Value Webs:
Ontology-Based Bundling of Real-World Services

Hans Akkermans, Ziv Baida, Jaap Gordijn

Free University Amsterdam VUA
Faculty of Sciences, Section Business Informatics FEW/BI
De Boelelaan 1081a, NL-1081 HV Amsterdam, The Netherlands
Phone: +31 20 444 7718, Fax: +31 20 444 7728
Hans.Akkermans@akmc.nl, ziv@cs.vu.nl, gordijn@cs.vu.nl

Nieves Peña, Ander Altuna, Iñaki Laresgoiti

Labein
Parque Tecnológico de Bizkaia, Edificio 700
ES-48160 Derio, Bizkaia, Spain
Phone: +34 94 607 3352
npena@labein.es, aaltuna@labein.es, lares@labein.es

Preprint of an article to appear in IEEE Intelligent Systems, Vol. 19, No. 4, July/August 2004
(Special issue on Semantic Web Services, guest editors Ora Lassila and Terry Payne)

Abstract

Real-world service production involves a mixture of intangible and physical elements, including goods, acts as well as people required for delivery. In many cases, companies are not offering just a single service to a customer, but a more or less interrelated collection of them: service bundles. This makes it possible to have a broader and better coverage of customers' needs, and at the same time achieve scale and scope efficiencies in service cost by sharing and reusing service elements. In this article we show that business analysis and design of real-world service bundles stand to benefit from an interdisciplinary semantic approach. Specifically, we discuss a component-based ontology for real-world services, associated methods and tools for visual modelling, and knowledge-based configuration of service bundles, with applications from online events organization and the energy industry.

Keywords: semantic profiling of services, ontological engineering, knowledge-based configuration methods, validation by industrial use cases.

1. Semantic Approach to Service Business

The advanced economies in the world are service economies. Economic production today is no longer dominated by industrial production and sales of physical goods, but by the service sector. Typical figures amount to about 70% of GNP in most countries. The notion of what we will call real-world services covers a very broad range, from banking and insurance, consultancy, education, Internet provision, to conference events, music, and haircuts.

Real-world service production involves a mixture of intangible and physical elements in the real world, including goods, acts as well as people required for delivery. Real-world, i.e. non software-based, services significantly differ from Web Services, usually defined as software functionality accessible and configurable over the Web. Because of the economic, social, and business importance of the service concept in general, we believe it is necessary to rethink what 'service' actually means in an ontological and computational sense. The focus of most current semantic approaches is on the computer science and software aspects of services insofar they lie within the Web environment. In our view, the state-of-the-art Web programming view on services is incomplete and unnecessarily limiting, as it ignores significant elements of the service context. Our work shows that ontology research and application is able to cover and handle more.

First, semantic web service research still has a lack of convincing realistic industrial use cases that are the basis of practical use and empirical validation of proposed ontologies and problem-solving methods. Second, an interdisciplinary approach to services that not only draws from computer science and AI, but also from economics, systems theory, and business practice, leads to ontological descriptions that capture much richer service profiles, moreover of hitherto unseen services. For example, there is a need to make a clearer distinction between requester (customer) and supplier-oriented service descriptions: their different roles and viewpoints are reflected in different ontological commitments, service descriptions *and* dynamics. Service industry practice further shows that so-called non-functional aspects (such as quality features) are empirically important to service selection, composition, marketing and sales. Third, implementation of richer service descriptions requires a variety of flexible Semantic Web reasoning methods beyond (description or frame) logics or process flow-oriented algorithms such as AI-style planning. Accordingly, this article gives a brief overview of what a broader interdisciplinary approach brings to the semantics of services. It also outlines their application to some novel industrial use cases, for which there are moreover currently no working solutions by mainstream non-semantic methodologies.

2. Service Bundling

This article particularly discusses *service bundling*, as it is a major business issue in all industry sectors that are faced with increasing competitive market pressures or technological innovation (or both). Companies do not usually offer just a single service to a customer, but a more or less interrelated collection of them: this makes it possible to have a broader and better coverage of customers' needs, and at the same time achieve scale and scope efficiencies in service cost by sharing and reusing service elements.

To handle service bundles we have to deal with two simultaneous dimensions of complexity. First, real-world services are known for their high variability, in that they consist of very many diverse elements. In the trading of commodity goods, which is still dominating current eCommerce, a componential approach is commercially by now well developed (witness for example companies such as Dell and Cisco). In the service sector, component-based support for the bundling of services is yet inadequate. A likely reason is the high inherent variability and intangibility of real-world service elements, which easily escapes automated description and design.

Service bundling is different from web service composition. The latter is done at the workflow or service process level. Associated computational techniques such as planning thus incorporate ontological commitments to temporal notions such as states and their ordering. In contrast, service bundling is a business development activity that rather lives at the service profile level (to use the DAML/OWL-S terminology). Its declarative specification does not make commitments in terms of time and flow, but only to mereological and topological notions such as connectivity. This leads to the need for other reasoning methods. For example, we describe below configuration techniques that are adaptations to the service domain of problem-solving methods originating from large-scale technical systems design. Even more advanced are some other e-service industry scenarios we are working on (but not discuss in this paper), where there is a need for dynamic multi-dimensional resource allocation methods collaborating with distributed digital control systems.

A second dimension of complexity is the fact that many services are co-productions of different suppliers. A commonly quoted aspect of services is that customers themselves often take part in their production. For instance, many fast-food outlets suppose their customers to clean the table. Another common example is Internet service provision, which requires IP facilities as well as physical telecoms connectivity. These usually come from different companies involved in one and the same service offering to the market. So, one must be able to model and implement a co-production network of component-based service supply.

