Definition of a Description Language for Business Service Decomposition*

Lam-Son Lê, Aditya Ghose, and Evan Morrison

School of Computer Science and Software Engineering
Faculty of Informatics, University of Wollongong
New South Wales 2522, Australia
{lle,aditya,edm92}@uow.edu.au

Abstract. In the last few years, service-oriented computing has become an emerging research topic in response to the shift from product-oriented economy to service-oriented economy and the move from focusing on software/system development to addressing business-IT alignment. From an IT perspective, there is a proliferation of methods and languages for describing Web services. There has not been as much work in defining languages or ontologies for describing services from business perspectives.

In this paper, we analyze the landscape of service representation and discuss the needs of having a description language for business services. By leveraging existing work on describing service capabilities and properties, we define a specific description language that explicitly addresses the decomposition of business services and their non-functional properties. The language is defined both informally (as a list of descriptive concepts) and formally (by means of meta-modeling and declarative modeling).

Keywords: Service-Oriented Computing, Service Engineering, Strategic Alignment, Business-IT Alignment, Description Language, Meta-modeling.

1 Introduction

In the past ten years, software and system modeling have become rapidly growing and high profile topics in the field of information systems. The proliferation of methods for modeling software and systems has expanded dramatically to many different paradigms, including: component-based software development, rapid application development, iterative and incremental development.

Currently, there are two emerging topics of interest: service-oriented computing and enterprise architectures, the focus has shifted from software and system development to the convergence of enterprises, organization and information systems. We deal with not only software and system development, but also the way they are exploited to make business more efficient and effective. Enterprise architecture deals with the alignment between business and Information Technology

* Funding of this research was provided by the Smart Services CRC Initiative http://www.smartservicescrc.com.au/

J.-H. Morin, J. Ralyté, and M. Snene (Eds.): IESS 2010, LNBIP 53, pp. 96–110, 2010.

(IT) in order to make the enterprise more competitive (e.g. more cost-effective, better client support) [13]. In service-oriented computing, the goal is to produce a more modular and loosely coupled organizational system, where changes within the organization is managed in a less risky fashion than more traditional major change systems [9].

From an IT perspective, there is a proliferation of methods and languages for representing Web services. There has not been as much work in defining languages or ontologies for describing high-level services from a business perspective. In a broader context, we need a new approach in describing the long-term strategy of an organization and the way it is aligned to high-level business services. There is a need to understand the degree of strategic alignment of a service portfolio to support service re-alignment in the face of changing strategic landscapes.

Motivated by the needs of describing business services from a pure business perspective, we define a specific description language that explicitly addresses the decomposition of business services and their non-functional properties. We base this work on existing work that established basic description for service capability and properties.

The remainder of this paper is structured as follows. Section 2 discusses the needs of a dedicated description language for business services and their decomposition. The description language that we propose for business services is defined in Section 3. Section 4 presents work related to the representation of business services. Section 5 ends the paper by drawing some conclusion remarks and discussing future work.

2 The Needs of Describing Business Services and Their Decomposition

In this section, we discuss why we need a specific description language for business services. Subsection 2.1 addresses the landscape of service representation within which the representation of business services is positioned. Subsection 2.2 describes an example that will be used for formulating the requirements of such a description language. Requirements of a description language for business services are presented in Subsection 2.3.

2.1 Representation of Services

Figure 1 illustrates a description continuum of strategy, goals, services and processes that would be necessary for describing an organization. This continuum has two ends that correspond to the specification and the operationalization of an organization. High-level specifications and long-term strategies appear to the left most side of the spectrum. The granular detail increase towards the right of the figure until operationalization of high-level specifications and strategies is achieved.

To the specification end of the continuum lay the modeling languages for strategy and goals. These modeling languages are used for capturing the vision and the requirements of an organization. Some research has been put forward in this area. The e3 Forces [11] proposes a framework for modeling 3 perspectives of an organization one of which is focused on business strategy modeling. In the InStAl method [16], strategy of an organization is represented in terms of strategic objectives and strategic goals with respect to the vision of the organization's stakeholder. Goal-oriented Requirement Language [19] supports goal-oriented reasoning by establishing correspondences between intentional elements (goal, softgoal, task, believe, resource) and non-intentional elements - which may be imported from an external model, in a scenario. Other researches on goal modeling include Tropos (an agent-oriented software development method based on goal-oriented requirements) [4], GOORE (a goal-oriented method for requirements elicitation) [14] and Lightswitch (definition of early requirements of an enterprise system) [12]. In our group, we are developing a specific method and an associated toolkit that enable the representation of business strategies and business services in a hierarchical approach as well as the alignment between them. This work, which, together with InStAl method [16] and e3 Forces [11], can be classified as Strategy Modeling Language (SML) as illustrated in Figure 1. While GRL primarily deals with goal modeling as the name suggests, SML nevertheless addresses a wider range of high-level strategy and requirements .

