Modeling service systems

Wout Hofman[1] , Yoa-Hua Tan[2],

[1] TNO, P.O. Box 5050, 2600 GB, The Netherlands (wout.hofman@tno.nl)
[2] Faculty TBM, TUDelft, Postbus 5015, 2600 GA, Delft (y.tan@tudelft.nl)

Abstract. Service systems in general take a holistic approach to include several aspects, e.g. organizational structure, economics, people, and IT. They are in principle not limited to organizational boundaries. This paper considers a multi-actor service system of collaborating organizations for value exchange, the Beer Living Lab [33], to analyze the applicability of a particular modeling technique for modeling service systems. The focus of the paper is on information aspects; other papers already describe value exchanges modeled by e^{3}value [33]. WSMO, Web Service Modelling Ontology [1], a formal specification method for web services based on Abstract State Machines [14], is analysed as a basis for modelling information exchange. WSMO consists of a number of concepts to support mediation between goals of service consumers and web service capabilities of service providers. It can easily be applied to simple, atomic interaction patterns like request – response, but usability of WSMO for modelling service systems is insufficient. Modelling service systems with many organizations must support autonomy of those organizations. Mediation can offer solutions, but is not yet fully feasible.

Keywords: Service system, service science, web services, WMSO, e-customs, ITAIDE

1 Introduction

Over the past decades, services have become the most important part of economies [3]. Basically, the service economy refers to the service sector. It leads to more sophisticated forms of cooperation, or what is called value co-creation [4]. From an economic perspective, these systems are described as service systems. Research in this particular area is called service science [4]. There is sufficient technology available for technically integrating these systems, but our basic research question is whether conceptual methods support modelling these types of systems. More in particular, we explore the applicability of WSMO for modelling service systems, since it is technology independent.

We consider the Beer Living Lab, a living lab of ITAIDE, as a useful case since it relates to development of a complex, highly regulatory infrastructure with organizational, legal and IT aspects covering all EU Member States. It is an example of a multi-actor service system. We first present the case, secondly discuss WSMO, and thirdly present the results of our research. We will show, that although WSMO supports modelling complex services, mediation is crucial for these systems. The required levels of mediation are not yet feasible.

2 The Beer Living Lab: a multi-actor service systems

The European Commission has launched the initiative for developing one contact point in all EU member states (Single Window) for customs control of goods movements into (import) and out of (export) the EU and between (transit) EU Member States as part of the e-customs program. ITAIDE, an EU FP6 funded project, contributes to this infrastructure by developing and testing web services in practical applications, one of them being the movement of excise goods (beer) between two EU Member States. Development of web services is based on a data structure that is modelled with UMM (UN/Cefact Modelling Method [2]).

The Beer Living Lab is an example of a service system is briefly introduced with relevant issues from a modelling perspective. In ITAIDE, the focus has been on risk reduction from a customs perspective by introducing two innovations. In this particular paper, we are interested in modelling information exchange from a service systems perspective.

2.1 Introduction to the BeerLL

The Beer Living Lab, abbreviated to BeerLL, is a living lab case study that was conducted in 2006-2007 as part of the research project ITAIDE[1]. Participants in the BeerLL were Heineken NL, a large Dutch beer producer, Heineken UK, an independent company that buys beer from Heineken NL, a UK retailer, and the Dutch Tax Administration. In the BeerLL innovative IT was piloted to introduce electronic customs, where paper-based control procedures were replaced by electronic procedures. The ultimate goal of the project was to use electronic customs to improve the security and efficiency of international supply chains in the beer industry. One of the improvements is the introduction of a smart container seal for monitoring the physical status and movement of a container, the TREC device, and the other web services to constantly monitor this status and the accompanying administrative data. e3value has been applied to model the current and the to-be situation from the perspective of value exchange.

The BeerLL is a complex service system composed of different service providers to transport containers from the Netherlands (input) to the final destination in the UK (output). From a customer's perspective, it may be a pull by a retailer to Heineken UK for delivering a beer. Heineken UK may 'pull' a container from Heineken NL. For timely delivery to a customer, intermediate storage is introduced, e.g. the UK warehouse. A retailer also may have its particular warehouse facilities, in which case the scenario for delivery of beer to a customer is different from the one based on the UK warehouse. Although the case focused on the risk reduction from a customs perspective by introducing two innovations, we use the complete case as the basis for specifying a service system like the BeerLL in WSMO (see further).

The BeerLL represents one value chain in the selling and distribution of beer to customers from Heineken's perspective. It reflects a situation of direct delivery from Heineken NL to a supermarket by a carrier controlled by Heineken UK. Heineken UK

[1] The ITAIDE project is funded by the 6th Framework IST Programme of the European Commission, for further information see www.itaide.org.

arranges transport between the Netherlands and the UK to the supermarket, which is visualized by one carrier. In practice the carrier might be a forwarder that arranges transport to a Dutch port, sea transport to the UK, and transport to the supermarket, which adds complexity to the BeerLL service system. Excise payment to UK Tax is by the UK retailer upon reception of the beer.