Our research has focused on improving the support for this. Specifically, we developed a generic component-based ontology for real-world services. This service ontology is first of all a formalization of concepts that represent the consensus in the business science literature on service management and marketing. In addition, our ontology embodies a number of systems-theoretic notions that specify how diverse service elements, seen as individual *'Lego-blocks'*, can be connected to each other to form a larger service system: a service bundle. Here, various topological connection and typing rules play a key role, in a way reminiscent of how complex engineering systems are designed. Furthermore, we have expressed our service ontology in a graphical, network-style, representation plus associated support tools that facilitate end-user modelling of services. Finally, to enable profiling and analysis of service bundles, automated knowledge-based configuration methods are aiding business designers and analysts. We illustrate our semantic support tools with a few of our recent practical applications, from online events design and the energy industry.

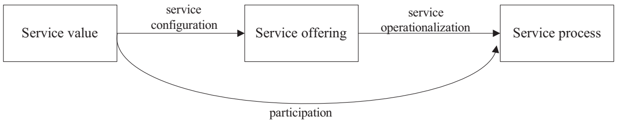


Figure 1. The OBELIX service ontology distinguishes three top-level viewpoints.

3. The OBELIX service ontology

Service is by now a rather overloaded term. Until recently, research on services was the domain of business schools, where since the late 70s a wealth of literature on service marketing and management has been produced. This literature gives a general framework on what (real-world) services are and in what sense they are different from physical goods. Widely used recent texts are [1] and [2]. An important observation for ontology work is that this literature now shows a consensus on many important points. Representative definitions of what a service is often contain the same recurring elements. For example:

- Zeithaml and Bitner: "…services are deeds, processes and performances …"
- Kotler: "…any act or performance that one party can offer to another that is essentially intangible …"
- Grönroos: "…activities …of a more or less intangible nature that normally …take place in interactions between the customer and service employees and/or physical resources or goods and/or systems of the service provider, which are provided as solutions to customer problems".

This literature has provided input material for our ontological description of real-world services. We distinguish three interrelated top-level viewpoints in our ontology, as sketched in Figure 1: service value, service offering, and service process. The *service value* perspective describes the service from a customer's perspective; the service offering perspective describes it from a supplier's perspective; the service process perspective describes how the service offering is put into operation. The service value and offering perspectives extend our e^3-*value* business model ontology that provides an ontological and graphical approach to networked business modelling, as discussed in previous articles [3, 4].

The *service value* perspective, which is often overlooked in IT approaches, takes the customer viewpoint: it expresses the customer needs or demands that should be satisfied by acquiring a service of a certain quality, in return for a certain sacrifice. An object model of the major concepts in this sub-ontology is shown in Figure 2. The ontologies discussed in this article have an RDFS implementation, and their domain instances specific to individual industrial use cases are automatically generated from this and exported by our tools in RDF format. Within this sub-ontology, service quality is modelled by importing existing quality frameworks used in business (e.g. the SERVQUAL model). Quality features are important in profiling and composing services because they serve as selection handles for customers. Examples are hotel star classifications, but we also find them in eCommerce: low resolution digital photographs may be downloaded for free, but one has to pay for high resolution quality. Sacrifice includes price, but also intangible relationship costs such as customer effort spent in service co-production, and inconvenience costs (waiting in line, search and download time).

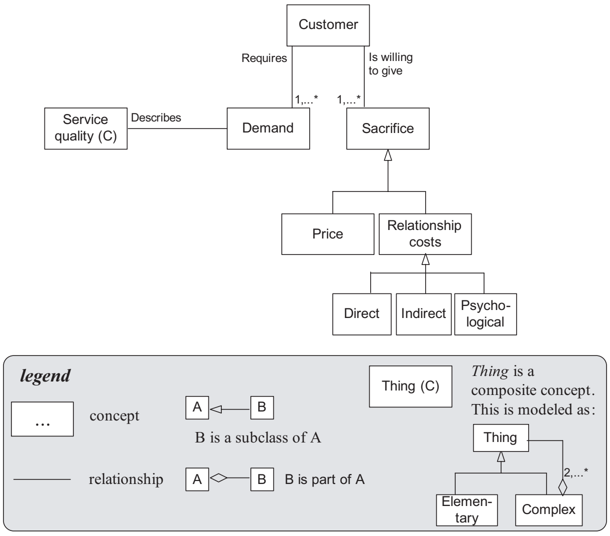


Figure 2. Service sub-ontology representing the service (customer) value perspective.

The *service offering* perspective (Figure 3) represents the supply-side viewpoint upon a service. This perspective centers around the concept of service element. Figure 3 only shows the ontological elements necessary to understand the business nature of this perspective. **Service elements** represent what a supplier offers to its customers. It is what the business literature defines as service, a business performance of a typically intangible nature. Examples are money transfer, transport, haircuts, Internet radio, electricity supply. A service element can be decomposable into smaller service elements, as long as these smaller elements can be offered to customers separately, possibly by different suppliers. The business literature distinguishes different roles or functions of service elements from a supplier perspec-

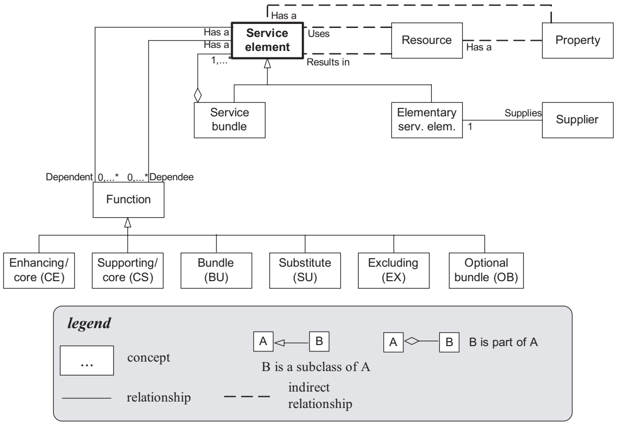


Figure 3. Service sub-ontology representing the service offering perspective.

