Value-Based Service Modeling and Design: Toward a Unified View of Services

Hans Weigand[1], Paul Johannesson[2], Birger Andersson[2], and Maria Bergholtz[2]

[1] Tilburg University, P.O. Box 90153,
5000 LE Tilburg, The Netherlands
H.Weigand@uvt.nl
[2] Royal Institute of Technology
Department of Computer and Systems Sciences, Sweden
{pajo,ba,maria}@dsv.su.se

Abstract. Service-oriented architectures are the upcoming business standard for realizing enterprise information systems, thus creating a need for analysis and design methods that are truly service-oriented. Most research on this topic so far takes a software engineering perspective. For a proper alignment between business and IT, a service perspective at the business level is needed as well. In this paper, a unified view of services is introduced by means of a service ontology, service classification and service layer architecture. On the basis of these service models, a service design method is proposed and applied to a case from the literature. The design method capitalizes on existing value modeling approaches.

Keywords: service design, business services, value modeling.

1 Introduction

Service-Oriented Architectures provide major advantages for today's enterprise information systems by presenting the interfaces that loosely coupled connections require [Pa05]. Web services [WS04] seem to become the preferred implementation technology for realizing the SOA promise of service sharing and interoperability.

In the view of Papazoglou and Van den Heuvel [PH06], (web) service design and development is about identifying the right services, organizing them in a manageable hierarchy of composite services and choreographing them together for supporting a business process. A *business service* (in this context, a service implementing a business process) can be composed of finer-grained services, which in turn are being supported by infrastructure services. Following previous work on SOAD [Zi04], they distinguish between top-down, bottom-up and meet-in-the-middle approaches and discuss major principles of service design such as low coupling and high cohesion. Although the paper provides useful criteria, it considers service design mainly as a software engineering problem, which in our view is not sufficient. As stated in [NL07], "the current trend toward a service-oriented enterprise necessitates a formal characterization of business architecture that reflects service-oriented business thinking". The starting point for design should be the business level at which services

P. van Eck, J. Gordijn, and R. Wieringa (Eds.): CAiSE 2009, LNCS 5565, pp. 410–424, 2009.

can be identified that provide value to customers and can be offered in an economically viable way.

IBM's SOAD has evolved into SOMA [AG08] described as "a software development life-cycle method invented and initially developed in IBM for designing and building SOA-based solutions". SOMA advocates a meet-in-the-middle approach. Domain decomposition is a top-down analysis that starts with analysis of the functional areas in the business domain and of the business processes. This is complemented by a bottom-up asset analysis. The two lines are pulled together by Goal-Service Modeling (GSM). SOMA incorporates many more methods and techniques, including conceptual data modeling, and advocates a fractal model for software development. On the basis of the information available, we infer that SOMA is a comprehensive method based on real industrial experience but the same remark applies as above: it does not consider service-oriented business thinking.

In contrast, [Sp08] introduces the notion of service system in an abstract way that can be applied to the business. The service system is defined as an open system capable of improving the state of another system through sharing or applying its resources (providing value), and capable of improving its own state by acquiring external resources (receiving value). The way of thinking is resonating the earlier work of Norman as well as Prahalad on co-creation of value [NR93, PK08], and existing work in the IS field on value modeling [Go00, Mc82].

What is lacking so far is a principled way of linking the two different branches of service science [Al08]. If a business is viewed as a service system, what does this have to do with service-oriented software design?

The objective of this paper is precisely to establish this link by (a) proposing a generic service model in which software services are defined as a service subtype, and (b) providing a method for service identification that starts from a value-based business model. Section 2 sets the stage by reviewing the most relevant business modeling approaches. It also introduces a running example, taken from [AG08] to allow easy comparison. In section 3, we provide a general service model rooted in the REA ontology, and extend it with a fully integrated service layer architecture. Section 4 introduces a value-based service identification method and illustrates it using the running example. Section 5 contains conclusions and directions for future research.

2 Business Modeling and Service Systems

There exist a number of approaches, languages, and ontologies for business modeling in literature, e.g., [Go00], [Di05], [Us96] and [Mc82]. In [An06] the e3value [Go00] and the REA ontologies [Mc82] were compared (together with a third business ontology – the BMO [Os04]) in order to establish a common reference business ontology. One result of that comparison was a set of mappings between e3value and REA indicating strong similarities between the concepts of the two.

