# Chapter 4
# Service System Approaches
## Conceptual Modeling Approaches for Services Science

Roberta Ferrario, Nicola Guarino, Romano Trampus, Ken Laskey, Alan Hartman, and G. R. Gangadharan

**Abstract** Over the last several years, *services science* has emerged as an effective means to understand services and the socio-technical systems in which they are deployed. This systemic view requires a genuinely interdisciplinary approach to the study of services. In this chapter, we review a number of significant approaches to analyze, understand and model service systems, with an emphasis on showing similarities and differences that highlight the many aspects of a rich service ecosystem. The goal of this chapter is to provide developers with an overall perspective on such rich service system models, as a basis for choosing those which mostly fit their own needs.

Roberta Ferrario, Nicola Guarino
ISTC-CNR, Laboratory for Applied Ontology, via alla Cascata 56C, 38123 Trento, Italy,
e-mail: roberta.ferrario@cnr.it, e-mail: nicola.guarino@cnr.it

Romano Trampus
University of Trieste, Piazzale Europa,1 34127 Trieste, Italy, e-mail: trampus@units.it

Ken Laskey
The MITRE Corporation, M/S H305, 7515 Colshire Drive, McLean, VA, 22102-7508, USA,
e-mail: klaskey@mitre.org

The author's affiliation with The MITRE Corporation is provided for identification purposes only, and is not intended to convey or imply MITRE's concurrence with, or support for, the positions, opinions or viewpoints expressed by the author.

Alan Hartman
IBM Research Lab, Haifa University Campus, Mount Carmel, Haifa, 31905, Israel,
e-mail: hartman@research.ibm.il

G. R. Gangadharan
IBM India Research, India, e-mail: gangadharan@in.ibm.com

A. Barros and D. Oberle (eds.), *Handbook of Service Description: USDL and Its Methods*,
DOI 10.1007/978-1-4614-1864-1_4, 

## 4.1 Introduction

A relevant feature of services is that they are never given in isolation. They are typically conceived as being composable one with the other, and they always exhibit their effects in larger contexts. So, a new subject of inquiry has emerged in the recent years: service systems. As often happens, the expression "service system," while being relatively popular, is understood in different ways by the various communities. In some cases, it mainly refers to a set of interconnected services, while in other cases it is used to include other entities besides the service itself, i.e., people, artifacts, resources, the external environment. In these cases, a service system is a complex socio-technical system.

The latter view — service systems as complex socio-technical systems — is strongly advocated by the founders of a new science, services science [10], urging for the need of a radically interdisciplinary approach to model, understand and control in a systemic way all the economic and social aspects behind the notion of service. Of particular value note, the concept of co-creation is the sharing and distribution of labor, investments, expertise, risk, and — most of all — knowledge. In the last few years, the studies dedicated to this new field have multiplied ([20, 21, 26]), involving disciplines as diverse as economics, sociology, computer science, philosophy, psychology, and linguistics.

We shall review in this chapter the main service modeling approaches that borrow this wide services science perspective. After a recap of Steven Alter's view of service systems as work systems, we present Ferrario and Guarino's foundational ontology of services (General Service Model — GSM), which among other things proposes a unifying definition for the general notion of service, and clarifies the difference between services and service systems. We introduce then three different approaches which more or less build on the notion of service system, namely the TEXO Service Ontology, the OASIS SOA Reference Architecture Foundation (SOA RAF), and the IBM Service Design Model (SDM), discussing their differences and similarities among themselves and with respect to the GSM. While there are other examples in the literature (for example, [9]) that provide a representation of service systems, the ones discussed in this chapter emphasize an ontological approach, paying explicit attention to the nature and structure of service systems, the various entities involved with them, and their mutual relationships.

In order to facilitate understanding and comparisons, we will use a recurring example throughout the chapter: car washing. This is indeed a very popular example of service, thoroughly discussed in literature.

## 4.2 Alter's Framework: Service Systems as Work Systems

Before defining service systems, Steven Alter considers first possible independent definitions of services and systems, such as: "Services are acts performed for someone else, including providing resources that someone else will use," and "a system

is a consciously designed combination of things or parts that perform useful work" [6]. Quickly, however, he realizes there are problems with such definitions, and proposes to go *beyond a definition of service*, suggesting to focus on the broader notion of service system: whatever services are,[1] they are produced through service systems. The core of Alter's position is that service systems are *work systems*, where human participants or machines perform work using information, technology, and other resources to realize products. So the emphasis is more on *how* services (and products) are produced and *who* is involved in the production and consumption process, and less on *what* services are. Customers and customers' issues are prominent throughout the analysis of systems.

So, we can conclude that, according to Alter, describing a service amounts to describing the work system where the service is produced. Indeed, for him every work system is a service system [5]. Under this assumption, he presents three interleaved frameworks to describe a service system. The *work system framework* [3] provides a system-oriented view of any system that performs work within or across an organization, described in terms of nine basic conceptual categories. The work system framework puts customers first in the service process, and aims to indicate a path to customer satisfaction. The *service value chain framework* [4] augments the work system framework by introducing further notions that are associated specifically with services. The *work system life cycle model* [20] looks at how work systems (and therefore service systems) change and evolve over time. The three frameworks are the basis for a comprehensive business-oriented analysis, intended to be also used by IT professionals [3].

The *work system framework* is based on four general categories: processes and activities, participants, information, and technologies. Five more specific categories help to fill out the picture: products and services, customers, strategies, environment, and infrastructure.

The *service value chain framework* outlines service-related activities and responsibilities of the main parties involved (service provider and service customer) in the form of *service responsibility tables*. This framework is based on a number of assumptions, among which we find:

1. understanding services requires recognition of activities and responsibilities;
2. services are often co-produced by service providers and customers;
3. the idea of a service is the same regardless of whether services are directed at internal or external customers;
4. customer satisfaction is affected by the complete set of activities, responsibilities, and experiences occurring within the service system, as acquiring, receiving, and benefiting from a particular service;
5. the service is delivered as based on negotiated commitments, under which the service may be requested and delivered repeatedly;

[1] The difference between products and services has not been analyzed by Alter, who adopts a general, business oriented definition for services provided by Vargo and Lusch [29]: a service is "the application of specialized competencies (knowledge and skills) through deeds, processes, and performances for the benefit of another entity or the entity itself."

6. the service value is captured by the leftmost and rightmost portions of the service value chain, and includes all parties experience in the service exploitation.

In order to model the service value chain framework, Alter introduces *service responsibility tables*, a conceptual tool to clarify the service's scope and context, focus attention on activities and responsibilities, identify job roles, bring customer responsibilities into the analysis, and identify service interactions between providers and customers.[2]

The *work system life cycle model (WSLC)* provides a dynamic view of how work systems change over time. It is an iterative model based on the assumption that a service system evolves through a combination of planned and unplanned changes. The framework distinguishes several phases, such as *operation and maintenance*, *initiation*, *development*, and *implementation*.

All the three frameworks above can be deployed to support the analysis, design, and improvement of service systems, helping their participants and stakeholders in different ways, according to the different roles they play. Alter distinguishes five of such roles:

*Role 1. Executives* can use the work system framework to check whether all the relevant business aspects of the service system are properly covered.

*Role 2. Strategists* can use the three frameworks to provide some kind of organized access to all the relevant design variables (e.g., for performance simulation or optimization purposes).

*Role 3. Managers* can use service responsibility tables to understand the essence of main steps of service workflow without requiring detailed modeling tools such as flowcharts or database schemes.

*Role 4. Implementers* of service system changes can exploit the work system life cycle model to understand the effects of changes.

*Role 5. Consultants and IT professionals*, who have to deal with a large number of technical details, can use the three frameworks to communicate effectively with the other roles, mapping technical choices to high level business aspects.

Important characteristics of a service system can be described using a number of service-related design dimensions. These are important in the analysis of customer-centric service systems, and dimensions such as "product-like features" vs. "service-like features" help understanding the different viewpoints of different stakeholders. The analysis should consider dimensions that take into account the customer's needs, the product vs. service balance, the personalization and the coproduction or self-service approach, the technology, the infrastructure and the environment, among others [6].

[2] Along similar lines, there are some works in business processing, as for instance the Linear Responsibility Charts (or RASCI matrixes) that associate to each task relevant members inside the organization that are either responsible/accountable, or must be informed and/or consulted with respect to it.

## 4.3 The General Service Model

At the ISTC-CNR Laboratory for Applied Ontology in Trento, two authors of the present chapter, Roberta Ferrario and Nicola Guarino, have explored the ontological assumptions behind the notion of service (cf. [12, 13, 14]), by developing an approach recently presented as the General Service Model. As this work is still ongoing, we briefly present here its most recent version.

The initial motivation behind this work was to develop an ontology of services suitable to be used in the e-government domain, where the problem of interoperability is particularly crucial, and multiple understandings of the word 'service' co-exist. By looking at the computer science literature, it was immediately evident that most of the available models adopt the *black box view* of services, describing them as transfer functions from an input to an output state, with a strong focus on the external service interface.[3] Under this view, the internal details concerning *how* the service is performed are kept hidden, despite their relevance from the business point of view. Business applications need not only specify what the service does, but also how the service is performed and when the various processes involved in a service occur. Moreover, contracts and service level agreements need to refer to internal and contextual details (i.e., how the service interacts with its environment). In other terms, one needs to be able to look both *inside* and *outside* of the box, i.e., we need to adopt a *glass box view*, where the box is in this case, as Alter suggests, the whole service system.

However, adopting a glass box view to model a service system forces us to face some fundamental questions: what is there inside the box? What's the difference between a service system and a service? And what is a service, after all? The main contribution of Ferrario and Guarino is that a service — as opposed to a good — always develops in time, i.e, it has an essential temporal nature: ontologically speaking, services are complex *events*, while goods are *objects*.

The complex internal structure of a service, as well as its relationship with the broader service system, is depicted in Figure 4.1, which is a revised version of a similar figure presented in [12]. The picture clarifies Alter's idea of the service system life-cycle, presenting it as a complex temporal entity involving three main components, that are necessarily always present: the Service Commitment, the Service Process, and the Service Value Exchange. In terms of the DOLCE [22] ontology of temporal entities, the Service Commitment is a *state*, holding as long as the provider is willing to offer the service; the other components are dynamic *processes*, involving a number of different activities. An ontological dependence relation holds between the service commitment and the service process, in the sense that the latter cannot exist without the former. The interplay between service commitment, service process and service as a whole is described by the following informal definitions, adapted from [12]:

A *service commitment* is an agent's explicit and enduring commitment to guarantee the execution of some type of *core actions*, on the occurrence of a certain

[3] For a detailed description of these approaches see Chapters 5, 6, and 7 of this book.