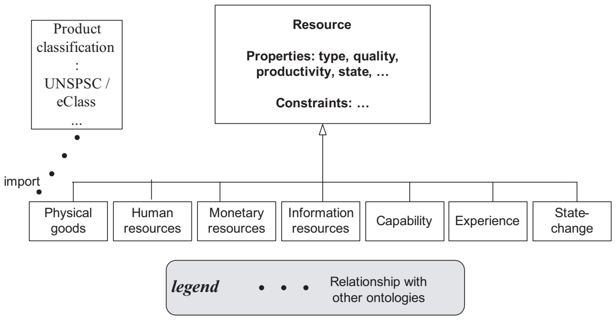


Figure 4. Resources are typed inputs or outcomes of real-world service elements.

tive: a core service (the main business), a supplementary service with a supporting role (i.e. necessary to make the core service possible) or a supplementary service with an enhancing role (improving the core service value by adding extra features). Service elements might also be substitutes of or exclude one another. These functional rules concerning individual service elements are important in computing feasible service bundles. A **service bundle** is a set of service elements that can be provisioned together as a whole compatible with these service ontology rules.

We will not discuss the service process perspective here, because there is already a lot available in the literature that can be adopted (see e.g. ebXML, WSFL, BPEL4WS, or the DAML/OWL-S service process [5]) and Petri Net semantics; our work focuses more on service profiles. However, an important concept that is key to both the service offering and service process perspectives is that of **resource**: anything that is either an input or an outcome of a service element. In real-world services, resources are typed (see Figure 4; note that this typing is absent in the DAML/OWL-S notion of resources). These types clearly show the mentioned great variability of services. Some services may result, at least partially, in rather tangible results (for example, physical things such as a meal, or a person available to do things for you, say, watching over your children when you go to the theater), whereas other service outcomes may be highly intangible. Visiting a theater or the Van Gogh museum aims to produce a memorable customer experience. A haircut, transport, or car checkup service basically results in a state change. Many services produce an even more abstract outcome, a capability or right to do something: a credit card, formal education, a driver license, clearing music rights. Computationally, this typing of resources is one important feature we use in our automatic configuration of services, as it provides additional constraints on the possible outcome-input connections between different services elements.

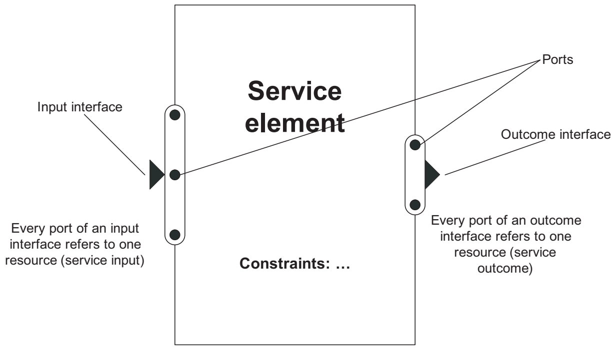
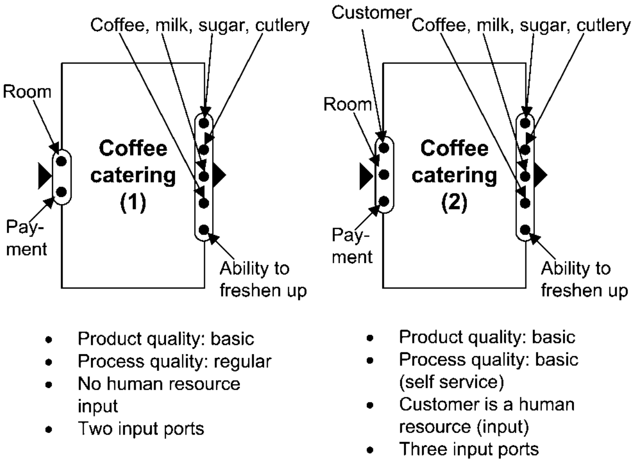


Figure 5. Visualization of a service element.

4. Graphical modelling and configuration

Our OBELIX ontology for real-world services has been developed using ontology editors such as ONTOEDIT and PROTÉGÉ and is computationally available in RDFS form. Testing in industrial use cases from very different sectors is the main empirical mechanism for ontology validation. However, not many practitioners will be able to do their work from such representations directly. Therefore, we have developed *graphical* representations for services and their composition that are much more intuitive for domain experts and practitioners. We show in this section how graphical techniques help model services in a configurable way.

Figure 5 shows the visualization of a service element in our ontology. Service elements are visualized similar to



(a) Coffee catering (1) (b) Coffee catering (2)

Figure 6. Different instances of the same service element within a bundle for conference events.

components in an engineering system, so that knowledge-based configuration methods can automatically create composite service bundles. Configuration is a well-researched design task, simplified by the availability of a set of predefined components, connections, and associated parameters and constraints [6, 7]. The possible connectivity that a component can have with other system parts is modelled by so-called **ports** [8]. In technical systems ports are often typed: a wall outlet actually represents ports for making electrical connections. In service modelling, we utilize analogous structuring ideas. Every service element has ports of two types: input ports and outcome ports. The provisioning of any service element requires **resources**, and results in the availability of other resources. A port indicates a certain resource that is either a pre-requisite for carrying out this service element (input port), or the result of executing this service element (outcome port). A special characteristic of our service ontology is that resources are *typed* (see Figure 4). Each resource has a corresponding port. An example of a service element from conference events organization is shown in Figure 6.