2.1 Business Modeling

The Resource-Event-Agent (REA) ontology was formulated originally in [Mc82] and has been developed further, e.g. in [UM03, Ge06, Hr06]. Its conceptual origins can be traced back to traditional business accounting. REA was originally intended as a basis for accounting information systems and focused on representing increases and decreases of value in an organization. REA has been extended to form a foundation for enterprise information systems architectures [Hr06], and it has also been applied to e-commerce frameworks [UM03]. The following is a short overview of the core concepts of the REA ontology. In section 3 their relation to the service concept is proposed and motivated.

A *resource* was defined as "any object that is of utility and under the control of some enterprise". Originally, only resources that could be exchanged were considered, such as goods, services and money. Later on, internal resources were taken into account as well, including intangible ones like knowledge.

Resources are modified or exchanged in processes. A *conversion process* uses some input resources to produce new or modify existing resources. For example, water and flour can be used as input economic resources in a baking conversion process to produce the output economic resource bread. An *exchange process* occurs as two agents exchange (external) resources. To acquire a resource an agent has to give up some other resource. For example, in a goods purchase a buying agent has to give up money in order to receive some goods. The amount of money available to the agent is decreased, while the amount of goods is increased.

The constituents of processes are called *economic events*. An economic event is carried out by an agent and affects a resource. In REA, the notion of stockflow is used to specify in what way an economic event affects a resource. REA identifies five stockflows: produce, use, consume, take and give, where the first three occur in conversion processes and the latter two in exchange processes. The stockflows produce and take are positive stockflows in the sense that they increase the value of some resource for an agent – an economic event with a produce stockflow creates or improves some resource in a conversion process while an economic event with a take stockflow transfers a resource to the agent in an exchange process. Similarly, the stockflows use, consume and give are negative stockflows in the sense that they decrease the value of some resource for an agent – an economic event with a use or consume stockflow uses or consumes some resource in a conversion process while an economic event with a give stockflow transfers a resource from the agent in an exchange process. An *agent* is an individual or organization capable of having control over economic resources, and transferring or receiving the control to or from other agents [Ga07].

The *e3value value ontology* [Go00] aims at identifying exchanges of resources between actors in a business case and it also supports profitability analyses of business cases. e3value includes a graphical notation for business models. The basic concepts in e3value are actors, resources, value ports, value interfaces, value activities and value transfers. An *actor* is an economically independent entity. An actor is often, but not necessarily, a legal entity, such as an enterprise or end-consumer or even a software agent. A set of actors can be grouped into a market segment. A resource (also called *value object*) is something that is of economic value for at least one actor, e.g., a car, Internet access, or a stream of music. A *value port* is used by an actor to provide or receive resources to or from other actors. A value port has a direction: in (e.g., receive goods) or out (e.g., make a payment), indicating whether a resource flows in to or out from the actor. A *value interface* consists of in and out ports that belong to the same actor. Value interfaces are used to model economic reciprocity and value bundling. A *value exchange* represents one or more potential trades of resources between these value ports. A *value activity* is an operation that can be carried out in an economically profitable way for at least one actor.

Both the e3value and the REA ontologies include concepts on the operational level as well as the knowledge level [Fo97], where the operational level models concrete, tangible individuals in a domain, while the knowledge level models information structures that characterize categories of individuals at the operational level. In REA almost all classes on the operational level have a corresponding class on the knowledge level, which is generally not the case for e3value. The REA ontology distinguishes between event type (abstract transfer of categories of resources) and event (actual transfer of tangible concrete resources), both of which correspond to e3value's value transfer.

2.2 Service Systems

The notion of service system as proposed recently by Spohrer and colleagues [Sp08] is based on Vargo's Service-Dominant Logic [VL06]. No explicit ontology or modeling technique has been published yet, but we can identify a number of key concepts. A *system* is a configuration of *resources*. Some resources are *operants* that act on other (operand or operant) resources. A *service* is the application of resources to bring about changes that have *value* for another system. Services are performed in the context of *economic exchanges* – the mutual value creation by two or more interacting systems. So value is created in *interactions* between service systems. As a first rough approximation, we could make the following mapping to REA: resource – resource, system – agent, and exchange – exchange process. For service it is not so clear.

For service system interactions, Spohrer proposes the ISPAR model that follows a kind of Conversation for Action protocol for reaching agreement, but draws particular attention to failed interactions as sources of learning. For [Al08], the service interactions (or service encounters as he calls them) are part of a service value chain with a certain division of responsibilities.

2.3 Running Example

XYZ Financial Services (XFS) is a fictitious company introduced in [AG08] developing new services for baby boomers. The analysis of XFS has revealed that as these customers advance retirement age, their investment strategies are becoming more risk-averse and they tend to shift their savings from stocks and securities to saving accounts and certificates of deposits. Realizing also that no-interest checking and saving accounts are becoming an important source of revenue, XFS wants to design services that would attract and retain these customers.