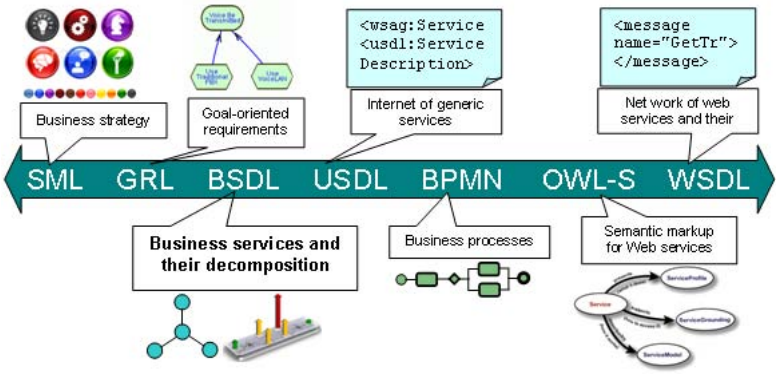

Fig. 1. Representation continuum of services with regard to strategy and goal modeling

Towards the operationalization end, there are different levels of service representation. Business services can be viewed as high-level implementation components that operationalize the organization's strategy. The business services can be decomposed (e.g. by subcontracting) and refined (e.g. by adding properties). This decomposition and refinement may result in a network of generic services that can be represented using Universal Service Description Language (USDL) [1] or Description of Service Capabilities and Properties (DSCP) [10]. A more

technical description language such as Web Service Description Language[1] could be used when IT supports such as Web services are needed. OWL-S[2] provides an ontology for describing functionality of web services, how they can be used and how to interact with them.

In this paper, we propose a description language for business services and their decomposition that we call Business Service Description Language (BSDL). It establishes the missing link between GRL and USDL by taking into account service decomposition and looking at services at a higher level than other service description languages do.

It is worth noting Business Process Modeling Notation (BPMN) - the most popular modeling language for business processes [18]. In the representation continuum of Figure 1, BPMN is placed closer to the operationalization end than BSDL and USDL are. This can be judged in the way that business processes are regarded as instances of generic services and business services.

2.2 Motivation Example

Let us consider an example. A large-scale construction company called Bridge-Builder (BB) offers bridge construction as a service. The company can describe this service to their potential clients (e.g. government) as follows: the construction of bridge will be done in a cost-effective, schedule-manageable manner but the total cost and the construction schedule should be negotiated based on technical specification given by the client.

Before building a bridge, the BB company and their client would: (i) detail technical specification, (ii) negotiate schedule constraints and financial issues, (iii) elaborate penalty conditions that may be applied in case the schedular constrains or the technical requirements are not met. After having reached agreement on these details, the BB company would start this business by breaking down the service "Bridge Construction" they offer into four constituent services each of them can be subcontracted to other companies who specialize in a specific area (see Figure 2). The constituent services are

- pre-construction clearance: ground needed for building the bridge is cleared
- pillar construction: pillars of the bridge are constructed
- span construction: spans of the bridge are built
- lighting facility: lighting systems are equipped on the bridge

In Figure 2, each bubble stands for a business service. The text inside each bubble describes the name, schedule obligation and penalty for not meeting the schedule of the service being represented. The four thick arrows coming from the bubble in the center of the figure represent subcontracting.

There are several challenges in describing the "Bridge Construction" service and its decomposition. First, in addition to describing the main function of this

[1] W3C Web Service Description Language `http://www.w3.org/TR/wsdl`
[2] OWL-S: Semantic Markup for Web Services `http://www.w3.org/Submission/OWL-S/`

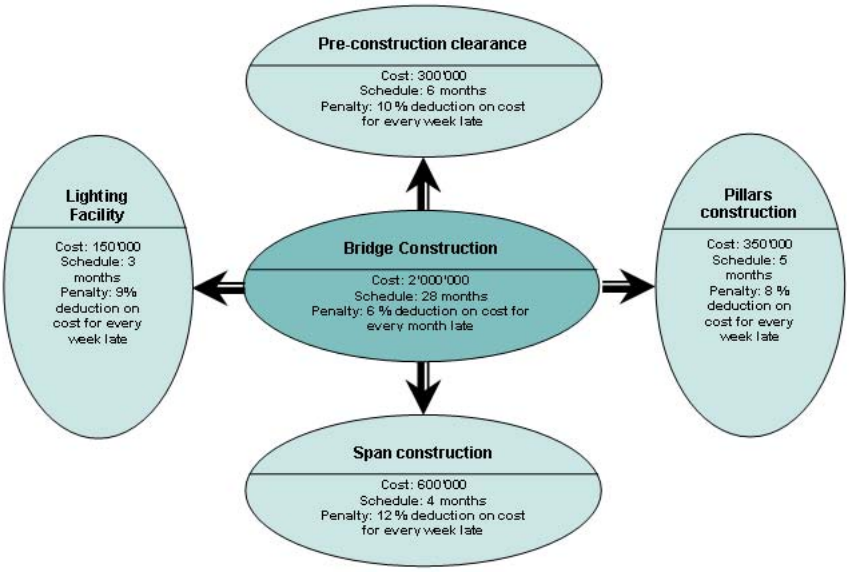

Fig. 2. Decomposition of the service "Bridge Construction" results in distributing its functional and non-functional properties into constituent services

service, we need to explicitly represent its *non-functional properties* including the costs, schedule, penalty as illustrated in Figure 2.