A service element can only be provisioned if all required inputs are available, and it results in the availability of all outcome resources. The set of all input ports, respectively all outcome ports of a service element form the element's **input interface**, respectively **outcome interface**. A further concept required to carry out knowledge-based configuration of service bundles are **constraints**. We distinguish between constraints on service-related values (such as quality properties of associated resources or conditional input-outcome constraints on the relation between quality level and payment), and constraints on the role dependencies between service elements. The latter kind of constraints is referred to as **functions**, see Figure 3. In this way, we have achieved an ontological and visual description of service elements as configurable components in a larger service bundle system.

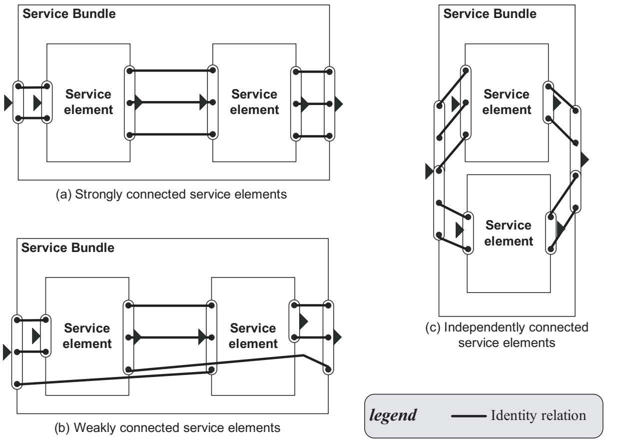
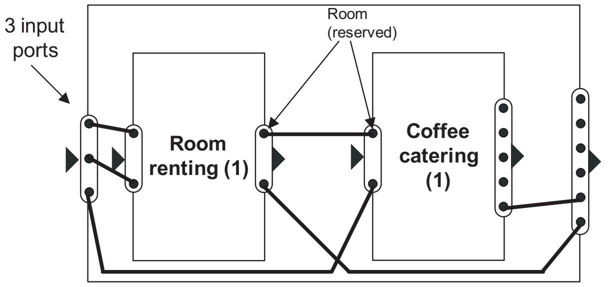
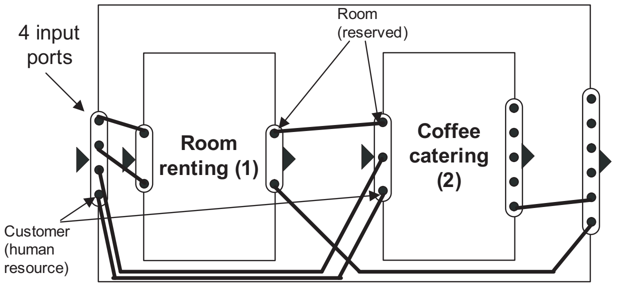


Figure 7. Visualization of different kinds of service bundling.

Figure 7 visualizes different ways to construct a service bundle from several service elements. As a composite service element, a service bundle also has an input and an outcome interface. The input and outcome interfaces of a service bundle are identical to the union of the input and outcome interfaces of all service elements included in that service bundle. Two exceptions to this general rule exist: (1) certain resources can be *consumable* more than once in several service elements (particularly *information resources* can be used multiple times); (2) when resources have the *compositeness* property, multiple resources of the same type may be modelled as a single resource. For instance, when two service elements are bundled, and both require a *payment* input, these two inputs can be composed into a single *payment* resource. Very often the price of such a bundle is lower than the sum of the separate prices. The input interface of a service bundle must provide all inputs of all service elements that are part of this bundle, unless they are provided internally (one service element may produce an outcome that is consumed as an input by a different service element). Connection links in Figure 7 between ports mean that one port uses a resource that another port provides. Now, we can configure real-world service bundles using the well-known problem-solving methods for knowledge-based configuration tasks.



(a) Room renting (1) and coffee catering (1)



(b) Room renting (1) and coffee catering (2)

Figure 8. Glassbox view of a small service bundle within conference events.

Note that service bundling is treated as a recursive notion: any service bundle is a service element. A small example from online events organization is given in Figure 8. The associated blackbox view containing only the inputs and outcomes of a service bundle as a whole is often the most useful one for the end customer in knowing whether his/her external requirements are satisfied. The glassbox view showing the internal service element structure of a service bundle is the one of particular interest to service suppliers. The combination thus yields a very rich approach to service profiling in our OBELIX ontology. Figure 9 gives an impression of the OBELIX service CASE tool architecture. The tool enables graphical modelling of services. It also contains the underlying business rules from the generic service ontology, so that users may be notified about any omissions or mistakes when ontological rules are violated. Finally, our e^3-service tool automatically generates the RDF representation of the service domain ontologies from the graphical model; this RDF representation is employed by our configuration engine.

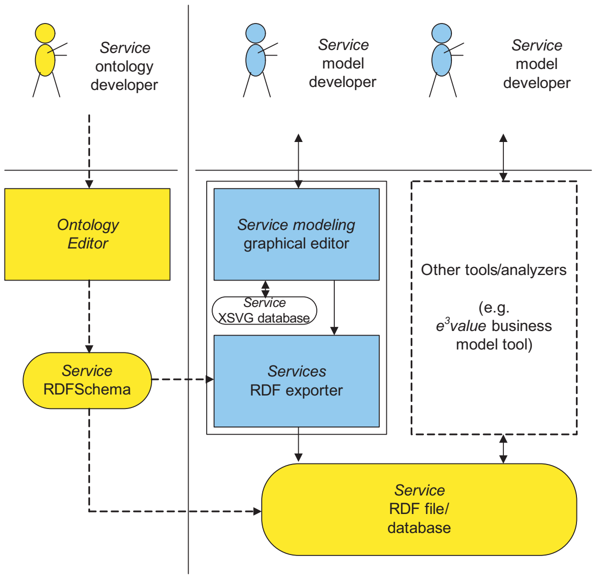


Figure 9. The OBELIX e^3-service tool.