Other value chains are also implemented in practice, supported by business process models of each organization in the logistic chain. One value chain is for instance that beer is already physically located in the UK for delivery to a supermarket, without excise payment. The containers with beer are stored in a warehouse under customs regime, a so-called excise warehouse [20], until the beer is actually upon order transported to a retailer and sold for consumption. Excise has to be paid by the UK retailer to the UK Tax office when a container with leaves the excise warehouse. Heineken has to provide the evidence to Dutch Tax that excise has been paid in the UK. Upon request, Dutch Tax can ask Heineken to provide the necessary documents for manual validation against a previously made declaration.

Whereas the total number of value chains from Heineken's perspective is over five, selling beer in the Netherlands (two value chains), selling beer in another EU Member State (two value chains, and selling beer outside EU, a logistic service provider acts as a 'hub' in many value chains. A modelling technique needs to be flexible to support these requirements.

A data structure has been developed by ITAIDE to support all data exchange in the BeerLL. The data structure is based on the Core Component Technical Specification (CCTS, [5] and [6]) and constructed in accordance to the UN/Cefact Modelling Method (UMM, [2]) based on the Unified Modelling Language (UML). UMM consists of a number of templates that need to be filled for modelling cases like the BeerLL. The main focus of these templates is to construct business models of each organization supporting a particular logistic scenario. The templates do not encompass business rules for the selection of a value chain. Core Components are standardized data types that can be simple or aggregated. 'Street' and 'date' are examples of simple types; 'address' can be defined as an aggregate component consisting of 'street', 'number', 'ZIP code', and 'city'. For customs purposes, concepts like shipper (Heineken NL in the BeerLL) and consignee (Heineken UK) need to be modelled. The concept 'shipper' refers to an 'address' component and is extended with the concepts 'identifier' and 'name'. The latter concepts are attributes of the UML class 'shipper'.

2.2 Requirements to a formal model

This section lists a number of requirements derived from the BeerLL for a formal method. Basically, a formal method needs to support both data and process aspects. Data aspects should support the following requirements:

- Reference to other data structures like the core components should be supported. Applicable core components should not be adjusted to cater for requirements of a particular business domain to allow interoperability.
- XML Schema of interactions between organizations should be directly derived out of data structure. It allows consistency of semantics across all XML Schema, which improves interoperability within logistic chains. This latter requirement was not supported in modelling the BeerLL case. Notice that in

international trade and logistics, Electronic Data Interchange is still widely used and thus also needs to be supported.
- Alignment of different data structures, which is solved in the BeerLL by composing a common data structure.

Process aspects of the BeerLL refer to modelling interaction sequencing of all possible value chains. There are two relevant issues that need to be supported by a formal modelling method:

- It should support the concept of business rules that govern the selection of an appropriate value chain for a particular customer order, i.e. the usage of a warehouse or the direct pull of beer containers from the producer, Heineken NL.
- It should support interleaving of interactions to allow for instance the reception of a planning from a carrier and a delivery instruction from Heineken NL by Heineken UK in an arbitrary order.
- Exceptions that occur in logistics should be supported, i.e. exceptions that arise due to accidents and damage.

As part of the BeerLL, processes are redesigned with UMM to safeguard changes. These changed processes are not implemented. Instead, the common model is implemented in a gateway linking to internal systems and supporting web services for data access based on that common model.

These requirements are all combined under the umbrella 'autonomy'. In a multi-actor service system, each actor is autonomous in decision making, business process configuration, semantics, and its supporting IT. A formal method should be able to support autonomy.

3 Web Service Modeling Ontology

Service Oriented Architecture (SOA, [7]) stems from integrating applications and is the main IT paradigm supported by open standards as underpinning technology for service systems. Organizations can be integrated, based on the description of their externally observable behaviour, without the need for knowledge of their internal functioning. Thus, SOA seems to be applicable for modelling a multi-actor service system.

Most often, SOA refers to open standards for the technical representation of services, i.e.. Web Service Definition Language (WSDL, [7]) and XML Schema (XML: eXtensible Markup Language). It has been widely accepted that these open standards lack semantics and behavioural aspects [11]. There are various initiatives for adding semantics to services, namely OWL-S (Semantic Markup for Web Services [8], [9]), SAWSDL (Semantic Annotations for WSDL and XML Schema [10], [11]), and WSMO (Web Service Modelling Ontology [1]). We will investigate whether or not WSMO as one of these developments is able to model service systems. Additionally, conceptual approaches have been defined for multi-actor service systems, e.g. COSMO [**Error! Reference source not found.**]. It is not feasible to discuss all relevant modelling methods and concepts. Therefore, this contribution is limited to WMSO. This section gives a brief overview of the main elements of Web Service Modelling Ontology (WSMO) and presents observations regarding its usability of WSMO.

