployed into consumer applications analogously to other services. The presented solution focuses on the functional level, i.e. describing and managing variability on the level of operations, message types and properties, and the dependencies among them. This appears to be the primarily relevant aspect to enhance the consumption support for existing business services which mostly are provided in form of WSDL interfaces along with natural language usage instructions.

In this article we complement our work in [1], where we introduced a process for Service Customizing by explicit Variability Modelling. In that work we present our revised variability metamodel, new features of the tools for service variability specification modelling and resolution, and we complement our work with an applicability study.

The structure of the article is as follows. At first, Section 2 provides a concise overview of the approach and defines the engineering process for service customization. Then, Section 3 specifies the metamodel for service variability modelling, and Section 4 defines the tool-supported procedures for service customization. Section 5 illustrates the technique for customizing a SAP Enterprise Service, and Section 6 evaluates the business relevance. Section 7 positions our approach within related work, and finally Section 8 concludes the article and outlines future work.

2. Overview

This section provides an overview of the approach for service customization. We define the central artefacts and roles involved in the process of providing and consuming customized services, outline the technical solution that is presented in detail in this paper, and motivate the need for such technologies.

Fig. 1 provides a comprehensive overview of the engineering process for service customization, identifying the involved roles, phases, and relevant artefacts for providing, preparing, and consuming customized services. We distinguish three main roles: the Service Provider develops and publishes services, the Domain Expert prepares them for customization by creating a variability specification model along with pre-configurations for respective user groups, and the Service Consumer customizes and personalizes the service in order to incorporate it into the specific application context. Note that these three roles represent abstractions from the several sub-roles that can be found in real-world service engineering processes.

Fig. 1: Service Customization – Roles, Phases, Artefacts

The following explains the service customization engineering process in detail. In the first phase, the Service Provider develops a service and publishes it in a repository. The service interface describes the functional aspects in terms of operations, messages, types, and endpoints. In the context of model-driven engineering, this is described by a Service Model on the basis of a respective metamodel, such as a WSDL metamodel (e.g. [8]), SoaML as the OMG standard for service modelling...
In the second phase, the Domain Expert prepares the service for customization. For this, he creates a **Variability Specification Model** that describes the variable aspects of the service. Considering the functional aspects, this covers the explicit declaration of mandatory and optional operations, messages, and message types as well as the dependencies among them; additional variability models and management techniques can be defined for business-relevant aspects such as quality-of-service indicators, service-level agreements, or usage and pricing models. The variability specification is defined in accordance with the **Service Variability Metamodel**, which provides the necessary constructs for modelling the variability of services. There might be multiple variability specification models for a service where each one is pre-configured for a particular application scenario (e.g. for specific industry sectors or geographical usage contexts).

In the third phase, the Service Consumer adapts the service to the individual consumption context by creating a personalized variant. For this, he selects the desired features can define concrete values for static parameters (those that are not changed dynamically during the service invocation). This is described by a **Service Resolution Model** wherein the variable aspects as defined in the variability specification are resolved, so that it presents a valid instantiation of the variability specification that only contains the desired features while all usage conditions, constraints, and dependencies among the elements of the original service are satisfied. This is ensured by the tool-supported explicit variability modelling and the validation of the usage conditions throughout the customization process, so that the correct consumption of the service is guaranteed. Finally, a **Variant of the Service Model** is generated from the resolution model. This contains only the selected features while assuring the proper consumption of the service, therewith presenting a variant of the service that is adapted to the individual application scenario of the consumer. The variant is described in terms of the same metamodel as the original service, so that it can be consumed analogously to any other service. There can be several variants associated to each variability specification model, one for each consumer application scenario.