5. Supporting business applications

Along the above lines, we modelled numerous services that can be offered to customers in a bundle in various industrial case studies. One of them is a business analysis of service bundling related to energy supply, carried out for a power distribution utility in Norway. In this case, individual service elements include: electricity supply, electricity transmission, hot water distribution (for room heating and tap water), broadband Internet access, IT services, sales and installation of electrical appliances (heat pump and energy control system, to reduce energy consumption and to regulate temperature), and remote control services. An e^3-service tool screenshot for this case study is shown in Figure 10; it also shows the dependencies between service elements (CS means that a service element is a necessary part of the bundle in relation to the core service, OB means that it is an optional additional feature in a bundle, EX means that two service elements exclude one another and so cannot be part of the same bundle). The screenshot shows a draft model that serves as input to our configuration engine. Other figures in this article show model visualizations of the situation *after* configuration has taken place.

Different business considerations were driving this case study. Electricity is a common mass commodity in a competitive market (power markets are liberalized world-wide; Norway did so already in 1991), and to differentiate oneself from the competition as a power company is not easy. Since customers have a free choice regarding their energy supplier, one strategic option is to offer better and richer cus-

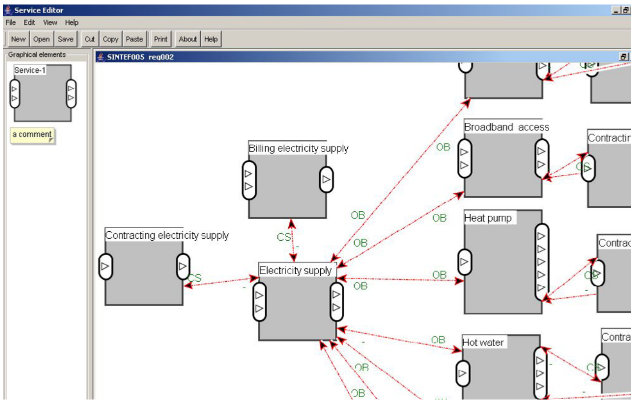


Figure 10. Screenshot from the OBELIX e^3-service tool, showing a use case for energy service bundling.

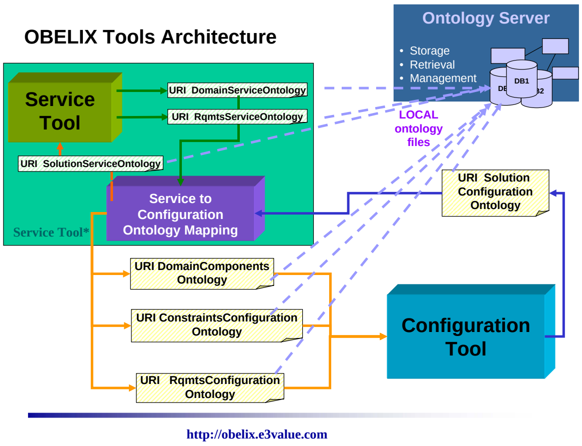


Figure 11. Collaboration of the OBELIX ontology tools in service configuration.

tomer service to achieve customer retention. Moreover, additional customer service should preferably exploit cost advantages by sharing and reusing existing service elements (e.g. billing), and improve return on investment of existing infrastructure by using it as the carrier for more services (asset management: e.g. fiber optics is in place to control the power grid, but there is enough capacity to support broadband Internet access). Thus, the starting questions are: what are feasible new service bundles here, such that they are financially attractive for both customer and supplier, as well as technically realizable?

Once all individual service elements are modelled, with their mutual functional dependencies, and the set of customer requirements is established, a knowledge-based configurator produces all service bundles that are feasible with respect to all posted constraints and requirements. The problem solving method used is explained below.

Configuration Algorithm

The configuration method we use for service bundling is based on a generic configuration design ontology (with core concepts: components, connections, associations, constraints, and requirements). Upon input to the configurator, the various elements of the service ontology are mapped onto the configuration ontology; next, the configurations are computed. Figure 11 shows how the OBELIX ontology-based tools collaborate in this process of service configuration. The configuration algorithm then works as follows.

Inputs:

1. A components ontology in which components are described (service elements are components), including their associated resources (inputs and outcomes).
2. A constraints ontology in which various types of constraints are defined. These include mainly: (a) Functions: dependencies between service elements; (b) Inherent constraints: no loops are permitted, an outcome port is to be connected to an input port, and various other business rules stemming from the service ontology.
3. A requirements ontology, which describes restrictions on desired inputs/outcomes to guide the configuration process: they express the end customer demands by defining the type of resources required and the constraints on their property values.

Outputs: An ontology (in RDF form) that provides all feasible service bundles such that the computed solutions:

1. explicate what service elements are part of the bundle, and how they are connected through the resources used;
2. obey all given constraints, and
3. satisfy all input customer requirements.

The configuration process has two phases. Suppose the user has as requirements to get resources $R1$ and $R2$, and that $R1$ is provided by service element $X1$, and $R2$ by services $Y1$ and $Y2$.

1. **High-level configuration:**
   - *Identifying initial elements for the service bundle*: Based on the given requirements, the configurator knows which resources are required ($R1$, $R2$). It searches for all service elements that provide the requested resource types, and finds that

$R1$ is provided by service $X1$, and $R2$ by services $Y1$ and $Y2$.