3 A Unified Service Model

Based on a survey of the literature on services [UM03, Pr04, WS04, VL06, OA06, Sp08, among others] it is possible to identify five salient characteristics of services that apply both to business services and software services.

- A service is an economic resource, since it is an object that is considered valuable by actors and that can be transferred from one actor to another.
- A service is always provided by one actor for the benefit of another actor.
- A service is existence dependent on the processes in which it is produced and consumed, which means that the service exists only when it is consumed and produced. In other words, a service is consumed and produced simultaneously. In contrast to goods and information, a service cannot be stored for later consumption.
- A service encapsulates a set of resources owned by the provider. More precisely, when an actor uses a service in a process, she actually uses the resources encapsulated by the service. When an actor acquires a service in an exchange process, the customer does not get ownership rights on the encapsulated resources, but only use rights.
- A service is always governed by a policy. This means that when a service is used, the resources encapsulated have to be used in compliance with a number of rules formulated in a policy.

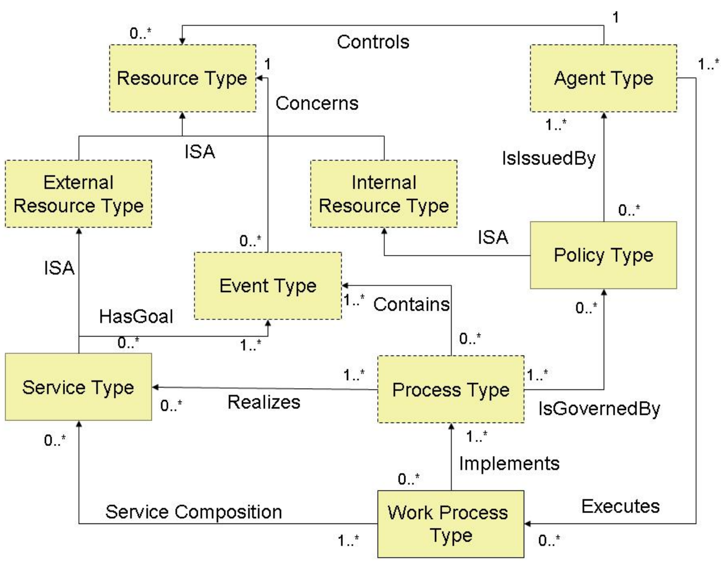

Fig. 1. Basic Service Ontology (core REA concepts in dashed boxes)

3.1 Service Ontology

As has been noted REA offers a comprehensive ontology of business concepts, but does not elaborate on the concept of a service. Our ontology (Fig. 1) is based on the core REA constructs Resource, Event, and Agent. All concepts are modeled at the knowledge level [Fo97]. In the following explanations of the concepts the word 'type' is sometimes omitted to reduce repetition.

A service is a resource as it is viewed as valuable by some agent and can be exchanged between agents. This is captured in the model by modeling Service Type as a subtype of Resource Type. As such, it automatically inherits all features of external resources, in particular:

- it can be exchanged between agents
- it is realized by a (conversion) process
- it can be used within a (conversion or exchange) process

A distinctive feature of a service is that it has a goal to modify (and hence add value to) other resources which is modelled by the association *hasGoal* from Service to Economic Event. For example, the goal of the hairdressing service is to convert the customer's hair. A service does not specify how it is to be realized, i.e. how its goals are to be achieved. Instead, a service can be realized in many different ways. This is captured in the model by means of the association *Realizes* from Process to Service. To realize a service, the process must achieve at least the goal of the service. To be precise, the event being the goal of the service is contained in the process realizing it.

Services are exchanged like other resources in an exchange process (not included in the Fig. 1) that meets the REA duality principle. This exchange process needs to be distinguished conceptually from the process realizing the service, but they are interwoven in time. The typical sequence of stockflow events is like this:

```
Resource (consume/produce) Service  (take/give) Service (consume/produce) Resource
                           Money    (take/give)  Money
```

On the left hand side, we find resources that are consumed or used to produce the service. In the middle, we see the exchange of the service –which is part of an exchange process. On the right hand side, the service is consumed resulting in a change of the customer's resource, which corresponds to the goal event.

A process in the REA sense represents only changes in value of resources as expressed through economic events; it does not address control flow or temporal aspects. In order to represent the latter, a new class Work Process has been introduced. A work process implements a process by executing a number of activities according to some composition (orchestration). So the work process may involve a number of (sub-) services. A work process is executed by an agent.