The technical solution for supporting service customization presented in this paper encompasses the specification of the service variability metamodel and tools for the creation of variability specification models by domain experts as well as for variability resolution modelling and service variant creation by service consumers. We here focus on the functional level (i.e. the variability modelling and management on the level of operations and the associated data types), as this appears to be the primarily relevant aspects for enhancing the consumption efficiency for existing business services which mostly are provided via WSDL interfaces. The overall idea for enabling service customization by variability modelling is adopted from works on variability management in Software Product Line Engineering (SPLE, e.g. [3]), which however deal with different elements and thus employ different models and techniques (see Section 7 for a more detailed discussion). In order to ensure the efficiency of the service engineering process, we consider a model-driven approach where the service and variability models are defined on the architecture level, i.e. on the PIM level in MDA terminology [12]. Our prototype implementation of the service customization tools supports SoaML and WSDL as service models; however, the service variability metamodel is defined orthogonally to the base model, so that it can also be applied to other service metamodels.

Before presenting the technical solution in detail, let us discuss the motivation and business relevance of such a service customization technology. As outlined above, the need arises from the growing complexity of services, which particularly occurs in the context of business applications. For example, consider a business service for creating and managing sales orders. A sales order is a relatively complex data object (consider the standard data structured as defined in RosettaNet or ebXML). Furthermore, there are differences in the details of the data structure in the standards for different industries. To be reusable in various applications within several industries, a general purpose business service needs to support all options and specific features that are relevant for the targeted usage industries. In consequence, its interface becomes very complex regarding the number operations, the size of input- and output objects, and the conditions and constraints for proper and consistent consumption. A specific consumer typically only uses a subset of the provided features for his individual sales order management. However, due to the complexity, the general purpose service is not easy to understand, and its configuration for the individual needs of the consumer is a time consuming task that usually requires external support.

Our approach enables a step-wise reduction of the complexity and improves the technology support for customization. At first, a domain expert can define variants that are pre-configured for specific user groups, e.g. one variant for the automotive industry, one for steel production, and another one for the
telecommunication sector. Furthermore, the detailed usage conditions for each variant are described explicitly in terms of a variability specification model. On this basis, a service consumer can then define personalized variants by selecting the desired features. The model-driven approach facilitates the abstraction from technical details as well as the provision of intuitive graphical user interfaces for modelling support, and the applied variability management techniques models ensure that the selections by the consumer are compliant with the usage conditions of the service. The generated technical service interface for the personalized consumer variant is naturally significantly less complex than the one of the original sales-order service while it adheres to its usage conditions, so that a correct and consistent consumption is ensured. In order to demonstrate and evaluate the business relevance of the service customization technique, we shall illustrate the customization of a business service in Section 5 and discuss the surplus value for the enhancing the consumption of business services in detail in Section 6.

3. Service Variability Metamodel

This section introduces the metamodel for service variability. The metamodel defines all necessary constructs for defining variability specification and resolution models, therewith providing the basis for the procedures and tool support for service customization that are presented the subsequent sections.

The following provides a comprehensible overview on the overall design of the service variability metamodel and the supported variability management mechanisms. For this, we explain the metamodel in a subsequent manner. We first introduce the top-level elements for variability specification and resolution models as well as the central elements for variability modelling and management, and then present the constructs for variability modelling on the level of operations and data types, therewith covering the functional aspects of services.

3.1 Main Elements

We distinguish two groups of main elements of the Service Variability Metamodel. The first group covers the main elements for variability specification and variability resolution models as well as the binding to the base model, i.e. the metamodel used for service descriptions, while the second group defines the constructs for the supported variability management mechanisms for service customization on the functional level.

![Fig. 2: Service Variability Metamodel – Main Elements](image-url)
We commence with the first group, which consists of the elements shown in Fig. 2. A *VariabilitySpecification* serves as container for the variability definitions of the variable artefacts of a service. The variable artefacts are described by instances of subclasses of *VariableElement*; concrete properties are defined for the subclasses on the operation and data level as explained below. A *VariabilityResolution* describes a personalized variant of the service. The resolution of the variable aspects as defined in the corresponding *VariabilitySpecification* is defined by instances of *ResolutionElement*. The binding to the original service description defined as a reference from the *VariabilitySpecification* to an *EObject*. This allows the binding to various service description models (e.g. SoaML, WSDL, USDL, etc.), so that the Service Variability Metamodel is orthogonal to the used base model.