- *Applying functions (dependencies between service elements)*: Some service elements may require/exclude others. Say, we have the following functions: (a) Excluding($X1$, $Y1$); (b) Core/enhancing($Y2$, $Z1$). Applying the first function means that $\{X1, Y1\}$ cannot be put together into the same service bundle. The second function means that $Y2$ may — but does not necessarily have to — be bundled with service element $Z1$. Hence, two possible service bundles are inferred: $\{X1, Y2\}$ and $\{X1, Y2, Z1\}$. Each of them is a good solution. Functions are applied to all considered service elements in a bundle; whenever a service element is added to a bundle, available function information is applied again to check consistency of the bundle.

This results in a set of feasible service bundles. These bundles do provide the required resource types, but do not yet guarantee that all values of resource properties as specified by the customer are satisfied. The latter task is performed by the next step of the algorithm.

2. **Detail-level configuration:** In this phase we create connections between ports of the service elements that are included in a bundle. This step is repeated for every service bundle that was generated by the first phase. As a heuristic, it is required to connect as many ports as possible within a bundle. Two ports within a bundle may be connected if:
   - They belong to different service elements, and
   - They have different types (input, outcome), and
   - They require/provide the same resource (same resource type, plus same — or better — *comparable* property values).

   All restrictions regarding the range of service property values are then checked by the configuration engine. Standard constraint solvers turn out to be adequate to do this. Any input/outcome port that is not connected to other ports within the same bundle, will appear in the input/outcome interface of the service bundle. Note that any high-level bundle may have multiple detail-level solutions.

The service configuration process will generally result in a set of possible service bundles. One of the answers we get from the configurator in this case is visualized in Figure 12.

The next step of the service business analysis is, as indicated in Figure 9, to feed the configuration results into another tool for analyzing the financial attractiveness of new business models. For this we have developed the e^3-*value* tool, which is based on our ontology for networked e-business models (discussed in previous articles [3, 4]). An example of a new business model reflecting computed service bundling decisions resulting from this tool is shown in Figure 13. This modelling and simulation tool further carries out quantitative net present value calculations of the cash flows between all actors in spreadsheet form, in order to establish the profitability of new business models and service bundles. It furthermore gives the possibility to carry out numerical sensitivity analyses of key parameters by computing the results of what-if scenarios.

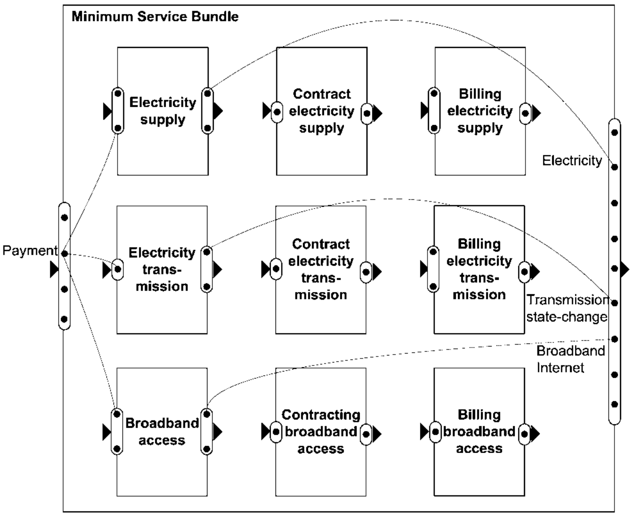


Figure 12. One computed service bundle including electricity supply (the core service) and broadband Internet access.

6. Discussion and Conclusion

The OBELIX ontology library for real-world services has been validated by a variety of industrial case studies, and provides several new capabilities not available in current Web service modelling frameworks and ontologies.

Related Work

The OBELIX service ontology and application work is mainly complementary to the DAML/OWL-S framework [5]. Overall, our work adds a needed interdisciplinary perspective. Its contributions and strengths are (purposely) at the service profile level, where we offer (1) business-relevant extensions important to any generic service ontology; (2) additional computational methods such as configuration. We believe that, specifically, the value-based approach to and recursive notion of service bundles, strong typing of resources, computational treatment of non-functional service properties such as quality, the ontolog-

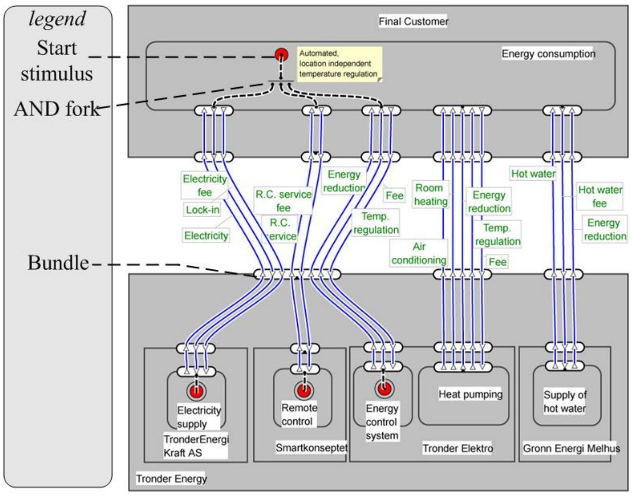


Figure 13. A possible new business model reflecting computed service bundling decisions, as produced by our e^3-*value* tool. This tool further helps analyze projected cash flows to assess the financial attractiveness of envisaged service bundles.

ical elaboration of the service requester/supplier distinction, the explicit modelling and handling of the dependencies between configurable service elements, and last but not least our visual representations of service models are significant OBELIX contributions to service design, composition and analysis. The DAML/OWL-S work has, in contrast, its main computational focus at the service process level and its WSDL grounding, but there seem to us to be no major technical obstacles in linking this to the OBELIX service profiling ontologies and methods. The WSMF/WSMO work (www.wsmo.org) is still in a very early and underspecified stage. In essence it is an extension of UPML [9], which on its turn is a distributed, Web version of the CommonKADS knowledge modelling framework [10]. It therefore has a knowledge modelling outlook that is similar to OBELIX. It mentions the distinction between web services and real-world services, but the general concepts inherent to services are, in contrast to OBELIX, not implemented in the ontology and framework itself. DOLCE [11] has like WSMO a weak intrinsic conceptual notion of what services are, but is of interest as it provides mechanisms for further formalization and axiomatic grounding of service ontologies such as ours in terms of foundational ontologies.