A policy is a set of assertions intended to govern some behavior. In REA terms, it should be classified as an internal resource that can be created, used and converted but not exchanged. Policies do not apply to a service as such, but to the process in which the service is exchanged or to the process in which the service is produced. An *external* (or public) *policy* constrains the set of events contained in an exchange process. For example, an exchange process involving a loan service should include the exchange of an exemption statement. Note that a contract can be modeled as a special type of external policy. An *internal policy* constrains the set of events in a conversion process that realizes a service (or any other resource, for that matter).

Our service ontology unifies the notion of business service (like hotel rooms, loans and hair dressing) and software service (e.g. web service by means of which hotel reservations can be made). The ontological representation of service is fully in line with SOA as a way of hiding process details to the service customer.

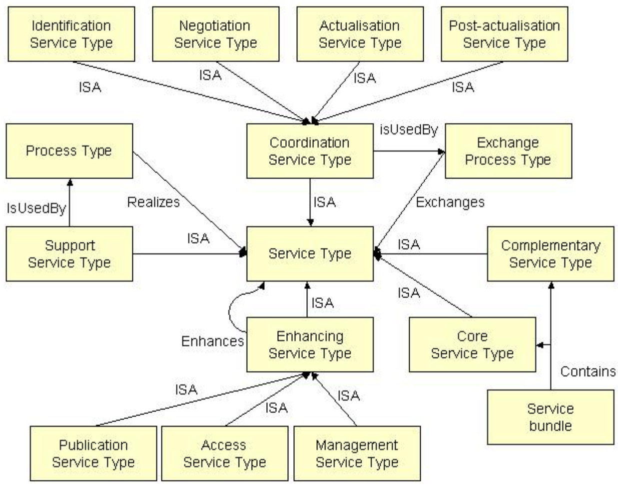

Fig. 2. Service Classification

3.2 Service Classification

The starting point of our service classification is the recognition of what we call core services. The reason for viewing these services as core is that they provide the raison d'être for an actor in a value network, as they specify what value the actor is able to provide to the network. Core services are easy to identify.

Given a set of core services, there is a need for a number of services that add to or can improve on the core services. We divide these services into four categories: complementary services enhancing services, support services, and coordination services.

Complementary service
A service complements another service if they are part of the same service bundle and their goals concern the same resource [We07]. For example, a gift-wrapping service complements a book sales service by having as goal to improve the book by packaging it in an attractive way. Thus, both services concern the same resource, the book. A service bundle is defined in our ontology as the services provided to a customer in the same service exchange process.

Enhancing service
An enhancing service is a service that adds value to another service (rather than to some other kind of resource). The possibility of enhancing services follows from our conceptualization of a service as a resource. The enhancing service has an effect on the quality of another service or some feature like visibility or accessibility. By definition, it is existence-dependent on the other service. On the basis of a review of cases, we have identified the following types of enhancing services:

- Publication service. A publication service provides information about another service (or any other resource) e.g. by means of a web page, a TV ad or a public service registry. Hence it produces visibility of the service. At the same time, it increases the knowledge of customers, so it has a dual focus.

- Access service. An access service gives an agent access to another service, i.e. the agent uses the access service to invoke the other service. An advantage of using an access service is that it can act like Facade object in Software Engineering [Ga95] that induces loose coupling by hiding the service details from the consumer. At the same time, it can contain medium-specific logic. Formally, if A is an access service to B, the following must hold: the goal of B is included in the goal of A, and B is a core service.
- Management service. A management service is a service that has as goal to maintain or optimize another service, typically by changing some feature of the work process. The management service can be seen as a service provided by some agent to the owner of the operational service being the customer. The service management includes several tasks that can be delegated to supporting management services, such as monitoring services, controlling services, authorization services, and evaluation services.

Support services
A service A supports a service B if A has as goal to produce B, or if A has as goal to produce a resource that is used in a process that produces B [Er07].

Coordination services
A coordination service is any service that supports (ontologically speaking "is used in") an exchange process. It is used for ensuring that communicating parties in a business relation are coordinated or synchronized. The value object exchanged can be a service but also a good.

Coordination services can be classified according to the stage in a business relation where the stages are identification, negotiation, actualisation, and post-actualisation [UM03]. For example, a catalogue service is instrumental in the identification stage. In the negotiation stage the terms and conditions of resource deliveries are formed (negotiation service, brokerage service) or reservations are made (reservation service). The actualisation stage is concerned with the actual deliveries of offered resources, including payment (payment service) whereas the post-actualisation stage may include all kinds of in-warranty services.

Coordination services are most elaborate when services are exchanged on a market and usually more simple when the services are exchanged between departments or individuals within an organization.