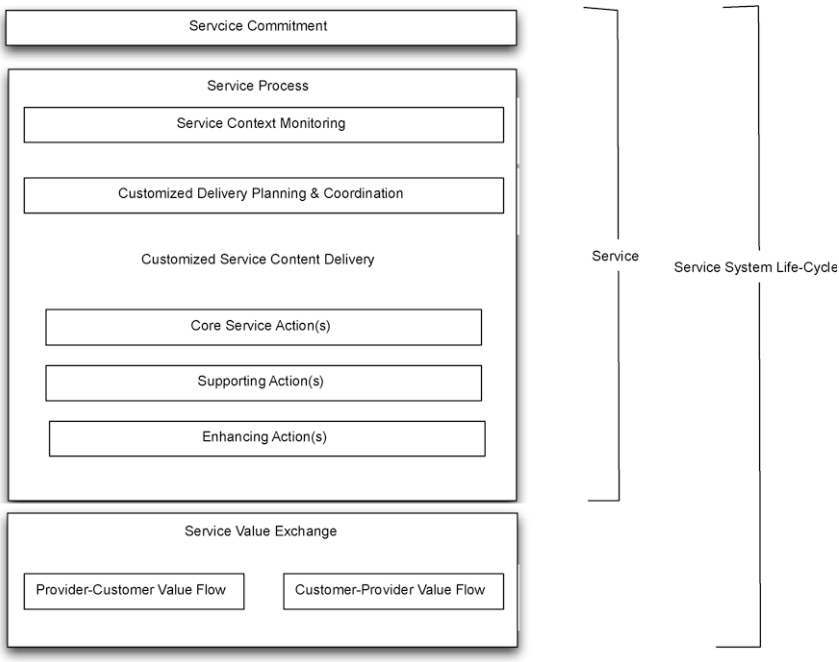


Fig. 4.1: Service and Service system.

triggering event, in the interest of another agent and upon prior agreement, according to a certain specification (*service description*) which constrains the way service actions will be performed. In most cases, two kinds of service commitment need to be distinguished: a *generic* commitment towards potential customers, whose service description is intended to facilitate service discovery, and a *specific* commitment towards a particular customer, where the service description takes the form of a binding *contract*, resulting from a negotiation process.

A *service process* is the actual implementation of a service commitment, consisting of a number of interdependent actions including those necessary to monitor the triggering events, the core actions mentioned in the commitment, and any further actions aimed at supporting or complementing the successful execution of such core actions. What actually happens in the service process is constrained by the *service description*, which defines and constrains the type of actions that must and/or can be executed in the service process.

A *service* is a complex temporal entity (a *complex event*)[4] consisting of a service commitment and the corresponding process.

[4] Generic temporal entities are called *perdurants* in DOLCE, and include *events, states,* and *processes*. However, sometimes "event" is also used as synonymous of perdurant, leaving the context to disambiguate between the generic use and the specific use of the term.

The *service system* is defined as the sum of all the objects anyhow involved in a service (through a *participation* relationship). In other words, while a service is a complex *event*, a service system is a complex *object*, consisting of all the objects somehow participating to any of the sub-events, processes or states constituting the service: typically, a service system includes the provider, the customers, the resources used to produce the service, and so on.[5]

The *service system life-cycle* is a complex temporal entity corresponding to the dynamics of a service system. So the difference between a service system and its life-cycle is like the one existing between a person and its life.

The *service value exchange* is a crucial part of the service system life-cycle. It is a complex process involving two symmetric value flows: the *provider-customer* value flow accounts for provider's costs in implementing the service process, and the corresponding benefits on the side of customers; the *customer-provider* value flow accounts for the costs customers incur in order to receive the service, and the corresponding benefits on the side of providers. Such value flows are also events, and, altogether, the service value exchange is also ontologically dependent on the commitment. Note that the service value exchange is not part of the service itself, since it involves activities occurring at the customer's side: it is rather part of the service system life-cycle.

Relating Figure 4.1 to the car washing example, the service commitment starts when the car wash owner goes to the chamber of commerce to attend all the bureaucratic practices that are necessary to start the commercial activity. Among these practices, there will be some signed official declaration in which the main features of the service are described. In this description, the car wash owner commits to certain business intentions (to be integrated with the content of the ads he or she publicly posts).

The service process is composed of various events, and sub-processes, including the events that trigger the service, e.g., a request by the customer who brings his or her car to the car wash. After the initiating event, we find the customized delivery planning and coordination; here we can imagine that the car wash offers a range of different possible implementations of the service, such as washing only the outside of the car, cleaning the inside, using particular products, such as specific shampoos or waxes etc. In the customized delivery planning phase, the customer and the car wash personnel agree to all these details.

With respect to the service delivery, the core action is washing the car; singling out supporting actions is a bit harder in the example, as there are many actions that are necessarily preparatory to the service but are not explicitly mentioned as constituting the service. Examining possible examples, we could say that the activity of removing loose items from the car in order to be able to clean the inside could be considered a supporting action, as well as buying the cleaning products. Enhancing actions are actions meant to augment the value of the service. Here we could think about an additional service that is connected but not strictly included in the service,

[5] To stress that the notion of service system really includes the context it is embedded in, the expression *service ecosystem* might be appropriate (see also Section 4.5). We shall stick however to service system in the following.

such as replacing air filters or, alternatively, we could think about a luxury service in which someone picks up the car at the customer's location, takes it to the car wash, washes it and then brings it back. The picking up and bringing back would be in this case enhancing actions.

Finally, all the activities connected to the flow of value, both from the customer to the provider (such as payment, loss of customer's time etc.) and from provider to customer (such as time, labor and resources implied in the service production) constitute the service value exchange. In this case, these are the transfer of money and the time spent to drive to the car wash, wait for the car to be washed and drive back for what concerns the customer-provider flow and the time, labor and materials used in washing the car for the provider-customer flow.

A UML diagram of the General Service Model is shown in Figure 4.2.[6] There are three main classes: *Service system, Service system life-cycle,* and *Service system description.* The elements of these classes have a different ontological nature (not shown in the figure): service systems and their parts are *objects*, service system life-cycles and their parts are *events* (generic temporal entities), service system descriptions and their parts are *informational objects.* We adopt specific relations to account for the way an object participates to an event, called "thematic relations" in linguistics [15, 13]. Typical thematic relations are:

| | |
|---|---|
| *Agent* | pointing to the entity that plays an active role in the event |
| *Theme/Patient* | pointing to what undergoes the event; the patient changes its state, the theme does not |
| *Recipient/Beneficiary* | pointing to what receives the effects of the event |
| *Instrument* | pointing to what is used to perform the event |

This choice allows the authors to propose a formal version of Alter's responsibility tables [4, 13] where rows represent specific service sub-events, and columns describe the specific structure of such events, in terms of thematic relations. Starting from the center of Figure 4.2, we see that a service system life-cycle has two mandatory parts, the service itself and the service value co-production process. In turn, a service has two essential parts: a commitment, and a process that realizes it. The commitment's theme is a *service description* that says what the service is supposed to do. In particular, such description constraints the *core actions* to be performed during the service process. The service description is part of a more general *service system description,* which accounts for the *service value co-production* between the customer and the provider, describing (among other things) the price policy and the legal constraints which limit or regulate the service's range of applicability. Participants to the service system life-cycle are all the parts of the service system, including the service system context (for instance the surrounding economic, legal, and social

[6] Note that Figure 4.2 provides a UML representation but additional work is ongoing to develop models that exhibit more semantic and logical expressiveness. An example would be a GSM model using a first order logic formalism and possibly translatable at least partially to OWL.

systems) and the various actors, such as the service provider, service customer, service producer, and service consumer.[7]

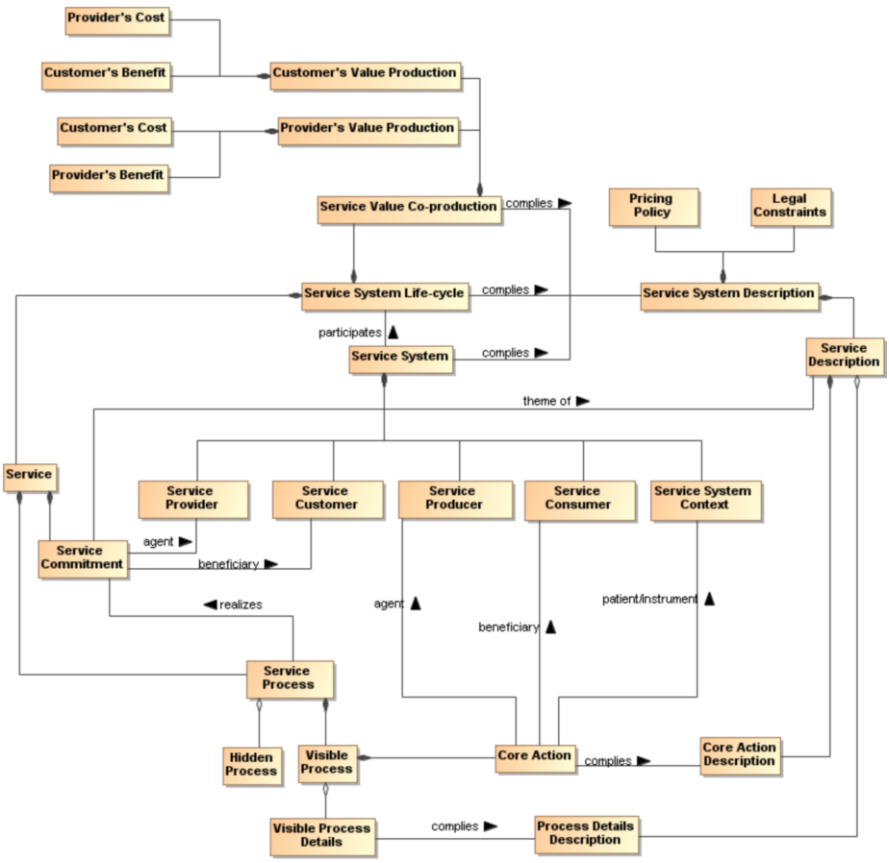


Fig. 4.2: The General Service Model (revised version from [14]).