Second, as the the "Bridge Construction" service is decomposed into four *constituent* services, the same extent of description should be achieved for all of its constituent services.

Third, the company that offers one of the consistent services may decide to *further* subcontract, outsource or delegate part of its service, leading to additional service decomposition.

2.3 Description Language Requirements

The challenges pointed out in the previous subsection drive us to formulate the following requirements of a description language dedicated to the representation of business services and their decomposition.

1. **distribution**: functions and non-functional properties of a business service are distributed into constituent services
2. **decomposition**: a set of concepts that can explicitly describe how to break down business services
3. **uniformness**: the above abilities should uniformly be applied across the decomposition hierarchy of business services

In Section 3, we define a description language that addresses the requirements listed above. We call this language the Business Service Description Language (BSDL).

3 Business Service Description Language

This section first gives an informal definition of BSDL (Subsection 3.1). The BSDL is then formally defined by means of meta-modeling (Subsection 3.2).

3.1 Informal Definition of BSDL

The DSCP is a description language for business services [10]. This language addresses the description of both the functional aspect and non-functional properties of a business service. In our understanding, this work has the most detailed description of business services. We base our BDSL on this work[3], particularly in the representation of service capability whilst adding the decomposition of business services. We also take a different approach for formalizing our BSDL (see Subsection 3.2).

In DSCP [10], the functional aspect of business services is captured by service capabilities whilst the non-functional aspect is covered by service properties. We borrow these descriptive concepts while adding more concepts that describe the decomposition of business services and some non-functional properties. Table 1 summarizes the descriptive concepts defined in the BSDL. Each concept is defined in English and is classified either as basic, functional, non-functional, lexical or decomposition. Note that the added concepts are marked with star symbols (*) in this table.

3.2 Formal Definition of BSDL

Meta modeling is a popular approach for formally defining modeling and description languages. In this subsection, we present a meta-model that formally defines the BSDL. Typically, a meta-model has three elements: a diagram, description rules and sample instantiation. The diagram gives visual representation of the concepts defined in the description language being formalized by the meta-model. The diagram also visually shows relationships between these concepts as well as cardinalities and roles specified in the relationships. But diagrammatic representation is usually weak in capturing rules that a description language may have. We need a more formal means to represent the rules. The third element of a meta-model illustrates how instant models of the description language can be instantiated from the meta-model. The existence of this element is necessary to prove that the first element and the second element are consistent and the meta-model as a whole can be instantiated. A non-trivial sample instantiation is usually included in the meta-model for this purpose.

The DSCP [10] is formalized using Object Role Modeling (ORM) [5] - a visual conceptual data modeling technique with the advantage of being able to include a sample population directly in the ORM diagram that helps to validate the model and demonstrate how it is used. In this approach, ORM diagrams give visual representation of the concepts defined in DSCP. The way that ORM include the rules

[3] An online version of DSCP can be found at `http://www.service-description.com`

Table 1. The building blocks of BSDL are informally defined in English. Some of them are originated from DSCP [10]. The newly-defined building blocks are marked with *.

Concept	**Group**	**Informal definition of element**
Business Service	Basic	Represents a high-level service that is provided by a business entity (e.g. an enterprise, an organization, an individual).
Provider	Basic	Represent a business entity (e.g. an enterprise, an organization, an individual). A provider may not be a system that can operate without any human activities. A business service may be provided by more than one provider.
Requester	Basic	Represent a business entity (e.g. an enterprise, an organization, an individual) that requests a business service.
Capability	Functional	Represents the function of a business service does. A business service may have more than one capability.
Rule	Functional	Represents an effect or a pre-condition of a Capability. A capability may have multiple preconditions and effects.
Signature	Functional	Represents input or output of a Capability. A signature can be regarded as a set of parameters.
Parameter	Functional	Captures a piece of information or data that a business service consumes or produces.
Property	Non-functional	Generic non-functional property of a business service.
Obligation	Non-functional	Represents responsibility that both the requester and the provider of a business service must fulfill. An obligation is associated with a penalty.
*Schedule**	Non-functional	Represents an obligation that mandates the time-frame of a business service.
*Environment**	Non-functional	Represents an obligation that mandates how environment-friendly a business service should be.
Payment	Non-functional	Represents an obligation that mandates how the service requester pays the service providers.
Penalty	Non-functional	A non-functional property associated with an obligation. This property represents penalty applied when the associated obligation is not fulfilled.
Price	Non-functional	A non-functional property that represents the amount of money being charged for a business service from the providers perspective. It may be called costs from the requesters perspective.
Lexical Term	Lexical	Represents a lexical term used in describing capabilities and parameters of business services.
Verb	Lexical	Represents a verb used in describing capabilities of business services.
Noun Phrase	Lexical	Represents a noun or a noun phrase used in describing parameters of business services.
Case Description	Lexical	Used for describing attributes (e.g. topic, location) of a Capability.
Ontological Source	Lexical	Contains definition of lexical terms, case descriptions and rules.
*Decomposition**	Decomposition	Represents the manner in which a business service is broken down into a number of constituent services.
*Subcontracting**	Decomposition	A method of decomposing service by which an external service provider is contracted to perform part of a business service. The part that is subcontracted can be regarded as a constituent service.
*Outsourcing**	Decomposition	A method of decomposing service by which a third-party service provider (potentially be an oversea provider) is contracted to perform part of a business service.
*Co-sourcing**	Decomposition	A method of decomposing service by which some part of a business service is performed both by its provider and by some external provider.
*Delegation**	Decomposition	A method of decomposing service by which some part of a business service is assigned to another service provider, usualy a person.