Reflections on ontology development and use

We view development of ontologies, as formal conceptualizations, essentially as a key scientific method for theory formation in Information Science that aims to bridge human and computer understanding. An ontology may then be called ‘good’ if it is used in and validated against several independent and external business scenarios and industrial use cases. In view of the current state of the art of semantic web services, the importance of industrial use case testing can hardly be overstated. We therefore carried out several real-world case studies in different industry sectors. They are practically useful in their own right, and also help convince practitioners about the added value of semantics-based methods. Moreover, case studies turn out to be methodologically important as stress tests and ‘triangulation methods’ for ontologies, reasoning techniques, and support tools. As a result, the OBELIX ontologies have undergone significant refinement over the past years as a result of their industrial application. Specialization of a generic service ontology to a specific domain is best done by domain specialists themselves, with a ‘helpdesk’ role for the generic ontology developers. Another experience is that for application-oriented ontology development, available general tools and languages are not very suitable for practitioners. For this reason we decided to develop the discussed graphical formats so that direct visual modelling becomes possible. The associated ontology representations such as RDF and OWL stay ‘under the hood’ of the tool, as they are automatically created from the visual diagrams. This significantly lowers the barriers for practitioners.

We intend to make publicly available extensive case and ontological data on an area of our investigations that we have not discussed here. It concerns the automatic clearance of music rights in the digital value chain for Internet radio stations. This domain contains yet another intriguing mixture of business logic considerations and technical challenges that must be simultaneously solved in a mutually consistent fashion, in a networked situation with many different actors (here, music rights societies, Internet radio stations, artists and producers, in different countries). Like the industrial use cases discussed in this article, this provides another rich challenge and test problem for any Semantic Web and Web Service approach to real-world services.

Acknowledgments. This work was partially supported by the European Commission in the context of the EU-IST Project OBELIX (Ontology-Based ELectronic Integration of compleX products and value chains, No. EU-IST-2001-33144). For models, tools, and data on application examples we are indebted to our colleagues Andrei Z. Morch, Hanne Sæle, Gunnar Salseggen (SINTEF Energy Research and TrønderEnergi, Trondheim, Norway), Hans-Peter Schnurr, Hans Trost, Wolfgang Sperling (Ontoprise, Karlsruhe, Germany), Bert Hazelaar (SENA, Hilversum, Netherlands), Tim Smithers, Jessica Aguado, Carlos Pedrinaci, and Amaia Bernaras (PTSS, San Sebastian, Spain). Much of the e^3-service and e^3-value tool development has been done by Arthur Koks and Stephan Hoekstra (VU Amsterdam). We also thank Terry Payne (University of Southampton, UK) for many in-depth discussions on DAML/OWL-S, and the anonymous reviewers for several useful comments that we have incorporated. The overview given in this paper is admittedly brief and thus high-level. The reported work is based on extensive technical reports concerning the OBELIX ontology development work, industrial case studies, demos, and tools, to be found on *http://obelix.e3value.com*. Several other examples and case studies of our ontology-based business modelling and analysis are available on *http://busmod.e3value.com*. Some of the discussed tools will also be made publicly available as open source software.

References

[1] C. Lovelock. *Services Marketing — People, Technology, Strategy*. Prentice Hall, Englewood Cliffs, NJ, 2001. 4th edition.

[2] C. Grönroos. *Service Management and Marketing: A Customer Relationship Management Approach*. John Wiley & Sons, Chichester, UK, 2000. 2nd edition.

[3] J. Gordijn and J.M. Akkermans. Designing and evaluating e-Business models. *IEEE Intelligent Systems*, 16(4):11–17, 2001. Special issue on Intelligent e-Business.

[4] J. Gordijn and J.M. Akkermans. Value based requirements engineering: Exploring innovative e-commerce ideas. *Requirements Engineering Journal*, 8(2):114–134, 2003. Also available frrom http://link.springer.de/link/service/journals/00766/contents/03/00169/.

[5] S. McIlraith, T. Son, and H. Zeng. Semantic web services. *IEEE Intelligent Systems*, 16(2):46–53, March/April 2001. Special Issue on the Semantic Web. For the current DAML/OWL-S description see http://www.daml.org/services/owl-s/1.0/.

[6] T. Gruber, G. Olsen, and J. Runkel. The configuration design ontologies and the VT elevator domain theory. *International Journal of Human-Computer Studies*, 44:569–598, 1996.

[7] C. Löckenhoff and T. Messer. Configuration. In J. Breuker and W. Van De Velde, editors, *The CommonKADS Library for Expertise Modelling — Reusable Problem Solving Components*, chapter 9. IOS Press, Amsterdam, NL, 1994.

[8] W.N. Borst, J.M. Akkermans, and J.L. Top. Engineering ontologies. *International Journal of Human-Computer Studies*, 46:365–406, 1997.