The picture explicitly shows the thematic relations characterizing the structure of *service commitment*. The commitment's agent is the *service provider*, while the beneficiary is the *service customer*. In car wash example, the service provider is the car wash owner, and the beneficiary is a generic (possible) customer, while the chamber of commerce is, in a sense, acting on behalf of these possible customers. The service description is possibly contained in a document that is stored at the chamber of commerce and includes an explanation of the service. What is written there is what the owner of the car wash is promising to deliver and is what can eventually be handled by the customers in case what was promised is not then realized.

[7] We implicitly assume that participation is distributive with respect to parthood, so if the service system participates to the service system lifecycle all its parts do the same.

In very simple terms, if the description only says that the service merely consists in washing cars, the customer can protest just in case his or her car is dirty after the execution of the service; but if the description specifies, for instance, that only ecological products will be used and the customer finds out that other products are used, he or she can claim that the commitment has not been honored. The service commitment has also a duration and location, which are the period and place where the owner guarantees that the service will be available. For the duration, usually it starts the first moment in which the car wash is open and lasts until the activity is ceased, i.e., the car wash will finally be closed. According to the modeling choices, one could decide to restrict the availability of the service to the opening hours of the car wash, but, as usual, this depends on what is written in the service description. In this example the commitment location is not particularly meaningful as it is identified with the car wash location, but there are more interesting examples, such as fire extinguishing, where the area in which the service is active must necessarily be specified beforehand.

The *service process* realizes the commitment, i.e., it is the execution of the actions described in the service description, according to the constraints there stated and is composed of two parts: the visible process (mandatory) and the hidden process (optional); these two can be roughly identified with the front end and the back end processes. The visible process has some mandatory core actions (those that in a sense define the service for what it is, i.e., the core action is what the service fundamentally does) and some optional visible process details.[8] These are usually enhancing or supporting actions, that may equally be visible or invisible. Also, the core action has to comply with the core action description, while the visible process details have to comply with the process details description. The core action description and process details description are both part of the service description (though only the former is necessary). The hidden process does not have a correspondent in the description because it contains all those actions that are performed but not constrained by the description, i.e., the provider is free to perform such actions as he or she wishes since they are not ruled by the commitment.

Note that the core action's agent and beneficiary are the *service producer* and *service consumer*, respectively, who may or may not coincide with the provider and the customer, depending on the kind of service. In the car washing example, the core action is the washing itself, whose agent is the worker who actually washes the car; this may or may not coincide with the owner; the consumer is the guy who goes to the car wash for having the car washed (also this may or may not be the owner of the car: in the former case he or she is also the customer, in the latter case he or she is not, think about someone who goes washing the car that a friend has lent him or her for a period who, though being the customer, is not the final beneficiary, i.e., the consumer).

The duration of the core action coincides with the time that is taken to actually wash the car and the location is again the car wash itself. The instruments here are the water system, the sponges, the brushes, shampoo, wax etc.

[8] Here "visible" and "hidden" refer to the customer's perspective.

Finally, the upper part of Figure 4.2 describes the service value co-production process, which is constituted of two symmetric "flows" of value: from provider to customer, and from customer to provider. What happens is that there is no real *flow*, since increase or decrease of value are subjective events resulting from different evaluations (from the provider's or the customer's side) of the same objective phenomena. Consider again the car washing. While the physical action is performed, there is in parallel a cost event on the side of the provider, while there is a benefit event on the side of the customer, starting from the time the washing is completed, and lasting for a while. Symmetrically, there is a cost event (a sacrifice) on the side of the customer at the payment time, corresponding to a benefit on the side of the provider. Modeling sacrifices and benefits as temporal entities having a non instantaneous duration allows us to account for different kinds of service, depending on how value is produced at different times. So we can say that, for instance, paying for having your car washed is a bad deal if the roads are muddy, so that you can enjoy your car clean only for a short time.

## 4.4 The TEXO Service Ontology

The TEXO Service Ontology [24] has been developed in the framework of the THESEUS/TEXO project [28], and has taken inspiration in its latest phases from the ongoing work on the General Service Model, as well as from several adjacent ontologies for capturing information about service innovation, pricing, licenses, rating, etc. In order to understand the rationale of the TEXO Service Ontology, one has to have a look at the *service lifecycle* (Figure 4.3), which loops between the innovation, offering, matchmaking, usage, and feedback phases.[9] The *innovation phase* allows for new business models and new consumption and development paradigms. In the *offering phase*, services are supplied to the market. Once a service is designed and developed, it needs to be turned into a commercial offer. In order to create a commercial offer out of a service implementation, several parameters need to be described and published on a service marketplace. The *matchmaking phase* denotes the process of matching a service provider's service offer to a service consumer's service need, i.e., the central application of service description. The *usage phase* in the service lifecycle essentially comprises the delivery of services. Feedback for future iterations of the service lifecycle is channeled back to the service provider during the *feedback phase*. This includes the analysis of the feedback from the applications monitoring.

It is the goal of the TEXO Service Ontology to provide a general scheme for master and transactional data across all phases of the service lifecycle. *Master data* comprise data which seldomly change over time, e.g., a price plan or service licenses. In contrast, *transactional data* grow over time, e.g., data about the service value

[9] This is analogous to the notion of service system lifecycle discussed before, with an emphasis on the fact that the way a service is conceived (and hence the service commitment) can evolve in time, on the basis of customers' feedback.

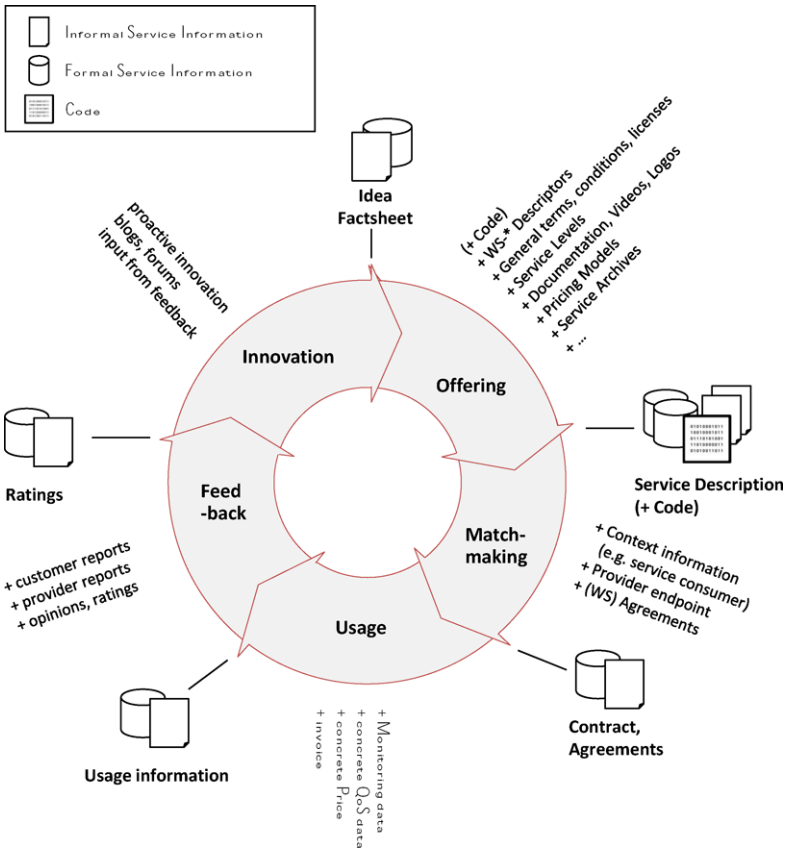


Fig. 4.3: Different phases in the lifecycle generate different kinds of service information.

exchange are generated whenever a service is consumed, and multiple contract data are generated (as transactional data) for one service master data description. Something similar happens in the usage and rating phases where specific monitoring data, concrete service levels, or ratings are generated.[10]

The resulting Service Ontology is depicted by a pyramid in Figure 4.4, which is a metaphor for the number of classes and relations that increases from top to bottom. The Service Ontology is specified in OWL-DL and consists of several modules. Each module basically coincides with an OWL file that imports other OWL files. The modules are depicted as parts of the pyramid. The ontology modules can be divided into four layers according to the requirements:

[10] Transactional data correspond to the *customized service content delivery* phase in the GSM, shown in Figure 4.1, which however, for the sake of simplicity, has not been considered in Figure 4.2.

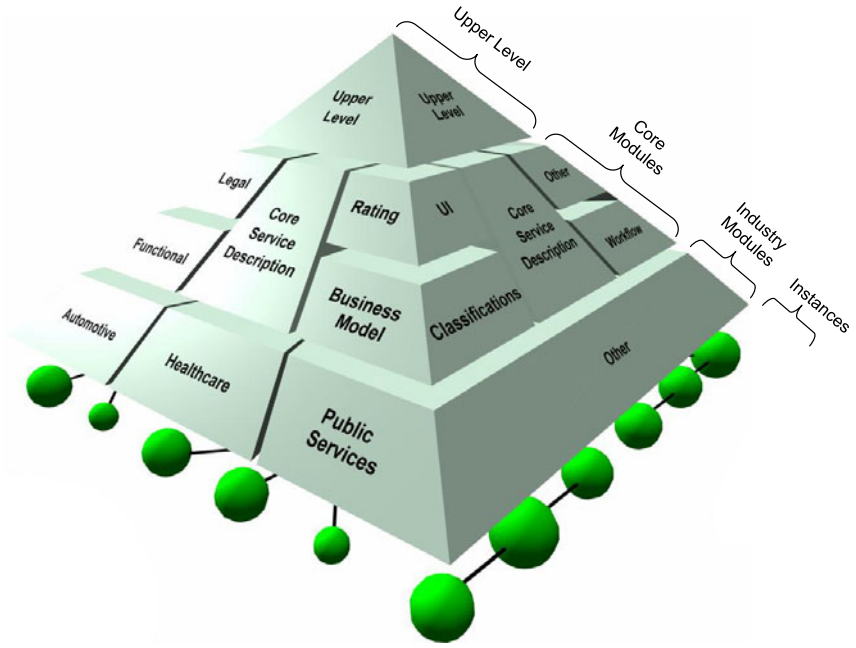


Fig. 4.4: The Service Ontology as a pyramid with increasing amount of classes and relations from top to bottom.