The second group of main elements defines the constructs for the modelling and management of the variability on the functional aspects of services. For this, the technique presented here supports the following 4 variability management mechanisms, which are realized by the metamodel elements shown in Fig. 3:

1. **The declaration of mandatory and optional elements**: this is modeled by the Boolean property *required* of the *VariableElement* class, which itself serves as the superclass for variability modelling on different levels of service descriptions (see below)
2. **The definition of dependencies among elements** modelled by the *Constraint* class with 3 subclasses: *ExcludesConstraint* for defining the mutual exclusion of two or more elements, *RequiresConstraint* for defining the mandatory inclusion of related elements, and *GenericConstraint* for defining all other constraints, extensible with formal constraint definitions
3. **The selection of desired features for a specific application context** within a variability resolution model, modeled by the Boolean property *selected* of *ResolutionElement*, and
4. **The definition of fixed values for static properties** that are not changed within the application scenario or *default values* that are used for invocation when no concrete value is provided; fixed and default values can only be defined for the classes *VariableProperty* and *PropertyResolutionElement*.

![Fig. 3: Service Variability Metamodel – Variability Mechanisms](image)

### 3.2 Operation Level

The main elements introduced above are used and refined for modelling the variability for functional aspects on the relevant levels. The first functional level supported by the metamodel is concerned with the operations that define how a service can be consumed by the exchange of messages.

Fig. 4 shows the metamodel elements for modelling service variability on the operation level. Here, the main element is *VariableOperation* that is defined as a subclass of *VariableElement* and thus inherits the concepts for the variability management mechanisms. This enables the definition of mandatory and optional operations, the dependencies and constraints among them, and the selection of the desired ones for a consumer variant. Each operation defined the base model (i.e. original service description) is bound to a *SimpleVariableOperation*. As an additional variability management mechanism, the class *ComplexVariableOperation* enables the grouping of related operations. Here, the Boolean property *multiple* allows modelling exclusiveness (the value *true* means that only one of the related operations can be used).
3.3 Data Level

The second level for variability modelling on functional aspects of services is concerned with the message types, i.e., the data structures used within the messages. Fig. 5 shows the metamodel elements for this. The structure is analogous to the operation level. Here, the actual variability is modelling on the properties of the types in order to ensure a high flexibility and the reusability of variability specification models. Hence, the main element is VariableProperty that inherits the constructs for the supported variability management mechanisms as introduced above. A VariableType is bound to a data type from the base model as a reference to an EObject in order to allow the binding to various service description models as explained above.

4. Service Customization Procedures and Tool Support

This section explains the techniques for service customization that work on the metamodel presented above. Refining the overall engineering process introduced in Section 2, we here focus on the methodological aspects while presenting the tooling support in the context of the illustrative example below in Section 5.

4.1 Service Variability Specification

As outlined above, the first activity for enabling the customization of services is the creation of a variability specification model. This is performed by a domain expert in the second phase of the overall engineering process (cf. Fig. 1 in Section 2). The procedure for creating a service variability specification consists of two tool-supported steps as shown in Fig. 6: at first the Extractor creates a skeleton of the variability specification by extracting implicit variability information from the service base model, and then the domain expert refines this by adding explicit variability information using the variability specification editor.
In the first step, the Extractor automatically generates a basic skeleton of the variability specification. For this, it creates the variability specification model and the corresponding variable elements for each operation, data type, and property that is defined in the original service as well as the bindings to the base model. In addition, basic variability information can be extracted, such as the declaration of mandatory elements or unary dependencies between operations and data types. A prerequisite for this is that the service provider has obeyed to an adequate SOA governance procedure, which ensures the correctness and completeness of the service descriptions and are commonly applied for the provision of business services [5]. The generation of the service variability specification skeleton is completely automated, therewith ensuring the completeness of the variability specification model and significantly reducing the manual modelling effort.