dealing with cardinality and uniqueness of the DSCP concepts and some instant model in their diagrams is an advantage of this approach. However, it is not clear if ORM diagrams can express all kinds of rule defined for a description language.

The meta-modeling approach that we follow in formalizing the BSDL is to use a Unified Modeling Language (UML)[4] diagram and a declarative language based on the first order logic and the set theory called Alloy [2]. By using a declarative language that has capability of processing the first-order logic, we can formalize a wider range of rules than using a diagrammatic approach in ORM. Using UML to represent the meta-model also brings a benefit regarding the potential implementation of BSDL. The UML diagram of the BSDL meta-model can be imported to quickly build a project baseline in Eclipse Modeling Framework[5] - a popular development environment in research community.

Meta-model of BSDL. The way we build a meta-model for BSDL can be summarized as follows. The descriptive concepts of BSDL (see Table 1) are visually expressed in a UML class diagram. The diagram is complemented by a list of description rules (see Table 2) that define the well-formedness[6] of BSDL. The UML classes and the list of description rules are together formalized in Alloy code which allows verification using object-oriented syntax and the capability of processing in first-order logic. This is an advantage of this approach over using UML and a separate constraint language such as Object Constraint Language[7]. Another advantage is that the Alloy language comes with a tool that allows checking the consistency and helps in generating a sample instant model of the formalized meta-model.

Figure 3 is the UML diagram of the BSDL meta-model. Each BSDL descriptive concepts (that is listed in Table 1) is represented as a UML class. Each `Business Service` is connected to a `Requester` and one or more `Provider`(s) through UML composition relations that have the role names `serviceRequester` and `serviceProviders`, respectively. A `Decomposition` connects a decomposed `Business Service` via the `decomposedService` role name and one or more constituent `Business Service`(s) through the `constituentServices` role name. The `Decomposition` is an abstract class of four concrete classes each of which represents a specific decomposition method for business service.

A straightforward way of representing service decomposition is to use the well-known "Composite Pattern" [3]. This pattern is most suitable for situation where there is a clear distinction between leaf nodes and composite nodes. However, the `Business Service` in this meta-model plays both the role of a decomposed service and the role of constituent services. We cannot tell if a business service is no further decomposed in order to qualify as the leaf in the decomposition hierarchy. In addition, by representing the service decomposition as a

[4] UML Resource Page of Object Management Group `http://www.uml.org`

[5] Eclipse EMF homepage `http://www.eclipse.org/modeling/emf/`

[6] Well-formedness refers to the way that a model is structured in the fashion expected, which is typically specified in terms of rules that are called well-formedness rules.

[7] OCL Specification `http://www.omg.org/technology/documents/formal/ocl.htm`

UML class instead of a UML composite relation, we can enrich the meta-model by specializing it and adding attributes.