At the top layer, the *upper level* module consists of a concise foundational ontology providing us with a generic set of classes and relations as well as ontology design patterns. More specifically, the DOLCE foundational ontology [16] is applied, which serves the following purposes: (a) DOLCE can be used as a modeling starting point because it provides a basic set of generic classes and relations valid in any domain. Using a foundational ontology as a modeling basis means relating core classes and relations to some proposed invariant categories of human cognition. This prompts the ontology engineer to sharpen his/her notions with respect to the distinctions made in the foundational ontology. What is typically gained is an increased understanding of one's own ontology as well as a cleaner design. (b) DOLCE can also help defining general ontology design patterns as best practices for reoccurring modeling needs.

At the middle layer, a set of *core modules* is built around the Core Service Description module, which captures information common to every service (e.g., info on service provider, quality of service, etc.). Note that the Core Service Description Module does essentially coincide with the General Service Model described above.[11] In addition, different aspects of a service description (legal, busi-

[11] For this reason in this section we won't make use of the car wash example, as it would be essentially represented in the same way as in the GSM.

ness model, technical, rating, UI, etc.) are placed in separate modules and linked to the classes belonging to the Core Service Description module. All modules in this middle layer are aligned under the common roof of the DOLCE foundational ontology. So far, several core modules have been designed to a mature state and published in diverse literature. The enumeration below provides an overview and pointers to the corresponding resources.

- Core Service Description Module [16]
- Idea Module [25]
- Pricing Module [18, 17]
- Legal Module [8]
- Rating Module [23, Section 9]
- Classification Module [23, Section 10]
- Documentation Module [23, Section 11]

At the third layer, *industry modules* (e.g., automotive, healthcare, or public services modules) can be modeled by exploiting the aforementioned ontology modules. The core knowledge specified in the Core Service Description module and adjacent aspect-related core modules discussed above can be specialized for specific industries. Industries can define their own hierarchies of service categories. It is expected that the modules will be populated at run-time by industry consortia or the like.

Finally, *instances* of the classes and relations (depicted as a mesh below the pyramid) can potentially be distributed across the Web according to the principles of Linked Data.

## 4.5 The OASIS SOA Reference Architecture Foundation (SOA-RAF)

The OASIS SOA Reference Architecture Foundation (SOA-RAF) [2] is an abstract realization of the service-oriented architecture (SOA), focusing on the elements and their relationships needed to enable SOA-based systems to be used, realized and owned, while avoiding reliance on specific concrete technologies. By use, the SOA-RAF captures what it is meant to participate in a space in which stakeholders (human and non-human), processes, and machines act together to deliver the effects of business functionality through services. The space with its stakeholders and the environment (or context) within which they all operate taken together forms the *SOA ecosystem*. This is consistent with Alter's idea of the service system. The SOA ecosystem assumes service implementations utilize capabilities to produce specific (real world) effects that fulfill business needs.

In our car washing example, the capability is the collection of equipment and car washing knowledge to produce the real world effect of a washed car. The service is the access to this capability to wash the car of a specific customer.

From a software perspective, the emphasis is often on the implementation of such business functionality such that it is accessible through a well-defined interface; this

is necessary but by no means sufficient. Our car washing example could proceed without an explicit mention of software (or, in general, automated) interactions. However, the customer may have made use of a browser-based interface to schedule a time for the car wash and the custom options desired. Also, payment may have been arranged through an electronic funds transfer (EFT). The capability of a bank to support EFT and the car wash's ability to access this capability fall in line with the previously discussed concept of supporting or complementary actions.

Both those using the services, and the capabilities themselves, may be distributed across ownership domains, with different policies and conditions of use in force. The role of a service in the SOA context is to enable effective business solutions in a distributed environment. SOA is thus a paradigm that guides the identification, design, implementation (i.e., organization), and utilization of such services.

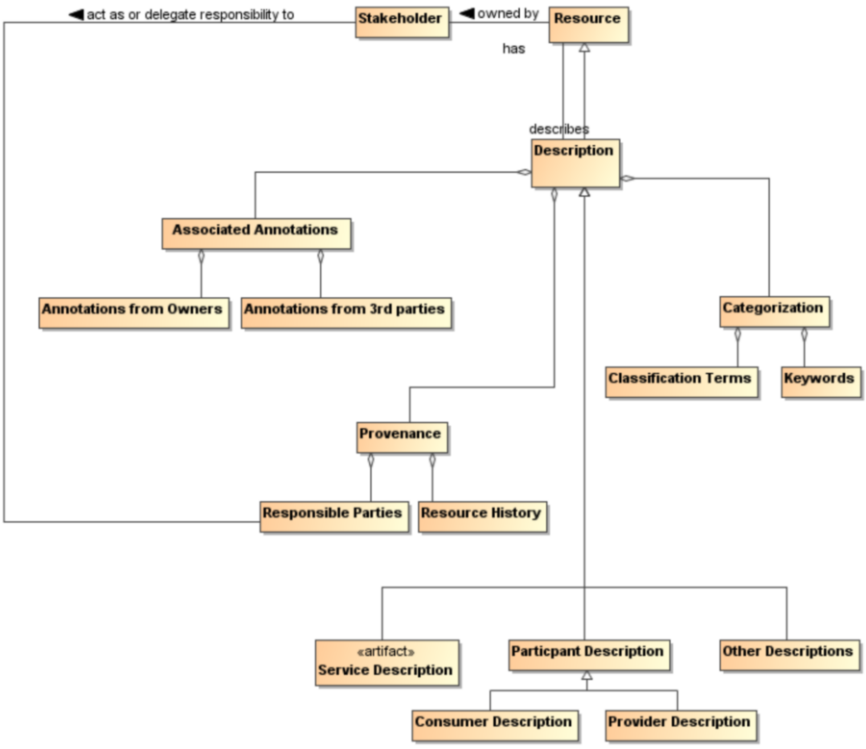


Fig. 4.5: General Description.

The SOA-RAF also discusses the realization and ownership issues involved in the SOA ecosystem. Realization relies heavily on service description, and this will be explored in detail below. Realization also requires sufficient visibility to establish willingness and communications among the participants, effective interactions, and

support for policy and contract statement and enforcement. In the realm of effective ownership, elements of governance, security, management, and testing are explored. These aspects of realization and ownership are consistent with the ontological underpinnings described in the General Services Model.

Much of the service realization and aspects of ownership rely on an accurate and sufficiently complete service description. As discussed in the SOA-RAF, a service description is an artifact, usually document-based, that defines or references the information needed to use, deploy, manage and otherwise control a service. This includes not only the information and behavior models associated with a service to define the service interface but also includes information needed to decide whether the service is appropriate for the current needs of the service consumer. Thus, the service description will also include information such as service reachability, service functionality, and the policies associated with a service.

Interactions within a SOA ecosystem rely on many resources, and the SOA-RAF introduces a general *Description* class in Figure 4.5 to represent a number of description properties that are expected to be common among all specialized descriptions supporting a service-oriented architecture. A registry often contains a subset of the description instance, where the chosen subset is identified as that which facilitates mediated discovery. Additional information contained in a more complete description may be needed to initiate and continue interaction.

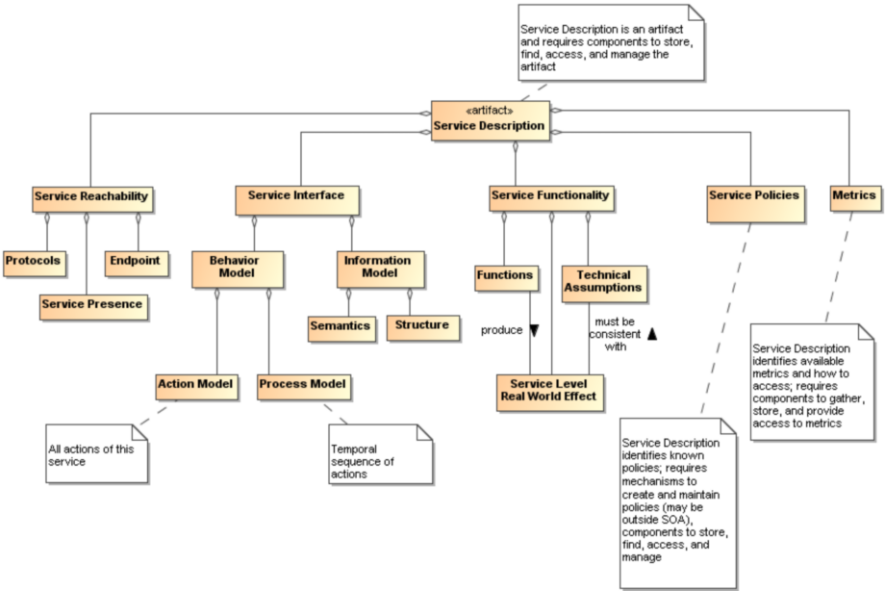


Fig. 4.6: Service Description.

The major description properties for the Service Description subclass follow directly from the areas discussed in the OASIS SOA Reference Model (SOA-RM)

[19] and are shown in Figure 4.6. In particular, the service description conveys the functionality of the service (including the real world effects that are realized through interaction with the service), the conditions of use defined through policies, the operational characteristics captured through metrics, the particulars of the service interface as defined by the service behavior and information models, and the endpoints and corresponding protocols through which message exchange with a present service is accomplished. In addition, provenance, characterization, and identity information and the ability to provide annotations are inherited from the general description.

If we assume we have awareness, i.e., access to relevant descriptions, the service participants must still establish willingness and presence to ensure full visibility as defined in [2] and to interact with the service. Service description provides necessary information for many aspects of preparing for and carrying through with interaction. Recall the fundamental definition of a SOA service in the SOA-RM is a mechanism to access an underlying capability; the service description describes this mechanism and its use. It lays the groundwork for what can occur, whereas service interaction defines the specifics through which occurrences are realized.

Figure 4.7 combines the detailed models for each descriptive element in Figure 4.6 to concisely relate *Action* and the relevant components of *Service Description*. The purpose of Figure 4.7 is to demonstrate that the components of service description go beyond arbitrary documentation and form the critical set of information needed to define the what and how of *Action*. In Figure 4.7, the leaf nodes from Figure 4.6 are shown in differently colored boxes.