In the second step, the domain expert can refine the service variability specification skeleton by adding explicit variability information, i.e., variable aspects that cannot be automatically extracted from the base model of the given service. Typical modelling tasks for the manual refinement are the grouping of related operations using the `ComplexVariableOperation` construct, the definition of mutual in- and exclusiveness among operations as well as between operations and their mandatory or optional input and output parameters by defining `Constraints` and using its specialized sub-classes, and the definition of dependencies and usage constraints among the distinct variable elements (e.g., that usage of operation B requires the inclusion of operation A, or that a certain type of a property restricts the possible values for another property). The manual refinement is not necessary for creating complete and semantically correct variability specifications, but it is highly recommended in order to obtain mature variability specifications as the basis for exploiting the full potential of the service customization technique. As presented in the next section, the variability specification editor supports the domain expert in the variability modelling in context sensitive manner, including basic validation mechanisms. Note that the definition of explicit variability information can in general not be automated because supplementary knowledge to the original service description is added. However, dependent on the structure of the service descriptions and the applied SOA governance process, further partial automation support can be implemented within the Extractor.

The final aspect of the service variability specification modelling is that several variability specification models can be defined for a single service. Each of the service variability specifications then represents a pre-configuration of the original service for a certain application context. A straightforward example for this is to define separate variability specification of a payment service for different countries where each one defines the respective VAT model in terms of fixed or default values. Note that within real-world scenarios the sets of mandatory elements as well as the usage conditions can differ between the distinct variability specification models for a single service, as these denote the usage conditions adapted for a specific application domain and not merely the conditions for the technically correct consumption of the service.

### 4.2 Service Variant Creation

The second main activity is creation of a personalized variant of a service for a particular application scenario, which is performed by the Service Consumer in the last phase of the overall service customization procedure introduced above in Section 2. The overall aim is to enable the easy and
rapid creation of personalized variants, so that the Service Consumer can quickly create the desired variant without having to deal with technical details or inspecting the usage conditions in detail.

Fig. 7 below shows the detailed procedure for this. At first, the Service Consumer chooses a suitable variability specification model that has been previously prepared by a domain expert for the service that is to be used. Then, a personal variant is modelled by selecting the desired features and setting pre-defined values. This is supported by the variability resolution editor, a graphical tool that allows for quickly understanding the service and provides context-sensitive support for resolution modelling along with built-in runtime validation mechanisms for the constraints defined in the chosen variability specification model. The tool ensures that the user selections and inputs comply with the conditions defined in the variability specification model, therewith ensuring that the resulting variability resolution model is a valid instantiation of the chosen service variability specification.

Finally, a service description of the created variant is generated automatically. For this, the Generator reproduces the descriptions from the original service description only for those elements that have been selected in the variability resolution model along with the defined fixed and default values, and defines a new namespace for the variant. The service variant is described in terms of the base service model, so that it can be consumed analogously to any other service. The explicit variability modelling and the validation of the usage conditions throughout the customization process ensure that the generated service variant is valid for properly and correctly invoking the original service. With this, the generated service variant description provides the minimal interface description for the original service that is customized for the individual application scenario.

5. Illustrative Example

This section exemplifies the customization of an SAP Enterprise Service with the service variability technique in order to illustrate the techniques and tool-supported procedures introduced above. The following first introduces the example scenario, and then demonstrates the modelling of a service variability specification model as well as the creation of a personalized service variant.