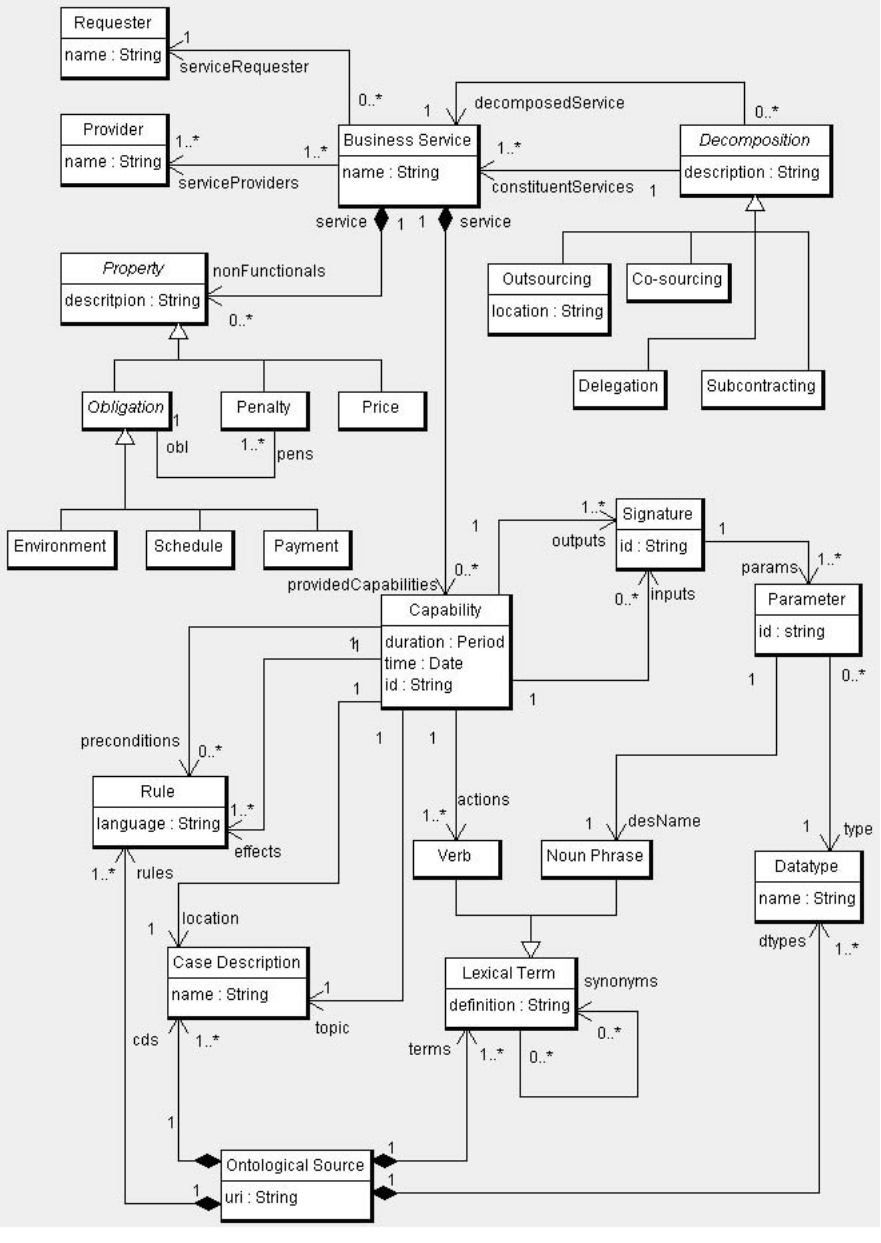

Fig. 3. The descriptive concepts of BSDL and the relationships between them can visually be expressed in a UML class diagram

`Property` is an abstract class that represents a generic non-functional property. This class is connected to the `Business Service` class through a role name called `nonFunctionals`. `Obligation`, `Penalty` and `Price` are subclasses of `Property`. Note that the classes `Obligation` and `Penalty` are connected by a UML association relation in accordance to the definition of the corresponding concepts given in Table 1.

The rest of the UML diagram is actually translated from the meta-model of DSCP [10] that was conceptualized using ORM [5]. Each `Business Service` is connected to some `Capability`(s) through a UML composition relation. A `Capability` is associated to `effects` and `preconditions` all of which are instances of `Rule`. A `Capability` is also associated to `inputs` and `outputs` that are actually instances of `Signature`. A `Capability` is lexically characterized by a `Verb`. A `Signature` has a number of `Parameter`(s) each of which has a `Datatype` and is lexically characterized by a `Noun Phrase`. Note that `Verb` and `Noun Phrase` are generalized into `Lexical Term`. All instances of `Rules`, `Lexical Term` and `Datatype` belong to an `Ontological Source`.

Table 2. Description rules constrain the way instances of BSDL building blocks are put together in a service description model

Rule	**Informal semantics**
Acyclic	There must be no cycle along the decomposition hierarchy of business services.
Mutual	For each business service, all properties and capabilities declared in it must take it as their sole service.
Uniqueness	The set of parameters of two different signatures must be different.
Same service	An obligation and corresponding penalties must be of the same business service.

The UML diagram of Figure 3 offers diagrammatic expressiveness for representing the BSDL descriptive concepts and their relationships but does not cover the well-formnedness of the whole BSDL model other than cardinalities of these relationships. However, the BSDL well-formedness is more than just about cardinality. For example, the decomposition hierachy of business services in BSDL is well-formed if it has no cycle. For this reason, we need a list of description rules that state how instances of BSDL descriptive concepts are put together to make a correct BSDL model. Table 2 lists the description rules of BSDL.

Formalization in Alloy. The UML diagram shown in Figure 3 and the list of well-formedness rules (Table 2) can be formalized together in single Alloy code. As the Alloy language offers an object-oriented syntax, we can translate the UML diagram to Alloy straightforwardly as follows

- A UML class is mapped to an Alloy signature (the `sig` keyword)
- A UML role name is mapped to an Alloy field (to be declared within an signature)

- The UML cardinalities `1`, `0..1`, `1..*` and `0..*` are mapped to the `one`, `lone`, `some` and `set` keywords of Alloy, respectively
- The UML generalization is mapped to the extension mechanism in Alloy with the `extends` keyword
- For the sake of simplicity, we can ignore UML attributes of which types are primitives (e.g. `Date`, `String`) because they are not referred to in the BSDL description rules.