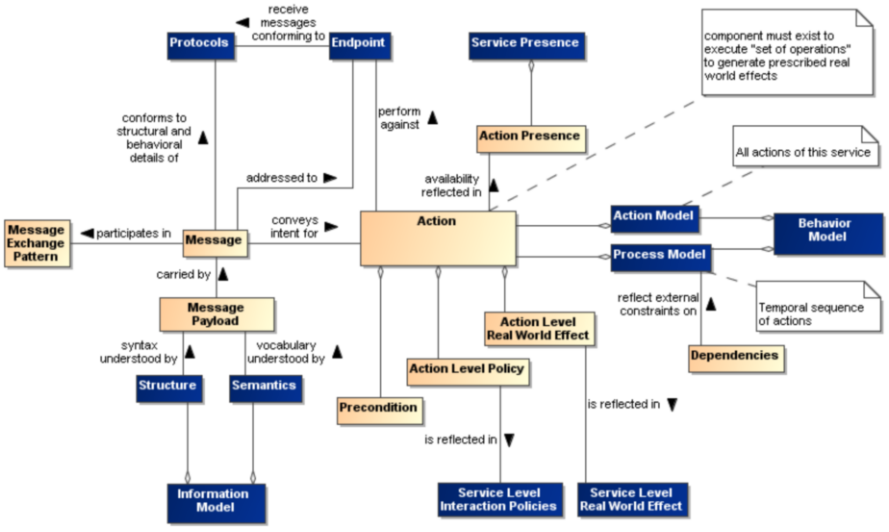


Fig. 4.7: Relationship between Action and Service Description Components.

An adequate service description *must* provide a consumer with information needed to determine if the service policies and the (business) functions and service-level real world effects are of interest and there is nothing in the technical assumptions that precludes use of the service.

Note at this level, the business functions are not concerned with the Action or Process Models. These models are detailed separately. The Actions in the Action Model and their temporal dependence as captured in the Process Model only apply to externally facing actions with which a service consumer would interact. Internal processes are only exposed to the extent they are reflected in policies and other conditions of use or metrics and other measures of operational characteristics. This is the counterpart to the GSM "glass box."

The service description is not intended to be isolated documentation but rather an integral part of service use. Changes in service description should immediately be made known to consumers and potential consumers.

The description of service description indicates numerous architectural implications on the SOA ecosystem:

1. Description will change over time and its contents will reflect changing needs and context. This requires the existence of:
    a. mechanisms to support the storage, referencing, and access to normative definitions of one or more versioning schemes that may be applied to identify different aggregations of descriptive information, where the different schemes may be versions of a versioning scheme itself;.
    b. configuration management mechanisms to capture the contents of each aggregation and apply a unique identifier in a manner consistent with an identified versioning scheme;
    c. one or more mechanisms to support the storage, referencing, and access to conversion relationships between versioning schemes, and the mechanisms to carry out such conversions.
2. Description makes use of defined semantics, where the semantics may be used for categorization or providing other property and value information for description classes. This requires the existence of:
    a. semantic models that provide normative descriptions of the utilized terms, where the models may range from a simple dictionary of terms to an ontology showing complex relationships and capable of supporting enhanced reasoning;
    b. mechanisms to support the storage, referencing, and access to these semantic models;
    c. configuration management mechanisms to capture the normative description of each semantic model and to apply a unique identifier in a manner consistent with an identified versioning scheme;
    d. one or more mechanisms to support the storage, referencing, and access to conversion relationships between semantic models, and the mechanisms to carry out such conversions.

3. Descriptions include reference to policies defining conditions of use. This requires the existence of:
    a. descriptions to enable the policy modules to be visible, where the description includes a unique identifier for the policy and a sufficient, and preferably a machine processable, representation of the meaning of terms used to describe the policy, its functions, and its effects;
    b. one or more discovery mechanisms that enable searching for policies that best meet the search criteria specified by the service participant; where the discovery mechanism will have access to the individual policy descriptions, possibly through some repository mechanism;
    c. accessible storage of policies and policy descriptions, so service participants can access, examine, and use the policies as defined.
4. Descriptions include references to metrics which describe the operational characteristics of the subjects being described. This requires the existence of (as partially enumerated under governance):
    a. the infrastructure monitoring and reporting information on SOA resources;
    b. possible interface requirements to make accessible metrics information generated or most easily accessed by the service itself;
    c. mechanisms to catalog and enable discovery of which metrics are available for a described resource and information on how these metrics can be accessed;
    d. mechanisms to catalog and enable discovery of compliance records associated with policies and contracts that are based on these metrics.
5. Descriptions of the interactions are important for enabling auditability and repeatability, thereby establishing a context for results and support for understanding observed change in performance or results. This requires the existence of:
    a. one or more mechanisms to capture, describe, store, discover, and retrieve interaction logs, execution contexts, and the combined interaction descriptions;
    b. one or more mechanisms for attaching to any results the means to identify and retrieve the interaction description under which the results were generated.
6. Descriptions may capture very focused information subsets or can be an aggregate of numerous component descriptions. Service description is an example of a likely aggregate for which manual maintenance of all aspects would not be feasible. This requires the existence of:
    a. tools to facilitate identifying description elements that are to be aggregated to assemble the composite description;
    b. tools to facilitate identifying the sources of information to associate with the description elements;

c. tools to collect the identified description elements and their associated sources into a standard, referenceable format that can support general access and understanding;
d. tools to automatically update the composite description as the component sources change, and to consistently apply versioning schemes to identify the new description contents and the type and significance of change that occurred.

7. Descriptions provide up-to-date information on what a resource is, the conditions for interacting with the resource, and the results of such interactions. As such, the description is the source of vital information in establishing willingness to interact with a resource, reachability to make interaction possible, and compliance with relevant conditions of use. This requires the existence of:

   a. one or more discovery mechanisms that enable searching for described resources that best meet the criteria specified by a service participant, where the discovery mechanism will have access to individual descriptions, possibly through some repository mechanism;
   b. tools to appropriately track users of the descriptions and notify them when a new version of the description is available.

## 4.6 The IBM Research Service Design Model

The Service Design model described in [11] was conceived as a meta-model for a service in the widest sense of the word. There was an explicit attempt to go beyond the world of Web services and include services with a large element of human involvement in their delivery and consumption. The examples that motivated this original meta-model came from the world of IT services — outsourced support of IT infrastructure, help desks, call centers, and the like. Subsequently, an attempt was made to incorporate the main elements of the formal model for service delivery described in [7], and to widen the context beyond IT Services to include other domains, including Public Services [27].

The initial model was intended primarily for recording the design parameters of a service without explicitly stating their values at the design phase. The design meta-model has been then extended to a solution meta-model where the parameter values were instantiated to create a specific service instance. This approach has been developed further with the incorporation of feature modeling techniques to describe the design model as a service product line using concepts from the world of software product lines [30]. The benefits of feature modeling are mainly in increasing the accessibility of modeling to non-technical users of the system. The emphasis in the research work has been on creating a well-defined formal representation using UML and BPMN notations which are amenable to analysis and automated transformation into service implementation and simulation artifacts. A further focus is on

the facilitation of a participatory design methodology,[12] where all stakeholders provide input and feedback on the services being designed for them. A key challenge has always been to hide the complexity and formality from service domain experts, service consumers, and other non-technical stakeholders.

The tooling which is being developed to accompany the design model exposes simple form filling interfaces to gather the data necessary to populate the model, together with a logical organization of the forms which parallels the workflow. The artifacts produced by the tooling are presented to the stakeholders in the service through a deliberation platform to facilitate participatory design.

The Service Design Model consists of the following elements (cf. Figure 4.8):

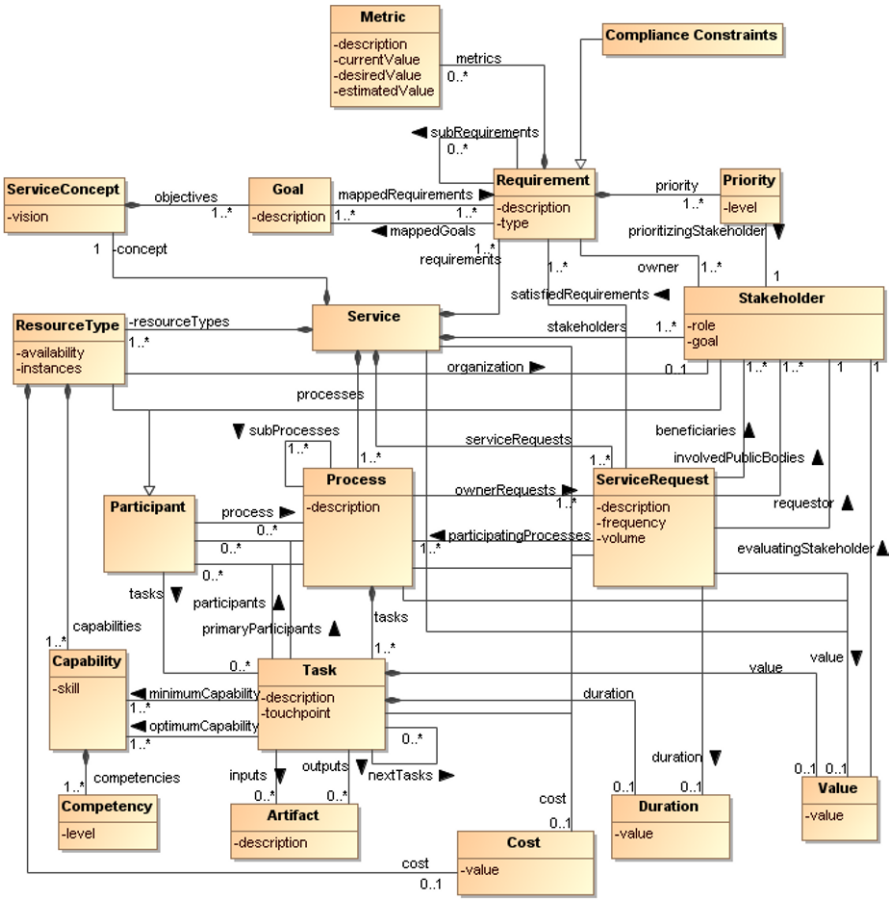


Fig. 4.8: Service Design Model.