The prototypical tooling support for both service variability specification modelling and the creation of personalized service variants is implemented as a set of plug-ins for the Eclipse Modelling Framework (EMF, see www.eclipse.org/modelling/emf). The tools implement editing support with the runtime validation mechanisms and the automated extraction and generation facilities as explained above. The prototype supports bindings to various base service description models, namely WSDL [8], SoaML [17], and USDL [22]. In the following, we demonstrate the service variability tools for SoaML that are provided as integrated components of the open-source tool suite for the model-driven development of service-based system landscapes that has been developed in the European research project SHAPE [21] (see www.shape-project.eu). In addition, the tools are available as a stand-alone EMF application and as an integrated component of the comprehensive Service Delivery Framework developed by SAP Research in the context to the Internet of Services activity (see www.internet-of-services.com).
5.1 Customizing the Goods Movement SAP Enterprise Service

As the example scenario, we illustrate the customization of the Enterprise Service “Goods Movement”. This is a real-world business service available in the SAP Developer Network (see www.sdn.sap.com), providing basic business functionalities for managing the movement of goods and being used in business applications for transport & logistics management and warehouse management. It offers operations for creating, deleting, and updating goods movement objects (i.e. the information object describing that a good – e.g. product or a sales order item – moved from place A to place B), and works with complex data structures which represent the relevant standards for describing goods and related business objects within the supported industries.

The aim of the illustrative example is to demonstrate the concepts and tool support for customizing services with the service variability technique. For this, we here consider a simplified version of the goods movement Enterprise Service; we shall discuss the complexity of actual SAP Enterprise Services below in Section 6 in more detail. The service offers four operations: two for creating goods movement objects (one with references to related documents like e.g. sales orders, and one without), and operations for deleting and updating goods movement objects. The relevant message types are the input and output parameters of the messages, which refer to the relevant standard business objects for goods movement and related business objects like item, tax, transportation, and material.

For illustration, we will create a simplified variant that only contains 2 operations (creation of goods movements with reference to a sales order, and the deletion operation), and the minimal set of types that are needed for consuming the service via the variant. Fig. 8 shows the SoaML descriptions of the original demonstration service and the variant whose stepwise creation we explain in the following.

![Fig. 8: Example Scenario – Customization of Goods Movement SAP Enterprise Service](image)

5.2 Service Variability Specification Modelling

The first step in creating the personalized variant of the goods movement enterprise service is the creation of the service variability specification model. As explained above in Section 4.1, this is supported by the tools as a two-step activity: at first the built-in extractor facility automatically generates a skeleton of the variability specification, which then is refined by the domain expert with additional variability information.

Fig. 9 shows this for our example. Part (a) on the left-hand side shows the automatically generated skeleton. This defines the variable elements for each operation, data type, and property along with the bindings to the base model. The figures shows the SimpleVariableOperation elements that correspond to the four operations explained above; analogously, the ‘Data Level’ tap contains the variable elements for all types defined in the original service description along with their properties. This ensures the completeness of the service variability model, so that the domain expert can focus on defining the explicit usage conditions. Part (b) illustrates the tool support for two of the most common modelling activities. The first one is concerned with grouping the two operations for creating goods movement objects into a ComplexVariableOperation, indicating that they provide similar functionalities. Here, the two operations createGoodsMovement and createGoodsMovementWithReference are non-exclusive, i.e. they can both be used in a consumer application; this is explicitly modeled by setting
the value of the *multiple*-property of the complex operation to false. The Variability Specification Editor provides context-sensitive editing facilities for this, ensuring that the user can only perform structurally valid modelling activities.

![Fig. 9: Tool-supported Service Variability Specification Modelling](image)

The second example for explicit variability modelling is concerned with the dependencies among the input and output parameters of operations. The operation `createGoodsMovementWithReference` enables the creation of a goods movement object along with a reference to a related business object, e.g. a sales order. For this, the ID of the referenced business document is a mandatory input, which is defined by `baseDocumentID`, a property of the data type `Item`. This however is an optional input field within the interface description of the original goods movement service; it is only mandatory as an input parameter for the `createGoodsMovementWithReference` operation. For modelling this precisely, the domain expert defines a `RequiresConstraint` for the operation with the `baseDocumentID` property as the related variable element, shown in the lower right-hand side of Fig. 9.