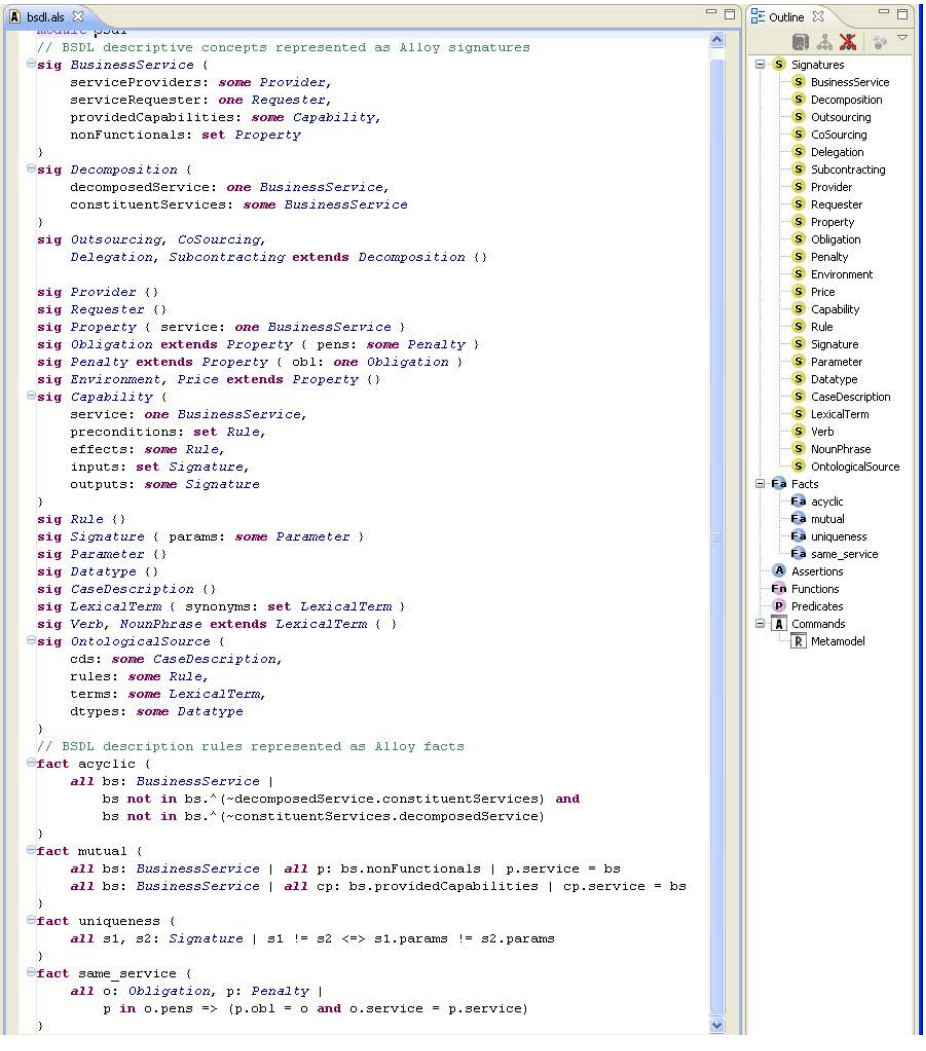

Fig. 4. The meta-model of BSDL is formalized in Alloy. Alloy signatures declare the building blocks while Alloy facts capture the description rules of BSDL.

Translating BSDL description rules to Alloy code can be done in two steps: write first-order logic [15] statements for the BSDL description rules before mapping these statements to Alloy facts (the `fact` keyword). For example, the *Acyclic* rule can be stated as: every business service is found neither the transitive closure of its parent service nor in the transitive closure of its constituent services. This statement is finally translated to the following Alloy formula

```
all bs: BusinessService |
        bs not in bs.^(~decomposedService.constituentServices) and
        bs not in bs.^(~constituentServices.decomposedService)
```

Figure 4 shows the Alloy code that formalizes the BSDL meta-model, including the descriptive concepts and the description rules. There are two panels in this figure. The panel to the left displays Alloy source code whilst its overview is shown in the panel to the right. Note that the names of signatures and fields in the Alloy code match those of the corresponding classes and role names in the UML diagram. Name matching is also observed between the BSDL description rules and the corresponding Alloy facts.

Consistency and Instantiation of the BSDL meta-model. To check the consistency of the Alloy code shown in Figure 4, we need to have it executed. The Alloy language is supported by a tool called Alloy Analyzer[8]. It is possible to add execution and instantiation commands to the code and run it on Alloy Analyzer. If the Alloy code is over-constrained (e.g if there is contradiction between Alloy facts declared in the code), the tool outputs a message notifying that the code is inconsistent and thus no instant model can be generated for it. Otherwise if the code can be instantiated, the tool generates and visualizes an instant model.