[12] There is an extensive literature on participatory design: see [1] for a quick summary.

Service Instances and Service Concepts

A specific service is always an instance of a well-defined ServiceConcept which specifies the vision and the goals of the service. Each goal is mapped to one or more Requirements. In the case of the car wash service, the concept could be one of a franchised car washing company which provides machinery, raw materials (soap, brushes etc.) and premises to franchise holders who pay off the initial investment to the franchising company over a three year period. A service instance in this case is a particular franchise with employees, and local adaptation to the environment and customers at the franchise location.

Service Requirements

A Service has a set of Requirements that are gathered from all the Stakeholders. Each Requirement can be split into sub-requirements and may also specify Metrics to validate if the requirement is met. Stakeholders can prioritize these requirements based on their importance. A Service also abides by certain compliance constraints including legal rules and policies. These constraints which inherit all the properties of a Requirement are high priority requirements that must be met. Typical requirements for a car wash are safety constraints for the workers, income requirements for the franchise company and franchise owner, speed and thoroughness of the cleaning service for the customer.

Service Stakeholders

A Service has a list of Stakeholders with different roles such as the owner (the entity in charge of the service), the beneficiary (the entity that is benefited from the service result), and the service provider (the entity responsible for provisioning the service).[13] The distinction between owner and provider is mainly relevant to the domain of outsourced services where the provider delivers the service under a contract to the owner. In many cases, however, the owner and provider are the same business entity. In the car wash, key stakeholders are the franchise company, the franchise owner, the customer, and the employees. A Stakeholder should own one or more Requirements, and each Requirement must have at least one Stakeholder as its owner.

Service Request

A Service occurs as a response to ServiceRequests which are triggered by a Stakeholder (requestor) who may or may not be the beneficiary of the results.

[13] Owner and provider as defined here correspond respectively to provider and producer in the GSM.

A ServiceRequest invokes a Process — which may invoke other Processes during its execution. Each ServiceRequest has associated parameters such as cost and duration of execution and value derived by the beneficiary. These parameters are aggregated from the sub processes and tasks performed while the request is being handled by the service delivery system. The car wash service request is initiated by the customer who arrives at the franchise.

Service Process

A Service contains many Processes that capture the internal service workflow. A Process has a sequence of Tasks. Each Task is associated with two sets of Capabilities, at different Competency levels. The minimum Capability is the minimal set of skills needed to complete the Task. The optimum Capability of a Task is the set of Capabilities needed to complete the Task with best possible performance. The Participants of a Task can be Stakeholders (e.g., for a customer submitting an application) or Resources (e.g., for front desk employee validating the application, or a computer system needed to complete the Task). A Task provides different Values for different Stakeholders. As a result the containing Process, ServiceRequest and hence Service have Values associated with them by aggregating the Values provided by the constituent Tasks. The service designer uses this Value in order to provide different design alternatives and hence achieve a balance between the Values perceived by different Stakeholders. The car wash process consists of payment (providing value to the owner and operator), cleaning outside and inside the car, drying the car body, and cleaning the windows (each of which provides value to the customer). The capabilities of the employees may include a specialist cashier, cleaning staff, and a sales person who approaches the customer at the end of the process to get feedback and sell a subscription to a set of car washes. In some cases, the cashier may also have the capability to perform some subset of the cleaning work.

Resource Type

A Service has a set of ResourceTypes representing the resource units (human/IT) involved in the service delivery. These ResourceTypes have instances (the actual persons or machines) and availabilities associated with them. They also have a skill set in the form of Capabilities at different Competency level. Aside from the human resources involved in the car wash, there is also car washing machinery, a cash register, and a subscription database sitting on a computing resource.

## 4.7 Discussion

The presented approaches each reflect differences in viewpoint. Avoiding any exhaustive comparison, we shall focus here on the key features more or less present in all approaches, and on the way the different solutions proposed complement each other by focusing on different perspectives, giving altogether a reasonably complete picture of the most important aspects of a service system.

### *4.7.1 Service Definition*

In Alter's works we find a definition of service system directly imported from Vargos and Lusch [5, 29] "the application of specialized competencies (knowledge and skills) through deeds, processes, and performances for the benefit of another entity or the entity itself."

The General Service Model (GSM) adds details to this informal definition by showing the relationship between service and service system, which is somewhat blurred in Alter's approach. The GSM defines a service as "a complex event composed of different sub-events," from the perspective of the whole service life-cycle. It also analyzes the notion of service value chain, and considers all the elements that may contribute to "value co-creation." The service *system* is instead viewed as a composite object that includes all the entities involved in the execution of interrelated services, such as the agents that participate in these service events, as well as the artifacts and resources that are used and possibly transformed by such services and that can be of different nature (physical, informational etc.). So, unlike Alter, the complex processes and actions at the core of the service notion are not considered as parts of the service *system*, but as parts of the *service* itself. According to the GSM, the difference between a service system and a service reflects the ontological difference between objects and events (endurants and perdurants in DOLCE): the service system is the complex object whose global behavior (i.e., the *service system lifecycle*) "produces" the service, so to speak: to have a service, you need a service system that produces it. To account for the dual nature of services, which always involves a value exchange interaction with the outer environment, Figure 4.1 shows how the *service system lifecycle* articulates into two main components: the service itself, and the service value exchange.

The TEXO Service Ontology imports the definitions of concepts constituting the core service description module directly from the General Service Model, but adopts a broader notion of service (system) lifecycle, which accounts for changes in the service commitment, as a consequence of interaction with the external context.

SOA-RAF discusses the inherent differences between services providing a business function and services considered as a software artifact in a SOA ecosystem. An important role in SOA-RAF is played by the notion of service description, defined as "an artifact, usually document-based, that defines or references the information needed to use, deploy, manage and otherwise control a service." This definition is

close to the GSM definition, according to which service descriptions are informational artifacts whose contents are the constraints on the way the service is supposed to be delivered. However, SOA-RAF does not provide a crisp definition of what a service is, although the SOA-RM definition concerning the nature of services as "mechanisms to access an underlying capability" fits with the GSM idea of detaching the service commitment from the actual execution of the core service action. As a matter of fact, it seems that SOA-RAF finds a crisper definition more of a distraction than a clarification.

The IBM-SDM neither provides an explicit definition of service, nor of service system; however, the meaning of such notions is weakly constrained by the UML model reported in Figure 4.8. According to this diagram, a service is described as an aggregate of several heterogeneous entities, notably including resources, which are not explicitly mentioned in the other approaches. The inclusion of entities as disparate as processes, resources and concepts as *parts* of a service is however confusing from the ontological point of view, and hopefully the present analysis will help realizing the need of some cleanup, especially from the conceptual point of view. The main point of the IBM-SDM is to provide a clear path to practical simulation models of service delivery as described in [11].

### *4.7.2 Service Application Perspective*

To better understand the various approaches presented so far, it is useful to discuss the different application perspectives they focus on, which determine the different service modeling goals and motivations.

Alter's framework models a service system as a work system, adopting explicitly a *business perspective*. His notion of *work system lifecycle* is pretty close to the *service system lifecycle* introduced in the General Service Model, whose viewpoint can however be better described as focusing on *service interactions*, i.e., on all interactions taking place in the service system (so among actors, artifacts, resources and the surrounding natural and institutional environment). Among other things, the GSM takes also into account the legal aspects of customer interaction, which are not considered in Alter's framework.

The TEXO Service Ontology's application perspective is on *service evolution*. The *service lifecycle* loops between the innovation, offering, matchmaking, usage and feedback phases. In each phase, both *master data* (that seldom change over time, such as licenses) and *transactional data* (that relate to a single service transaction, such as value exchange data) are exchanged.

SOA-RAF's focus is on the *integration between business needs and the available information technology*, as a key requirement to fully exploit a service-oriented architecture (SOA), intended mainly as "a paradigm for organizing and utilizing distributed capabilities (...)." While the goal of SOA-RM (the SOA Reference Model) is "to define the essence of service-oriented architecture, and emerge with a vocabulary and a common understanding of SOA"), the Reference Architecture Founda-

tion (SOA-RAF) focuses "on the elements and their relationships needed to enable SOA-based systems to be used, realized and owned. So, the emphasis of SOA-RAF is on service systems as composed by distributed capabilities that belong to a larger "ecosystem." Both SOA-RAF and SOA-RM model the service system as a snapshot, without referring to a service lifecycle.

IBM-SDM's main application perspective is on *reuse of service artifacts, and on the use of simulation as a design tool*. Three key stakeholders are present in the service lifecycle: the owner, the beneficiary (sometimes also called customer), and the service provider. The distinction between owner and provider resembles very closely the GSM one between provider and producer, as the owner is the one in charge of the business management of the service, while the provider is like an operations manager, responsible for day-to-day delivery. The term customer is instead more, as it may sometimes refer to the requestor and sometimes to the actual beneficiary, depending on context. The service lifecycle loops between design, solution, transition, and delivery. But only design and solution have been modeled so far.

### *4.7.3 Service System Perspective*

While the service application perspective concerns the service system as a whole, two different modeling perspectives can be isolated concerning how the service system is perceived and described from its two main players, namely the owner/provider and the customer/consumer. Here we shall label as inside-out an approach centered on the former perspective, and outside-in an approach centered on the latter perspective. The inside-out perspective focuses on how the service system satisfies the provider's needs; the outside-in perspective focuses on how the service system satisfies the customer's needs. Note that the choice to use the term "inside" for the provider perspective and "outside" for the customer perspective is arbitrary.

Obviously at some point in the service lifecycle the two visions converge, but they are initially different, since, for instance the customer awareness of the need for a service is a completely different starting point from the provider awareness that such a need exists. The customer often has a clear idea of what he or she wants, while possibly ignores what is really feasible or what is economically convenient, and vice versa for the provider. The IBM-SDM model attempts to converge between both outside-in and inside-out by promoting a participatory style of service design, taking into account not only the provider and customer, but also the needs and values of the delivery organization.

Note that this distinction turns out to be orthogonal to that between glass- and black-box models, as both the inside-out and the outside-in views can show internal activities vs. external behavior.