### 5.3 Service Variability Resolution and Variant Creation

We now turn towards the second activity of creating the actual service variant. As explained above in Section 4.2, this is done by the Service Consumer with tool support for the rapid and easy variant creation. At first, the desired features are defined in terms of a variability resolution model that represents a valid instantiation of the chosen variability specification, and from this a service description for the variant is generated which contains the minimal set of elements that is necessary for correctly consuming the service within the individual application context.

![Fig. 10 shows the variant modelling with the Variability Resolution Editor for our example. At first, the consumer selects the desired operations, here `deleteGoodsMovement` and `createGoodsMovementWithReference`. For the latter, the built-in runtime validation mechanism indicates a constraint violation. As discussed above, the property `baseDocumentID` is a mandatory input parameter for the operation and thus must be included in the variability resolution model (see step 1 in the left-hand side of the figure). To resolve this, the consumer switches to the 'Data Level' tab and includes the `baseDocumentID` property by activating the checkbox. He can further define a default value in order to always ensure a proper consumption of the service; in SAP systems, the value “0” denotes that an actual value is not given, so that the object can be treated accordingly. Finally, the consumer selects the optional properties `salesOrderId` and `salesOrderItemId` to be included in the variant, which allow defining the references of goods movement objects to the respective sales orders.

On the 'Data Level' tab, the types and properties declared as mandatory in the chosen variability specification are activated by default and cannot be deselected by the user. Another built-in facility for better supporting the variant modelling by a Service Consumer is a dynamic filtering feature for the relevant data types and properties: only those types and properties are displayed which are used as input or output parameters of the selected operations. This allows the Service Consumer to focus on the elements relevant for the individual application scenario while masking the irrelevant ones, which...
becomes in particular relevant for creating personalized variants of complex services where the majority of the encompassed elements are irrelevant for a particular consumption context.

The final step is the automated generation of the service variant description. When the variability resolution model is valid – i.e. all usage conditions as defined in the corresponding variability specification model are satisfied – the built-in generator defines a new namespace and reproduces the service element description for only those elements selected by the consumer. Fig. 11 shows this for our example: the file `My_MgnGoodsMovement.uml` generated from the variability resolution model is the SoaML description of the personalized variant as defined in the beginning (see Fig. 8). This presents a conventional service description, which allows the service variant to be used in the same way as any other service. Fig. 11 also depicts the default resource management structure of the service variability tool, which can be adapted to the individual service management infrastructure.

6. Evaluation

After having explained the technical realization, this section discusses the business relevance and expectable benefits of the service customization technique by variability modelling. For this, the following examines the qualitative improvements for the consumption of SAP Enterprise Services as well as for other real-world business services, and discusses the relevance of the presented technique for future service consumption frameworks.

The modern SAP business solutions are provided in form of a service-oriented architecture, whose basic building blocks are the ca. 4000 Enterprise Services that provide access to the principal business functionality like sales order or customer management. Our analysis of the existing SAP Enterprise Services has revealed two central aspects that hamper their rapid and easy consumption for the development of customer solutions, depicted in Fig. 12 below. The first one is the complexity of the service interfaces: part (A) of the figure shows the actual signature of the operation...
createGoodsMovementWithReference of the goods movement Enterprise Service that we have discussed above in our illustrative example. The input message consists of 104 data fields of which merely 12 are mandatory for the correct invocation of the service; the other 92 data types and properties support various usage options for different industries, working with respective standard data elements. This complexity makes it very difficult for consumers to understand how to properly use the service within an individual application context. However, the design of the service is appropriate with respect to its business purpose: the service shall provide a general-purpose facility for managing goods movement so that it can be re-used in various application scenarios, and it must support the standard data structures of different industries in order to ensure the interoperability with other business systems. Similar observations are reported in [10]: business services in the oil & gas industry need to work on the relevant industry standards in order to ensure system interoperability, and modern standards like the Production Markup Language (PRODML) are complex in order to cover all relevant aspects in sufficient detail. In consequence, the services for managing relevant business information expose a similar complexity as the SAP Enterprise Services, so that analogous hurdles arise for their consumption.