Figure 5 displays an instant model generated by the Alloy Analyzer tool. This instant model corresponds to the example presented in Subsection 2.2. In this figure, bubbles and rectangles represent instances of BSDL descriptive concepts. Note that there is text inside each of them. The text has two lines: one is the name of the BSDL descriptive concept, the other (wrapped by parentheses) is the name of the instance the bubble or rectangle represents.

4 Related Work

In this section, we relate the BSDL to existing work on service representation and management. SLA@SOI is a research and engineering project that can embed SLA-aware infrastructures into the service economy[9]. On of the key points of this project is to build an automated e-contracting framework that manages Service Level Agreement (SLA) for business services. However, a specific description language for business services and their decomposition is not addressed.

WSDL and BPEL4WS are dedicated for web services. Obviously, the properties defined in these languages are specific to web services. The decomposition of services can be represented via a mechanism called service invocation.

[8] Downloads and tutorials of this tool are available at `http://alloy.mit.edu/`
[9] SLA@SOI Project `http://sla-at-soi.eu/`

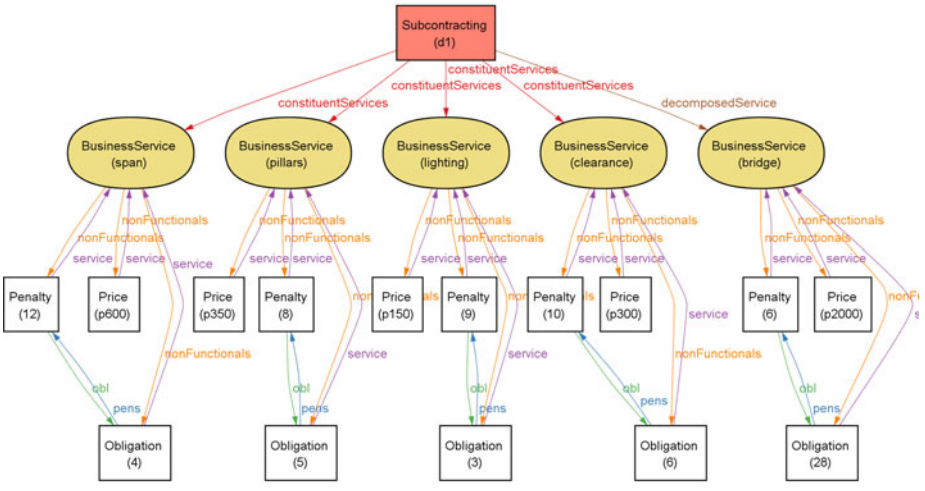

Fig. 5. The "Bridge Construction" example can be instantiated by Alloy Analyzer

The DSCP [10] on which our BSDL is based aims at modeling functional and non-functional properties of business services. To our best understanding, the range of service properties covered in this work is widest among those offered by research work on business service modeling. This work does not deal with the decomposition of business services. In USDL [1], services are considered as generic services that may or may not have technical aspect. Services are coupled from the standpoint of an internet of services. The decomposition of services are not addressed in USDL.

Various work on combining non-functional descriptions with service-oriented architecture has been undertaken. Wenting et. al. offer a weighted metric for service selection based on higher level non-functional goal models [7]. This work relies heavily on domain experts who provide preferences that can be combined to rank and rate the desires of each organization and aid in the decision process for service selection. This work is novel, providing a rank-based preference structure that can be used between higher level strategies and the correlation to services. It complements the work in [8] where a Quality of Service architecture for Web services has been described. The architecture proposed in [8] defines many descriptive terms for services that may be used in an general ontological service framework as well as the mechanics that may be used to incorporate a description language into a complete service provision framework by way of describing many non-functional attributes of services. These articles [7,8] do much in the way of formalizing the required language for service/strategy non-functional descriptions as well as providing implementation mechanisms that would allow services to invoke the ontology; however, the articles offer broad frameworks for representation of services. Key to the representation of services and the higher level business strategy is the vender selection problem[6,17]. In

[17], the authors expand on Weber's early work in vender selection criteria and methods. In this article Weber et. al. identify a series of qualities that a decision system for vender selection must have. They then describe a metric algorithm for the determination of optimal vender number selection. In [6], Kumar et. al. offer a fuzzy goal programming approach to the vender selection problem with multiple objectives, this work breaks down the task of selection using quantitative descriptions of constituent services. All of the aforementioned research does little in the way of framing the problem in a business domain. In this regard much work must be undertaken to resolve inconsistencies between the approaches described above to form a central body or language for describing the relationships between business service providers.