It is also different from the top-down vs. bottom-up distinction, as in the former case the two perspectives (outside-in and inside-out) are not related to how the modeling activity is performed: under both perspectives one can decide to analyze the service from details up or from the top into details.

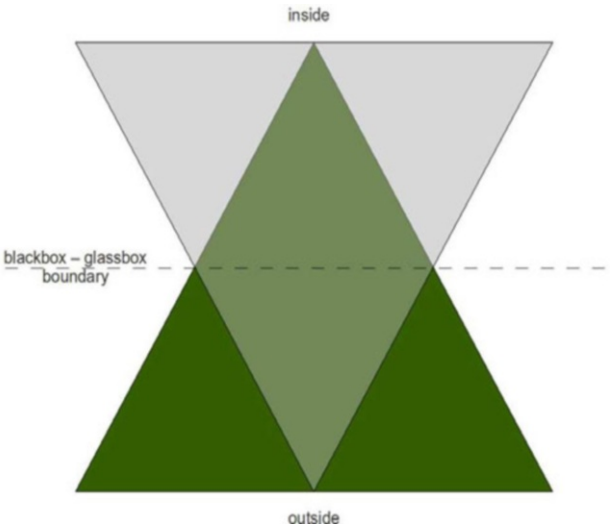


Fig. 4.9: In-out side approach vs black-glass box.

In Figure 4.9 the relation between the inside-out/outside-in choice and the black-box/glass-box choice is summarized. The upper (light) triangle represents the service system modeled according to the provider's point of view (inside-out perspective), while the bottom (dark) triangle represents the service system modeled according to the customer's point of view (outside-in perspective). The larger base of each triangle represents a larger amount of details about the particular point of view encoded in the model. The upper part of each triangle (crossing the "box boundary line") represents that part of the model that accounts for the other party's perspective.

Both approaches are compatible with the black- and the glass-box views. From the provider's view-point, the focus is on "internal" aspects of the service. The model has to be considered a black-box one when only the lowest part of the upper (light) triangle (of the service) is shown to the customer. It is instead a glass box when the whole (or almost the whole) upper triangle is visible to the customer. From the customer's viewpoint, a glass box model corresponds to the whole lower triangle, while a black box model corresponds to the upper part of the lower triangle, denoting a model which only picks up those customer's details which are of interest from the provider's point of view.

The intersection of the two triangles — the center diamond — is the common area of knowledge about the service; the four smaller triangles outside the intersection represent knowledge known to one side but not the other. The two small not overlapping upper triangles could refer to technical and strategic aspects that the provider wants to keep hidden to the customer. The two small not overlapping bottom triangles represent the customer's expectation on the service (what the ser-

vice should do and why the customer wants to use the service in his or her own application context).

Both inside-out and outside-in approaches are important in modeling a service, as in a glass box view, the larger is the section crossing the service border, the better is the model for both the provider and the consumer.

With respect to the two perspectives described above, Alter's approach lies in the middle, because although starting from an "inside" point of view, in the work system framework (taking into account participants, information, technologies, and processes and activities in a context of infrastructure, strategies and environment), he puts the customer and his/her role in the "service value chain framework," investigating his/her responsibilities in the service process.

The GSM aims at being comprehensive, but although the customer's perspective is taken into account while modeling the value co-production process, it has probably a bias towards an inside-out approach, because the internal aspects of the service process are accounted for in some more detail.

The approach of the TEXO Service Ontology is an inside-out approach, although the distinction between master and transactional data is useful for the customer in the understanding of the service, too.

SOA-RAF has aspects of both: the inside-out approach tries to answer the question "what I have to build and how I promote the use of the service system with the customer/consumer?;" the outside-in approach looks for a consumer to evaluate whether the service adequately addresses a need in a manner consistent with acceptable conditions of use.

IBM-SDM is an inside-out approach, because its focus is on how the service is built and how it is presented (delivered) to the customer.

### *4.7.4 Service Science Readiness*

An interesting way to compare service modeling approaches is to discuss, so to speak, their *service science readiness*, i.e., the extent to which they are suitable to be adopted within the large interdisciplinary perspective known with the term *service science*. To this purpose, we shall briefly discuss the presence of interdisciplinary aspects in the modeling proposals described so far.

Alter's framework does not explicitly adopt an interdisciplinary approach, as it mainly focuses on the work system under a business perspective, but certainly his work is in the spirit of a broad service science, and can be used by multiple categories of stakeholders, including non-business professionals in some of the proposed "roles."

The GSM approach is deliberately interdisciplinary, as it relies on theories belonging to philosophy, cognitive science, linguistics, and aims at including juridical and deontic notions as well as business process aspects and economic notions.

The TEXO Service Ontology adopts also an interdisciplinary approach, being based on different modules which account for business, legal, and industry aspects.

Industry modules are populated at run-time to match specialized service application contexts (automotive, healthcare, public services, . . . ).

SOA-RAF (SOA-RM) is partly interdisciplinary, in the sense that it is built from an IT perspective, but it is oriented towards ecosystems.

IBM-SDM initially lacked an interdisciplinary perspective, as it was mainly intended for IT services; nonetheless, the latest works focus on a more general notion of service, in which the organizational side is taken into account. The goal is to introduce engineering rigor in the design process, not to anchor concepts to a shared service ontology.

### *4.7.5 Value Modeling*

Since a core aspect of services is their (co-)production of value, the ability to account for the value production (or value exchange) process is an important aspect to consider while comparing the different modeling approaches. According to Alter's and other approaches, value production is not an exclusive responsibility of the producer.[14] Of course, we should also observe that value (as well as cost) has not just a monetary nature, so the notions of benefit and sacrifice for the two parties involved (service provider and service beneficiary) should be somehow considered in a proper account of value.

Alter explicitly considers value in the *service value chain framework*, which augments the *work system framework*. Because the value of a service is co-produced by consumer and producer, a chain of responsibility for each activity involved in the service process can be identified.

One of the three key components of the service system according to the General Service Model is service value co-production, seen as a complex event in which symmetric events corresponding to provider's or customer's costs and benefits co-occur. The value that is exchanged during such event is the result of positive and negative import of value both from the side of the provider and of the customer (value co-production) throughout all the phases of the service life-cycle, which are inspired by Porter's value chain.

The TEXO Service Ontology explicitly considers value exchange as generated whenever a service is consumed.

SOA-RAF (SOA-RM) represents values as *real world effects*. It could be the response to a request or a change of state for some defined entities. SOA-RAF distinguishes between *social effects* and *physical effects* as goals for the reference architecture.

IBM-SDM associates value with every service object, as a numerical estimate of the value provided by that object to each stakeholder. The model estimates costs

[14] In the present chapter, value co-creation is seen as a process that involves the whole service system and is mainly analyzed as one among several components of the service system life-cycle. Chapter 3 of this book is expressly dedicated to approaches that focus on service value networks.

and values of services, focusing on the values perceived by all different stakeholders. More precisely, value is represented as a vector with one component for each stakeholder.

### *4.7.6 Service Contract*

With the term "contract" we refer to an event in the service life cycle in which something is established that can be used by all involved parties to "judge" whether the value delivered matches all expectations. For such reason, "contract" and "value" are two different but related concepts, as the former could constitute a framework to evaluate the latter (although actual value may go beyond the contractual terms). An important part of a service contract is the Service Level Agreement (SLA). The contract may also be interpreted as a legal object, which can be used by any participant in the service process to enforce an action.

Alter's framework includes service awareness and negotiation in the *service value chain framework*. He assumes however that a service is always co-produced by customer and provider from early stages, independently of any notion of contract.

In the General Service Model there are two events that are connected with the idea of negotiation and contract: one is the service commitment, which in a sense constrains the provider to guarantee that the service is executed in a certain way. Note that such commitment is a generic one, as the service description is about types of action and does not refer to specific customers. But there is another event, customized planning and delivery, which is duplicated for each customer, and occurs after a negotiation, which results in a customization of the general constraints contained in the service description. So there is at least a clear room to include a notion of service contract.

The TEXO Service Ontology explicitly envisions service contracts between service customers and providers as part of the transactional data belonging to a particular service lifecycle. The notion of contract is however somewhat simplified, as it doesn't take into account the possibility that customer and consumer may be different entities.

SOA-RAF (SOA-RM) introduces *contract and policy* as one of the principal concepts involved. "A *policy* represents a constraint or condition on the use of the service." It is an assertion stated by one participant on the conditions of use of some resource. "A *contract* represents an agreement between two or more parties."

IBM-SDM is missing an explicit notion of service contract, apart from modeling it as a generic quality parameter that should be numerically represented in a service quality object. As the model allows representing service constraints, contract targets (as requirements on the quality of service) can be implemented as performance constraints.

### 4.7.7 Reusability

As it happens in any engineering process, a model is considered a solid one if it can be applied on a (relevant) number of application contexts. In the context of service science, with "reusability" what is meant is the capability to reuse the proposed modeling approach, rather than the capability of a model to reuse external components. Both viewpoints are important, of course, but since service systems involve knowledge and participants from different domains, it is even more straightforward that any proposed model has to reuse, at least, core concepts and components of each involved domain.

Alter's framework describes service systems from a business viewpoint, with no assumption whether IT is involved. The framework can be reused for any service system, automated or not.

Reusability is an implicit goal both of DOLCE and of the General Service Model as well. First of all, the level of generality is such that it is fairly straightforward to specialize the primitives to adapt them for more special tasks. On the other hand, the GSM can also be easily extended by adding elements built starting from DOLCE.

The TEXO Service Ontology is reusable in modeling any service lifecycle from the evolution viewpoint, as long as in the setup process of the service itself the innovation, the offering, the matchmaking, the usage, and the feedback phases are planned.

SOA-RAF (SOA-RM) is a reference architecture and is reusable by definition.

IBM-SDM reusability concerns configurability, variability, and extensibility of ServiceObjects used in the modeling process, at design level.

### 4.7.8 Service Time Frame

With the locution "Service Time Frame" we mean the overall period of time considered by a service model. For example, the time frame can span from the start of a contract to its end. Alternatively, the evolution of a service business model could start from initial design and proceed to final delivery. The different approaches described here differ in the time frame they assume. In particular, it is important to distinguish:

- the *service delivery time frame* concerning a service delivered to a specific customer, within the temporal validity of a specific contract;
- the *service commitment time frame* where a generic service commitment (involving multiple potential customers) continues to hold, satisfying the same generic description;
- the *service evolution time frame* concerning the evolution of a service system, where the service description changes in time to account for maintenance changes, new service policy choices, feedback from customers, and so on.