3.3 Service Layer Architecture

For a Service Layer Architecture that integrates business services and software services, we draw upon the enterprise ontology of Dietz [Di06] that distinguishes a social (performational) level, an informational level and a formational level. To illustrate: an order is a request at the social level, a message at the informational level and a document or file at the formational level.

In the context of IS design, an *informational* service is a software service that has as goal to produce information or enhance communication. A *utility* service is a service that is realized by means of IT hardware and supports informational services by storing, processing or transferring data. A *business service* is an economic service provided by an economic actor to fulfill a customer need. Both informational and utility services can be classified as supporting services (cf. section 3.2).

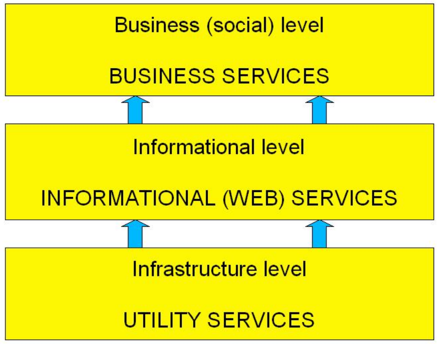

Fig. 3. Service Layer Architecture

Informational services are software services that are characterized by some economic autonomy. They can use other informational services and utility services as resources Informational services can be supported by management services whose aim is to maintain and improve the quality level of the managed service over time, and other enhancing services.

Utility services [Er07] are software services at the infrastructure level that are characterized by a certain economic autonomy (which makes them suitable candidates for outsourcing) and usually support more than one informational service. The value provided by the utility service is the storage of data or the execution of programs. Also at this level we can have managing and other enhancing services.

Informational services support the business level in different ways, depending on their focus:

- Decision support. Some business services are knowledge intensive. Take for example a credit-check service at the business level. Assessing the credit level of a customer is a responsible task assigned to some business actor. However, the service can make use of a web service that generates a credit rating on the basis of a database and/or business intelligence function. In this case, the web service at the informational level is a resource used by the credit-check process at the business level. It can even replace the credit-check if the service is completely delegated to the software. A decision service may be seen as the application of a set of business rules (that make up a policy) to generate a statement.
- Process support. Work processes at the business level can be supported by an informational service that takes care of the orchestration. Such services correspond to what [Er07] calls task-oriented services and what [Pa05] calls business process service.
- Information management support (for any kind of business service). Here we talk about what [Er07] calls entity-oriented information services that maintain and provide information about entities (business objects) and typically have a CRUD-style web service interface. For example, a web service that creates hotel reservations on request and stores them in the database is an information management service supporting a coordinating (reservation) service at the business level.

4 Service Identification – A Method Proposal

The service models introduced in section 3 enable the designer to start designing or identifying services in the business domain, and use this as input for the design of web services in the information system domain. As argued in [HJ07], this means that the web service designer does not need to start from scratch. The advantage of using value modeling is that it is already supported by established modeling techniques and methodologies. In section 4.1 we introduce a value-based service design method. In 4.2, this method is applied to the running example.

The first step in the method consists of creating a value model, which represents resource exchanges and conversions. However, in order to realize a value network as described by an e3value model, there is a need for a number of additional resources not explicitly visible in the e3value model. These resources are services required for managing the exchanges as well as the conversions of resources. Thus, the main purpose of the proposed method is to assist a designer in identifying these services in a systematic way.

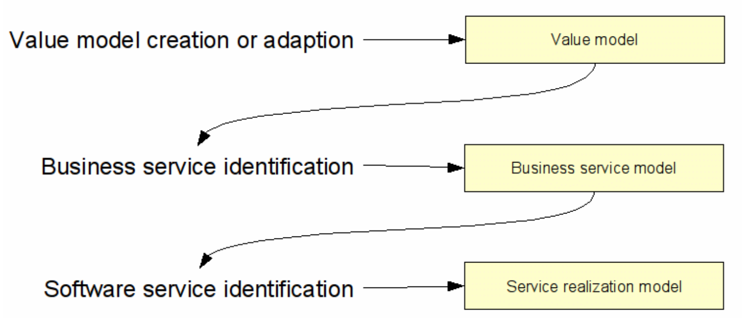

Fig. 4. Service design method schematic overview

4.1 A Service Design Method

Given the service ontology and architecture of section 3, a service design method can be developed that bridges the business and software level. The following description is not a complete cookbook, but is intended just to show how this bridging can be achieved.