[9] B. Omelayenko, M. Crubézy, D. Fensel, R. Benjamins, B. Wielinga, E. Motta, M. Musen, and Y. Ding. UPML: Language and tool support for making the semantic web alive. In D. Fensel, J. Hendler, H. Lieberman, and W. Wahlster, editors, *Spinning The Semantic Web*, chapter 5, pages 141–170. The MIT Press, Cambridge, MA, 2003.

[10] A.Th. Schreiber, J.M. Akkermans, A.A. Anjewierden, R. De Hoog, N.R. Shadbolt, W. Van De Velde, and B.J. Wielinga. *Knowledge Engineering and Management*. The MIT Press, Cambridge, MA, 2000.

[11] P. Mika, D. Oberle, A. Gangemi, and M. Sabou. Foundations for service ontologies: Aligning OWL-S to DOLCE. In *Proceedings 13th World Wide Web Conference WWW2004*, pages 563–572, New York, NY, 2004. ACM Press. Also available from http://www2004.org/proceedings/docs/contents.htm. For the DOLCE ontology, see http://wonderweb.semantic.org and http://www.isib.cnr.it.

The Authors

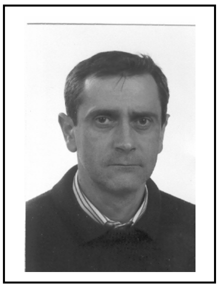
Hans Akkermans is a professor and head of Business Informatics at the Free University Amsterdam, and an international consultant. His research interests are interdisciplinary Information Science, distributed intelligence, and innovation with advanced ICT. He holds a cum laude PhD degree in theoretical physics from the University of Groningen. He has published over 150 scientific papers in information systems, physics, engineering, and e-business. He is chair of the board of the Netherlands Graduate Research School of Information and Knowledge Systems SIKS, scientific director of EnerSearch AB, an international industrial consortium on ICT in energy, and coordinator of this area for the European Commission. He leads VUBIS, the Amsterdam multi-disciplinary research center for Business Information Sciences, a collaboration of computer science, economic and social science research. Contact him at Hans.Akkermans@akmc.nl or at elly@cs.vu.nl (c/o Elly Lammers).

Ziv Baida is a researcher in the Business Informatics group at the Free University Amsterdam. He is developing the OBELIX service ontologies, tools, and some of their industrial use cases. He received an MSc in Computer Science and Software Engineering from the Free University Amsterdam in 2002. Contact him at ziv@cs.vu.nl; www.cs.vu.nl/~ziv.

Jaap Gordijn is assistant professor of e-Business at the Free University Amsterdam. He worked previously at Deloitte and Touche and at CISCO's International Business Solutions Group. His research interests are networked business modeling, value-based requirements engineering and innovative e-business applications, and he developed the e^3*-value* ontology, methodology and tools for this. He received an MBA from Tilburg University and a PhD in Computer Science from the Free University Amsterdam. Contact him at gordijn@cs.vu.nl; www.cs.vu.nl/~gordijn/research.htm .

Nieves Peña was born in Bilbao, Spain in 1973. She received her Bachelor in Physics (Electronic and Automatic) from the Basque Country University in 1996. Nieves joined LABEIN in 1996 with a scholarship, working in the development of Systems in the Internet technologies area. In August of 1999 Nieves joined Xerox at Cambridge to participate in the MOBILEDOC project integrating Internet technology with Mobile clients. Afterwards she joined again LABEIN in 2000 at the Information Society and Regional Development Unit, now working in the development of Intelligent Systems and in the Semantic technologies area. Contact her at Information Society and Regional Development Unit, Labein, Parque Tecnológico de Bizkaia, Edificio 700, 48160, Derio, Bizkaia, Spain; npena@labein.es.

Ander Altuna, born in Bilbao in 1980, is a research fellow in the Information and Knowledge Society Unit at Labein Technological Center. His current research focus is the use of ontologies as knowledge base to solve complex configuration problems. He received his MSc in electrical and computer engineering from the Faculty of Engineering of the University of the Basque Country in 2003. Contact him at Information Society and Regional Development Unit, Labein, Parque Tecnológico de Bizkaia, Edificio 700, 48160, Derio, Bizkaia, Spain; aaltuna@labein.es.



Iñaki Laresgoiti was born in Mexico D.F., Mexico in 1951. He received his Bachelors degree from the School of Engineering of the University of the Basque Country in 1973. He received a Msc. Degree in Mechanical Engineering from the University of California at Berkeley in 1976 and an Msc in Applied Mechanics from the University of Michigan at Ann Arbor in 1986. In 1987 joined the Electronics department of LABEIN and was responsible for the application of Artificial Intelligence techniques in Power Systems. Since 2000 he is responsible for the research activities on Information Technologies for Power systems. During all these years he has participated in several European research programs, being currently responsible for the ScadaOnWeb and OBELIX projects. He is the Spanish national representative at the WG14 of the TC 57 within the IEC. Contact him at Energy Unit, Labein, Parque Tecnológico de Bizkaia, Edificio 700, 48160, Derio, Bizkaia, Spain; lares@labein.es.

Contact addresses:

Hans Akkermans, Ziv Baida, Jaap Gordijn:	**Nieves Peña, Ander Altuna, Iñaki Laresgoiti:**
Free University Amsterdam VUA Faculty of Sciences, Section Business Informatics FEW/BI De Boelelaan 1081a, NL-1081 HV Amsterdam, The Netherlands Phone: +31 20 444 7718, Fax: +31 20 444 7728 Email: Hans.Akkermans@akmc.nl, ziv@cs.vu.nl, gordijn@cs.vu.nl	Labein Parque Tecnológico de Bizkaia, Edificio 700, 48160 Derio, Bizkaia, Spain Phone: +34 94 607 3352 Email: npena@labein.es, aaltuna@labein.es, lares@labein.es