This means that complexity is an inherent characteristic of services in real-world business applications, but the resulting complexity hampers the usability by service consumers. This is intensified by the fact that currently the usage conditions for business services are merely provided in form of natural language descriptions, which is the second central observation of our analysis. Part (B) of Fig. 12 shows a screenshot of the usage guide provided in the ES Workplace, the central repository for an SAP Enterprise Services: the information on the mandatory input parameters as well as the supported usage options and constraints on the optional parameters are described in form of free-text. Other commercial business services are provided analogously, e.g. the e-commerce services by Amazon are provided in form of a WSDL interface along with exhaustive free-text usage documentation (see aws.amazon.com). With this, a service consumer needs to manually inspect the conditions to use the service in a specific application, which usually consumes several basic business services. In consequence, the development of customer applications on the basis of business services is a highly cost-intensive and error-prone task, which is often reported as one of the main challenges for establishing service-orientation in industry.

Fig. 12: Complexity and Consumption Support for SAP Enterprise Services

The service customization technique by variability modelling presented here addresses both identified challenges. For the former, the concept of pre-configured variants in form of several variability specifications which prepare a service for different application context and particularly the concept of personalized variants that merely contain the minimal set of features needed for an individual
application scenario allow a better handling of the complexity inherent to real-world business services. For the latter, the explicit modelling of the usage conditions in a machine-processible manner and the consumer tool-support for easily creating service variants with the built-in mechanisms for constraint validation and dynamic filtering for the relevant elements allow overcoming the deficiencies of the natural language usage guides. In summary, our technique allows service consumers to better understand available business services and can achieve a significant reduction of the human effort for utilizing real-world business services as the building block for customer applications.

Besides the possible improvements for the consumption of business services in existing service infrastructures, the presented customization technique appears to also be relevant for integrated service delivery platforms as envisioned for the future Internet of Services (e.g. [7]). These shall provide integrated service engineering tools for the discovery, composition, integration, and the platform-independent delivery of services on any device. For this, a customization technique that facilitates service consumption via personalized variants with minimized technical interfaces appears to be desirable in order to enable the usage and processing of complex services. Concrete examples for these are the delivery of business services on mobile applications for which light-weight service descriptions are required (e.g. [9]), or for the application development with graphical mashup-up tools wherein only services of a limited complexity can be handled (e.g. [11]).

7. Related Work

This section positions our approach within related work. As analyzed in [4], a central challenge that arises for industrial scale engineering technologies is the handling of services that deal with large and complex data structures of real-world business objects. This emphasizes the need for sophisticated customization techniques, whose efficient and scalable realization remains a grand challenge [18]. Our approach aims at facilitating the customization of services by explicit variability modelling on functional aspects, and we are not aware of any other approach that addresses this problem in a similar way.

Most existing tool support for the configuration of services is limited to low-level technical aspects, such as the deployment to an execution platform or the configuration of technical parameters for invocation and runtime (e.g. [2]). Although helpful for the service engineers and thus being an integral part of modern industrial service engineering environments, these techniques can only be applied after the service has been adapted to the specific usage scenario, leaving the major part of the customization task to manual inspection.

Only few works address service customization on higher levels of abstraction. Most existing solutions merely provide methodological guidelines with limited tooling support for users (e.g. [6],[20]). This appears to be inadequate for reducing the effort and error-proneness resulting from manual customization. [23] presents an approach for user-driven service customization on the basis of usage policies; however, this works with mostly informal policy descriptions that do not allow an automated validation of human customization decisions. More advanced techniques achieve significant improvements by working on formal constraint and user request descriptions (e.g. [15]). This however requires significantly more effort for creating the additional resource descriptions than our approach.