Step 1: Value model creation or adaptation

In the value modeling step, we model the business activities from an economic perspective – in terms of value creation. In the current age of global networks, the focus cannot limit itself to a single actor only but should be on modeling value constellations of business parties. When modeling we do, however, take the perspective of one of the actors of the value network, to be called the focal actor, and identify services needed by that actor. We propose to use e3value modeling as introduced in section 2. Services do occur in the value model as a particular kind of value objects exchanged. The value model distinguishes service bundles and can identify complementary services. The value model also identifies supporting services needed to realize the core services.

The result of this step is a value model of the network that does abstract from IT services and from processes. The advantage is two-fold: first, this model is a relatively stable reference point when processes and particular service implementations evolve over time. Secondly, the model allows addressing business evolution at the appropriate level. For example, the reconfiguration of services for technical or logistic reasons is something that should be addressed at the information level. However, the replacement of one partner in the network by another one is a decision that needs to consider the strategic impacts.

The value model does not only contain core services, but can also represent quality features relating to these services, such as “convenience” or “low-budget” (so-called second-order values [We06]). They have an impact on how the service is provided, and hence may also influence the software service design.

Step 2: Business Service Identification

In the second step, the value model is used as a basis for identifying more business services in addition to the core services, complementary services and supporting services identified in step 1, in particular:

- enhancing services (access, information and management)
- coordination services that support the resource exchanges

The identification of business services includes the specification of business rules and policies governing the services (i.e., its exchange and conversion processes). A *table* notation can be used with four columns: the core service, its enhancing services, its coordination services and the applicable policies.

Step 3: Software Service Identification

The goal of this step is to identify services at the informational level and infrastructure level. A top-down approach can start from the (business) services identified in phase 2. Alternatively, a meet-in-the-middle approach can start from the available applications and identify services that embed these applications while at the same time supporting a business service. It is noteworthy that the informational services are exchanged between the IS domain and the business domain, usually represented by different departments each having their own responsibilities and autonomy, and so the service identification itself is very much a negotiation process.

The informational services are not logically derivable from the e3value model, but for various classes of business services (e.g. a reservation service, or a payment service) generic solutions can be applied. Such an approach can be seen as a concrete implementation of the suggestion made by [PH06] that service design should draw on business standards whenever possible.

Process support provides service orchestration. As work processes are governed by business rules/policies, it is recommendable to design decision services that incorporate these rules and can be used by the service orchestration.

Informational services can be further supported by management services whose aim is to maintain and improve the quality level of the managed service over time. Note that these are software services themselves.

Finally, the infrastructural support for the informational services has to be designed, including their sourcing and required SLA level.

4.2 Example: XYZ Financial Services

As described in section 2, XFS decided to target a particular market segment, the more risk-averse conservative senior citizens. A value analysis makes clear that there are opportunities for co-creation of value: for instance, certificates of deposit (CD) provide long-term funds to XFS, while at the same time offering the customer a safe place for their long-term savings. For the purpose of this example and because the data are limited, we have kept the value model very simple. One feature that we want to draw attention to is the second-order values "convenience" and "safety". To reach the target group, these can be an important competitive feature that should also be taken into account when considering the way the services are provided.

Step 2. Identification of business services. Evidently, there are two core services offered by XYZ to its customers: account management and money transfer (we ignore here the loan service that explores the money provided by this customer group but offered to another group). For lack of space, we do not elaborate on the enhancing services and lump them all together in a single access service.

Step 3. Software services. The core services identified in step 2 lend themselves to automation support. For transfer of money, XYZ needs to involve an inter-bank payment exchange service, and to access this external service, an access service must be added. The customer access service can also be supported by informational services. Examples are: product information service (that provides information about the product to potential customers), product contracting service (that allows a customer to open an account), a payment access service, and an account management service.

The payment access service is to be distinguished from the payment core service. The latter incorporates the basic functions of money transfer. The payment access service provides an interface to this service. Since convenience is a strategic second-order value, it is important to offer one or more user-friendly interfaces. For example, a pay-by-phone interface and a web interface. The distinction between payment access service and payment core service is apparently not made in SOMA. However, it is very useful way of improving business agility.