Alter's framework clearly adopts a service evolution time frame, as it focuses on service systems that "evolve through a combination of planned and unplanned changes." The temporal phases considered are "operation and maintenance," "development" and "implementation."

The GSM adopts instead a service commitment time frame, as the service system life cycle starts with the service commitment and ends with the service dismissal. The service delivery time frame is embedded in the service commitment time frame as a sub-event (indeed, a plurality of them, one for each customer contract), while the service evolution time frame is not modeled explicitly. There is a possibility to account for a service evolution time frame by allowing multiple services (at different temporal stages) within the same service system lifecycle, but what needs to be clarified, from the ontological point of view, is the notion of persistence of a service through time. Presently, in GSM a service loses its identity if the service commitment changes.

The TEXO Service Ontology focuses on the *service evolution time frame*, modeling a service whose generic description can change while it "loops between the innovation, offering, matchmaking, usage, and feedback phases," in a fashion very similar to Alter's model.

The SOA-RAF doesn't apparently adopt any time frame, as it models an evolving snapshot of a service system. For example, there are no time constraints concerning contracts and policies.

The IBM-SDM focuses also, such as the TEXO approach, on the service evolution time frame, whose phases are however slightly different from TEXO's ones: design, solution, transition, delivery and end of life.

## 4.8 Concluding Remarks

In this chapter, we have illustrated several approaches to service systems modeling. The key point is that while the various models emphasize different aspects of a larger system, there is consensus that the service conceptual models should account not only for the core service activities, but also for the business and social environment in which the service is used. While a complete comparison of the various approaches is not feasible, we have highlighted the differentiating points of focus of each of these approaches so as to enable designers to choose what is more suited to his/her own needs.

Abstracting away from the specificities of every single approach, in these concluding remarks we would like to make some considerations about the importance of service systems modeling.

First, in the services domain, it is important to have reference models (and several of the presented approaches can be considered as such) to facilitate sharing the fundamental primitives needed to define service ecosystems. The use of reference models enhances mutual understanding and improves the likelihood of interoperability.

Second, the shift from simply modeling services to modeling service systems helps to explicitly account for the social implications of the various aspects of service processes, making the whole service system transparent to designers and stakeholders.

Finally, it is noteworthy that most of these approaches make an explicit separation between design and realization level, distinguishing the service description and what is declared in contracts from the actual service execution. This allows one to verify the compliance of the executed actions with those specified in the service description, including any additional conditions agreed to in explicit contracts. Moreover, the interplay between these two levels is at the basis of complex chains of responsibilities, with duties and rights that can also be transferred through delegation. Models including the explicit representation of responsibility chains enable transparency and predictability, thus providing greater trustworthiness of services.

## References


1. Wikipedia entry on participatory design. Accessed March 2011, http://en.wikipedia.org/wiki/Participatory_design.
2. Reference Architecture Foundation for Service Oriented Architecture 1.0. Committee Draft 2, OASIS, Oct 2009. http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/soa-ra-cd-02.pdf.
3. S. Alter. *The Work System Method: Connecting People, Processes, and IT for Business Results*. Work System Press, Larkspur, CA, USA, Apr 2006.
4. S. Alter. Service responsibility tables: A new tool for analyzing and designing systems. In *Proceedings of the Thirteenth Americas Conference on Information Systems (AMCIS 2007) Keystone, Colorado, August 09 - 12 2007*, 2007.
5. S. Alter. Service system fundamentals: Work system, value chain, and life cycle. *IBM Systems Journal*, 47(1):71–85, 2008.
6. S. Alter. Viewing systems as services: A fresh approach in the IS field. *Communications of the Association for Information Systems*, 26(11), 2010.
7. G. Banavar, A. Hartman, L. Ramaswamy, and A. Zherebtsov. A formal model of service delivery. In P. P. Maglio, C. A. Kieliszewski, and J. C. Spohrer, editors, *Handbook of Service Science*, Service Science: Research and Innovations in the Service Economy, pages 481–507. Springer US, 2010.
8. C. Baumann and C. Loës. Formalizing copyright for the internet of services. In G. Kotsis, D. Taniar, E. Pardede, I. Saleh, and I. K. Ibrahim, editors, *iiWAS'2010 - The 12th International Conference on Information Integration and Web-based Applications and Services, 8-10 November 2010, Paris, France*, pages 714–721. ACM, 2010.
9. M. Böttcher and K.-P. Fähnrich. Service systems modeling: Concepts, formalized meta-model and technical concretion. In H. Demirkan, J. C. Spohrer, and V. Krishna, editors, *The Science of Service Systems*, Service Science: Research and Innovations in the Service Economy, pages 131–149. Springer US, 2011.
10. H. Chesbrough and J. Spohrer. A research manifesto for services science. *Commun. ACM*, 49(7):35–40, 2006.
11. K. A. Dhanesha, A. Hartman, and A. N. Jain. A model for designing generic services. In *2009 IEEE International Conference on Services Computing (SCC 2009), 21-25 September 2009, Bangalore, India*, pages 435–442. IEEE Computer Society, 2009.
12. R. Ferrario and N. Guarino. Towards an ontological foundation for services science. In J. Domingue, D. Fensel, and P. Traverso, editors, *Future Internet - FIS 2008, First Future*

*Internet Symposium, FIS 2008, Vienna, Austria, September 29-30, 2008, Revised Selected Papers*, volume 5468 of *Lecture Notes in Computer Science*, pages 152–169. Springer, 2008.
13. R. Ferrario, N. Guarino, and M. Fernandez-Barrera. Towards an ontological foundation for services science: The legal perspective. In G. Sartor, P. Casanovas, M. A. Biasiotti, M. Fernandez-Barrera, P. Casanovas, and G. Sartor, editors, *Approaches to Legal Ontologies*, volume 1 of *Law, Governance and Technology Series*, pages 235–258. Springer Netherlands, 2011.
14. R. Ferrario, N. Guarino, C. Janiesch, T. Kiemes, D. Oberle, and F. Probst. Towards an ontological foundation of services science: The general service model. In *10th International Conference on Wirtschaftsinformatik, 16th - 18th February 2011, Zurich, Switzerland*, pages 675–684, 2011.
15. C. Fillmore. Types of lexical information. In D. Steinberg and L. Jacobovitz, editors, *Semantics. An Interdisciplinary Reader in Philosophy, Linguistics and Psychology*. Cambridge University Press, London, UK, 1971.
16. A. Gangemi, N. Guarino, C. Masolo, A. Oltramari, and L. Schneider. Sweetening ontologies with dolce. In A. Gómez-Pérez and V. R. Benjamins, editors, *Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, 13th International Conference, EKAW 2002, Siguenza, Spain, October 1-4, 2002, Proceedings*, volume 2473 of *Lecture Notes in Computer Science*, pages 166–181. Springer, 2002.
17. T. Kiemes and D. Oberle. Generic modeling and management of price plans in the internet of services. In K.-P. Fähnrich and B. Franczyk, editors, *Informatik 2010: Service Science - Neue Perspektiven für die Informatik, Beiträge der 40. Jahrestagung der Gesellschaft für Informatik e.V. (GI), Band 1, 27.09. - 1.10.2010, Leipzig*, volume 175 of *LNI*, pages 533–538. GI, 2010.
18. T. Kiemes, D. Oberle, and F. Novelli. Towards a reusable and executable pricing model in the internet of services. In G. Kotsis, D. Taniar, E. Pardede, I. Saleh, and I. K. Ibrahim, editors, *iiWAS'2010 - The 12th International Conference on Information Integration and Web-based Applications and Services, 8-10 November 2010, Paris, France*, pages 722–729. ACM, 2010.
19. C. M. MacKenzie, K. Laskey, F. McCabe, P. F. Brown, and R. Metz. Reference Model for Service Oriented Architecture 1.0. Oasis standard, OASIS, Oct 2006.
20. P. Maglio and J. Spohrer. Fundamentals of service science. *Journal of the Academy of Marketing Science*, 36:18–20, 2008.
21. P. P. Maglio, S. Srinivasan, J. T. Kreulen, and J. Spohrer. Service systems, service scientists, ssme, and innovation. *Commun. ACM*, 49(7):81–85, 2006.
22. C. Masolo, S. Borgo, A. Gangemi, N. Guarino, and A. Oltramari. Ontology Library (final). WonderWeb Deliverable D18, Dec 2003. http://wonderweb.semanticweb.org.
23. D. Oberle. Service ontology final report. Deliverable D.TEXO.9.3.2b, BMWi, Theseus Programme, Use Case Texo, NOV 2010.
24. D. Oberle, N. Bhatti, S. Brockmans, M. Niemann, and C. Janiesch. Countering service information challenges in the internet of services. *Journal of Business & Information System Engineering (BISE)*, 5, 2009.
25. C. Riedl, N. May, J. Finzen, S. Stathel, V. Kaufman, and H. Krcmar. An idea ontology for innovation management. *Int. J. Semantic Web Inf. Syst.*, 5(4):1–18, 2009.
26. J. C. Spohrer, P. P. Maglio, J. H. Bailey, and D. Gruhl. Steps toward a science of service systems. *IEEE Computer*, 40(1):71–77, 2007.
27. Y. Taher, W.-J. van der Heuvel, S. Koussouris, and C. Georgousopoulos. Empowering citizens in public service design and delivery: a reference model and methodology. In *Proceedings of the Service Modelling And Representation Techniques Workshop (2010)*, 2010.
28. O. Terzidis, A. Fasse, B. Flügge, M. Heller, K. Kadner, D. Oberle, and T. Sandfuchs. Texo: Wie THESEUS das Internet der Dienste gestaltet — Perspektiven der Verwertung. In L. Heuser and W. Wahlster, editors, *Internet der Dienste*, acatech diskutiert, pages 141–161. Springer, 2011.
29. S. L. Vargo and R. F. Lusch. Evolving to a New Dominant Logic for Marketing. *The Journal of Marketing*, 68(1):1 – 17, 2004.

30. E. Wittern and C. Zirpins. On the use of feature models for service design: the case of value representation. In *Proceedings of the Service Modelling And Representation Techniques Workshop (2010)*, 2010.