5 Conclusion

Service-oriented computing has become an emerging research topic in response to the shift from product-oriented economy to service-oriented economy and the move from focusing on software/system development to addressing business-IT alignment. In the standpoint of service-oriented computing, services are considered as main vehicle for the operation of an enterprise or organization. Describing services is essential for an organization. There exist description languages that are dedicated to either web services or generic services. From another perspective, describing goals and strategy are necessary for capturing the long-term vision of an organization. To fill in the gap between modeling high-level strategy and the technology-focused representation of services, we define a description language called BSDL from a pure business perspective. The language is dedicated to the representation of business services, in particular the decomposition and non-functional properties of business services.

Future work falls into three directions: (i) improvement of BSDL; (ii) alignment between business services and strategy; and (iii) evaluation. In the first direction, BSDL can be enhanced to cover a wider range of non-functional properties and to provide formal semantics (e.g. by means of first-order logic) for service capabilities and service decomposition. A more technical work is to represent the syntax of BSDL in some markup language. In the second direction, we target the modeling of long-term strategy of an organization and the strategic alignment of business services that the organization offers. The goal of this research is to be able to identify the strategic antecedents of every service and the service-level operationalization of every strategy. In our group, we have an ongoing project where we are developing a specific modeling language for strategic alignment and implementing a toolkit. The definition of BSDL presented in this paper is taken as a baseline for this project. In the third direction, BSDL will be evaluated together with the work on strategic alignment using case-studies provided by industrial partners that get involved in the multi-partner project funding this research.

References

1. Cardoso, J., Winkler, M., Voigt, K.: A service description language for the internet of services. In: Proceedings of First International Symposium on Services Science, Berlin, Germany, March 2009, pp. 1–10 (2009)
2. Jackson, D.: Alloy: a lightweight object modelling notation. ACM Transactions on Software Engineering and Methodology 11(2), 256–290 (2002)
3. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, Reading (1995)
4. Giorgini, P., Kolp, M., Mylopoulos, J., Pistore, M.: The tropos methodology: An overview. In: Methodologies and Software Engineering for Agent Systems. Academic Press, London (2003)
5. Halpin, T.: Information Modeling and Relational Databases: from conceptual analysis to logical design. Morgan Kaufmann, San Diego (2001)
6. Kumar, M.: A fuzzy goal programming approach for vendor selection problem in a supply chain. Computers & Industrial Engineering 46(1), 69–85 (2004)
7. Ma, W., Liu, L., Xie, H., Zhang, H., Yin, J.: Preference model driven services selection. Advanced Information Systems Engineering 1, 216–230 (2009)
8. Maximilien, M., Singh, M.: A framework and ontology for dynamic web services selection. IEEE Internet Computing 8(5), 84–93 (2004)
9. Huhns, M.N., Singh, M.P.: Service-Oriented Computing: Semantics, Processes, Agents. Wiley, Chichester (2005)
10. O'Sullivan, J.: Towards a Precise Understanding of Service Properties. PhD thesis, Queensland University of Technology (2007)
11. Pijpers, V., Gordijn, J., Akkermans, H.: Business strategy-IT alignment in a multi-actor setting: a mobile e-service case. In: Proceedings of the 10th International Conference on Electronic Commerce. ACM, New York (2008)
12. Regev, G., Wegmann, A.: Defining Early IT System Requirements with Regulation Principles: The Lightswitch Approach. In: Proceedings of 12th IEEE International Conference on Requirements Engineering, USA, pp. 144–153 (2004)
13. Schekkerman, J.: How to Survive in the Jungle of Enterprise Architecture Framework: Creating or Choosing an Enterprise Architecture Framework. Trafford Publishing (2004)
14. Shibaoka, M., Kaiya, H., Saeki, M.: Goore: Goal-oriented and ontology driven requirements elicitation method. In: Hainaut, J.-L., Rundensteiner, E.A., Kirchberg, M., Bertolotto, M., Brochhausen, M., Chen, Y.-P.P., Cherfi, S.S.-S., Doerr, M., Han, H., Hartmann, S., Parsons, J., Poels, G., Rolland, C., Trujillo, J., Yu, E., Zimányie, E. (eds.) ER Workshops 2007. LNCS, vol. 4802, pp. 225–234. Springer, Heidelberg (2007)
15. Smullyan, R.M.: First-Order Logic. Dover Publications, New York (1995)
16. Thevenet, L.-H., Salinesi, C.: Aligning IS to Organization's Strategy: The InStAl Method. In: Krogstie, J., Opdahl, A.L., Sindre, G. (eds.) CAiSE 2007 and WES 2007. LNCS, vol. 4495, pp. 203–217. Springer, Heidelberg (2007)
17. Weber, C.A., Current, J., Desai, A.: An optimization approach to determining the number of vendors to employ. Supply Chain Management: An International Journal 5(2), 90–98 (2000)
18. White, S.A., Miers, D.: BPMN Modeling and Reference Guide. Future Strategies Inc. (2008)
19. Yu, E.S.K.: Towards modelling and reasoning support for early-phase requirements engineering. In: Proceedings of RE'97 - 3rd Int. Symp. on Requirements Engineering, pp. 226–235 (1997)