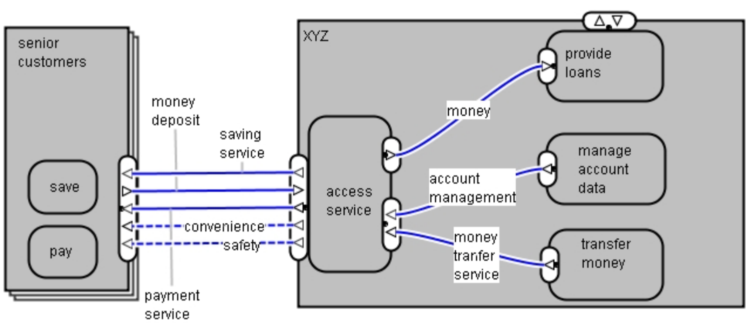

Fig. 5. Initial value Model for XYZ Case, business level only

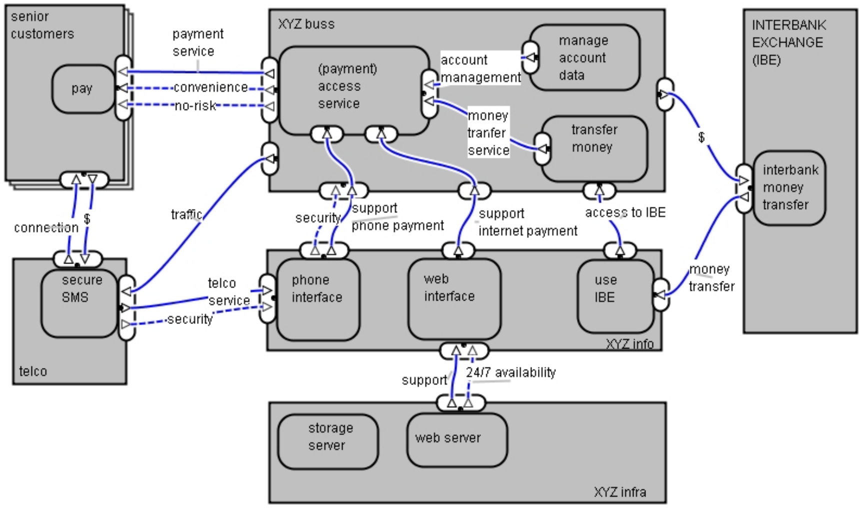

Fig. 6. Three-layer service model (for payment service only)

Information support services can be identified for the business objects involved, such as Customer, with a CRUD-style interface (create customer, update customer, delete customer, update address, etc).

In the case of a meet-in-the-middle approach, the informational services identified so far should be confronted with the existing legacy applications.

Some remarks

- when comparing our value-based model results with those from SOMA – however sketchy both are – one difference is that e3value tends to draw attention to value networks. In the XFS case, it is clear that XFS cannot deliver the proposed services on its own. In particular, a pay-by-SMS service is the result of a co-creation of value that benefits all parties involved in some way. In contrast, SOMA seems to restrict itself to the internal software services in the company.
- an advantage of the value-based method is that it not only identifies services, but can also record second-order values, corresponding to extra-functional properties such as security or availability that should be considered in SLA agreements.
- the value-based method clarifies the dual focus of informational services, such as an XFS web interface. It provides support to the business (the coordination services). At the same time, it is a kind of access service that serves the customer in gettingt to the payment service. The effect of this dual focus is that it has at least two goals: satisfy the customer (who interacts with the service) and satisfy the business (that has set up and maintains the service).

5 Conclusion

In this paper, we have developed a unified model of services. On the basis of the service ontology, we have proposed a service design method that starts from a value model and helps to identify core and enhancing services. Subsequently, it can help identifying possible web services. Whereas most SOA design methods consider service design as a software engineering problem, we consider it as both a business engineering and software engineering problem.

Topics for future research include validation by means of more cases, as well as the development or reuse of graphical notations supporting the service design. The graphical notation of e3value is useful but has problems with rendering larger models.

References

[An06] Andersson, B., Bergholtz, M., Edirisuriya, A., Ilayperuma, T., Johannesson, P., Gordijn, J., Grégoire, B., Schmitt, M., Dubois, E., Abels, S., Hahn, A., Wangler, B., Weigand, H.: Towards a Reference Ontology for Business Models. In: Embley, D.W., Olivé, A., Ram, S. (eds.) ER 2006. LNCS, vol. 4215, pp. 482–496. Springer, Heidelberg (2006)

[Al08] Alter, S.: Service system fundamentals: Work system, value chain, and life cycle. IBM Systems Journal 47(1), 71–86 (2008)

[AG08] Arsanjani, A., et al.: SOMA: A method for developing service-oriented solutions. IBM Systems Journal 47(3), 377–396 (2008)

[BP07] OASIS Web Services Business Process Execution Language Version 2.0 (2007), http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html

[Di06] Dietz, J.: Enterprise Ontology - Theory and Methodology. Springer, Berlin (2006)

[Er07] Erl, T.: Soa: principles of service design. Prentice-Hall, Englewood Cliffs (2007)