Our approach adopts concepts of variability modelling developed in the field of model-driven software product-line engineering (SPLE, [3]). We adopt the overall approach of explicitly describing the variable aspects and resolving them by selection and parameter instantiation for a specific usage scenario, as well as the principle of orthogonal design by separating the variability specification from the original service description [19]. However, there are substantial differences: at first, SPLE is concerned with describing the variability of products, while for services we need to consider other aspects like operations, messages, and messages types as supported by our metamodel. Secondly, SPLE variability management considers a closed world where all potential variations are known a priori, and variants are defined by substitution mechanisms that insert the predefined variations into the product description [13]. This appears to conflict with the nature of service-orientation where we need to consider an open world with unforeseen usage scenarios. A customization technique for complete SaaS applications that employs variability modelling is presented in [16]: the provider defines an application template where the configurable aspects are described by variation points; a consumer binds these to specific services or alternative processes in order to obtain a customized solution. Although the overall approach is similar, our metamodel is significantly more expressive for describing the variability of single services. Thus, the works can be considered to be complementary: the variability modelling and management techniques presented here can be used to enhance customization techniques for service-based applications.
The customization technique supported by our approach is the simplification of complex services. Another possibility is to define extensions to the original service, e.g. by adding supplementing features as suggested in [14]. This appears to be suitable for customizing services that do not encompass support for several application scenarios but merely provide core features that can be extended with specific functionalities for particular application scenarios, and thus can be considered to be complementary to our technique.

8. Conclusions

This article has presented an approach for service customization by the tool-supported creation of personalized variants that merely contain the minimal set of features needed for an individual application scenario, enabled by the explicit modelling of variable aspects and usage conditions of services on functional aspects.

The need for such techniques arises from the growing number of complex services that are designed for reuse in various application contexts and deal with extensive data objects, which can be particularly found in service-oriented business applications. Such real-world business services become too complex to be easily understandable by human users, and existing service engineering tools appear to not be suitable for supporting their employment in concrete consumer contexts in an adequate and cost-efficient manner. In order to overcome this, we propose a three-phased engineering process: domain experts prepare the services that have been published by a provider for specific usage contexts by defining variability specification models that explicitly describe the variable aspects; upon this, service consumers can easily create personalized variants that adopt the services to the specific context of the individual application scenarios.

In order to support model-driven development, we have defined a metamodel for describing the variable aspects of services on the functional level, i.e. considering the operations and data structures used in the input and output messages. The presented solution supports the necessary variability management mechanisms for the explicit declaration of mandatory and optional elements as well as dependencies among them, and for the selection of desired features as well as the definition of default and fixed values for a particular application scenario. We have developed tools for supporting domain experts in the specification of the service variability as well as service consumers in the creation of individualized variants with minimal modelling effort and abstracting from technical details, and we have demonstrated them for customizing an SAP Enterprise Service. We further have explained how the presented technique allows overcoming the main deficits of existing technology support for the consumption of real-world business services, namely handling of the inherent complexity and the deficiencies of free-text usage guides which require a cost-intensive manual inspection by the consumer.

The presented technique enables the consumption of complex services in individual application scenarios, with the main benefit of enabling the rapid and easy creation of personalized variants by service consumers while ensuring the correct consumption by the explicit variability modelling and the thorough validation of usage constraints throughout the customization process. We however consider the presented technical solution as a starting point for future research. For this, we particularly plan to incorporate additional aspects and techniques for managing the variability of services (e.g. support for business-relevant aspects such as quality-of-service and other non-functional aspects, mechanisms for handling conditions on instance data level, and support for service extensions mentioned above). In a longer perspective, the aim is to develop customization techniques for combined service bundles and applications, for which the presented variability modelling and management techniques for single services shall serve as a basis.

Acknowledgements: this article is mainly based on works supported by European Commission under the SHAPE project (FP7 - 216408).

References