[Fo97] Fowler, M.: Analysis Patterns. Reusable Object Models. Addison-Wesley, Reading (1997)

[Ga07] Gailly, F., Poels, G.: Towards Ontology-driven Information Systems: Redesign and Formalization of the REA Ontology. Working paper, Univ. Ghent (2008-03-27), http://www.FEB.UGent.be/fac/research/WP/Papers/wp_07_445.pdf

[Ga95] Gamma, E., et al.: Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, Reading (1995)

[Ge99] Geerts, G., McCarthy, W.E.: An Accounting Object Infrastructure For Knowledge-Based Enterprise Models. IEEE Int. Systems & Their Applications, 89–94 (1999)

[Go00] Gordijn, J., Akkermans, J.M., van Vliet, J.C.: Business modeling is not process modeling. In: Mayr, H.C., Liddle, S.W., Thalheim, B. (eds.) ER Workshops 2000. LNCS, vol. 1921. Springer, Heidelberg (2000)

[HJ07] Henkel, M., Johannesson, P., Perjons, E., Zdravkovic, J.: Value and Goal Driven Design of E-Services. In: Proc. of the IEEE Int. Conference on E-Business Engineering (Icebe 2007). IEEE Computer Society, Washington (2007)

[Hr06] Hruby, P.: Model-Driven Design of Software Applications with Business Patterns. Springer, Heidelberg (2006)

[Mc82] McCarthy, W.E.: The REA Accounting Model: A Generalized Framework for Accounting Systems in a Shared Data Environment. The Accounting Review (1982)

[NL07] Nayak, N., Linehan, M., et al.: Core business architecture for a service-oriented enterprise. IBM Systems Journal 46(4), 723–742 (2007)

[NR93] Norman, R., Ramirez, R.: From value chain to value constellation: Designing interactive strategy. Harvard Business Review, 65–77 (July-August 1993)

[OA06] OASIS. Reference Model for Service Oriented Architecture 1.0 (2006), http://www.oasis-open.org/committees/download.php/19679/soa-rm-cs.pdf

[Os04] Osterwalder, A.: The Business Model Ontology, Ph.D. thesis, HEC Lausanne (2004), http://www.hec.unil.ch/aosterwa/ (last accessed, 007-07-01)

[Pa05] Papazoglou, M.: Web Services Technologies and Standards. ACM Computing Surveys (2005)

[PH06] Papazoglou, M., van den Heuvel, W.J.: Service-oriented design and development methodology. Int. Journal of Web Engineering and Technology 2(4), 412–442 (2006)

[PK08] Prahalad, C.K., Krishnan, M.S.: The New Age of Innovation: Driving Cocreated Value Through Global Networks (2008)

[Pr04] Preist, C.: A Conceptual Architecture for Semantic Web Services. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, pp. 395–409. Springer, Heidelberg (2004)

[Sp08] Spohrer, J., et al.: The Service System Is the Basic Abstraction of Service Science. In: Proc. HICSS (2008)

[UM03] UN/CEFACT Modelling Methodology (UMM) User Guide (2003), http://www.unece.org/cefact/umm/UMM_userguide_220606.pdf 2008-02-19

[UN08] United Nations, Dept. of Economic and Social Affairs. Common DataBase (CDB) Data Dictionary (February 19, 2008), http://unstats.un.org/unsd/cdbmeta/gesform.asp?getitem=398

[Us96] Uschold, M., Gruninger, M.: Ontologies: Principles, Methods, and Applications. Knowledge Engineering Review 11(2), 93–155 (1996)

[VL06] Vargo, S.L., Lusch, R.F., Morgan, F.W.: Historical Perspectives on Service-Dominant Logic. In: Lusch, R.F., Vargo, S.L., Sharpe, M.E. (eds.) The Service-Dominant Logic of Marketing, Armonk, NY, pp. 29–42 (2006)

[We06] Weigand, H., Johannesson, P., Andersson, B., Bergholtz, M., Edirisuriya, A., Ilayperuma, T.: On the Notion of Value Object. In: Dubois, E., Pohl, K. (eds.) CAiSE 2006. LNCS, vol. 4001, pp. 321–335. Springer, Heidelberg (2006)

[We07] Weigand, H., et al.: Strategic Analysis Using Value Modeling–The c3-Value Approach. In: HICSS 2007, p. 175 (2007)

[WS04] W3C. Web Services Architecture W3C Working Group (2004), http://www.w3.org

[Zi04] Zimmerman, O., Krogdahl, P., Gee, C.: Elements of Service-Oriented Analysis and Design (2004), http://www-128.ibm.com/developerworks/library/ws-soad1/