3.1 Main elements of WSMO

WSMO is based on the theory of Abstract State Machines (ASM, [14]). In ASM a real world system is modelled by its state space and a set of transition acting on this state space. Transitions will give a new state formulated by post-conditions, if certain conditions are met (pre-conditions). Pre- and post-conditions can refer to the state space and external events that change the state space.

From a service perspective, a proper set of transitions needs to be discovered to meet customer requirements. To discover these transitions, the *goal* of a customer and the *capability* of a service provider have been introduced. Both goal and capability are defined by pre- and post-conditions, and additionally by assumption of the world outside the state space and possible effects by the transition on that outside world. After discovery of a capability, the proper transitions can be executed. These transitions are defined by the *interface* of a service with its *choreography*.

Additionally, the following design principles are supported by WSMO (see also[1] and [12]):

- Unique identification of resources and Namespace.
- Description of resources and data are ontology-based (the state space)
- Each resource is described separately to reflect the distributed nature of the web (decoupling).
- User requests can be formulated independent of service provision (goals and capabilities).
- Mediation addresses the issues of interoperability of heterogeneous resources.
- A web service is a computational entity and a service the actual value provided by invocation of a web service. Thus, a web service is the abstract specification of a service.

These design principles are supported by the following concepts:

- *Information semantics* to specify the state space by means of ontology. Ontology consists of concepts, associations, rules (called axioms in WSMO), and instances (see also [13]).
- *Functional semantics* of services for the purpose of service discovery are specified by capability. By mediation, a goal can be matched to a capability.
- The actual *behaviour* of a service is expressed by choreography offered across and interface.
- *Non-functional properties* that are constraints to services. They cannot be expressed in WSMO.
- *Grounding* WSMO service specifications to WSDL message types and operations [25]. Grounding provides functionality for transformations from XML data to ontology instances and vice versa, and references to WSDL messages.
- *Mediators* resolve interoperability mismatches between different elements, e.g. between two services. They solve heterogeneity problems.

Most examples of services modelled by WSMO are fairly simple, e.g. the booking of a trip that requires booking information (pre-condition) and results in a reservation (post-condition). The assumption for this capability is that a valid payment method is used and the trip has been paid for by charging the account linked to the payment method for the required amount (effect).

3.2 Observations

This part of the paper contains a number of observations that have implications to usability of WSMO for modelling service systems like the BeerLL. Ideally, functional semantics and behaviour are related. The pre- and post-conditions of a capability should relate to the state change invoked by a choreography. Currently, they can be specified independent of each other, which can lead to inconsistencies. This is a simple issue that needs to be solved in a workbench like WSMO studio.

A more important issue is the support of both choreography and orchestration by transition rules, which is inherent to the application of ASM to services. It implies that choreography of a service expresses not only the input and output state of that service, but can consist of various internal states specifying an orchestration of that service. Each individual transition rule of choreography will give another internal state. Depending on that particular internal state, one or more other transition rules of 'choreography' can be executed. The internal states are expressed by the condition and the effect of a transition, respectively. This issue directly relates to the application of ASM to web service modelling.

We observe two issues with respect to internal states. The first issue is of determinism relevant for machine-to-machine communication in multi-actor service systems. Sequence of transition is implicit. Two transition rules are executed sequentially, if the effect of the first gives a state that fits the condition of the second. Transition rules can also be executed in parallel, but parallelism can not be expressed separately. One could for instance specify two transition rules with identical conditions. In WSMO it is not determined which transition rule will execute. From an abstract perspective they will be executed both at the same time, which can lead to unexpected behaviour by a system initiating those transitions. The choice to execute one of two or more transition rules with identical conditions is said to be non-deterministic. Another way to cater for this problem is to define one transition rule encompassing the execution of two or more parallel ones. Thus the condition is only mentioned once. Other operators like 'join' are implicit in case two or more transition rules alter the internal state in such a way that a condition for one other transition rule can be met.

The second issue is a usability issue. A designer is not able to graphically design a flow of transition rules by drawing the internal states connecting those transition rules. Relations between transition rules are implicitly specified by the state of the shared information space.

In the past, a lot of effort and research is spent on developing formal specification methods [22]. Determinism is one of the challenges that need to be solved in those methods. Adding the notion of 'time' is a means to introduce determinism.

4. Application of WSMO to the BeerLL

We have identified a number of requirements to a formal method like WSMO to support the modelling of service systems like the BeerLL. Firstly, the support of BeerLL data requirements by WSMO are investigated and, secondly, the process requirements. We present our findings in the following section. Basically, we have transformed a simplified class diagram of ITAIDE in WSMO ontology and specified

a capability with its choreography. In our examples, we do not include aspects in this specification that are not relevant, e.g. non functional properties of a concept, an attribute, a web service, etc. These non functional properties refer to Dublin Core elements like creator, publisher, etc. [16]. From a modelling perspective, non functional properties are of importance with respect to maintenance of (parts of) ontology.

4.1 Data requirements: the ITAIDE ontology

A simplified class diagram of the BeerLL can be expressed as ontology. It specifies the data requirements of the complete service system and is therefore not limited to a particular organization in that service system.

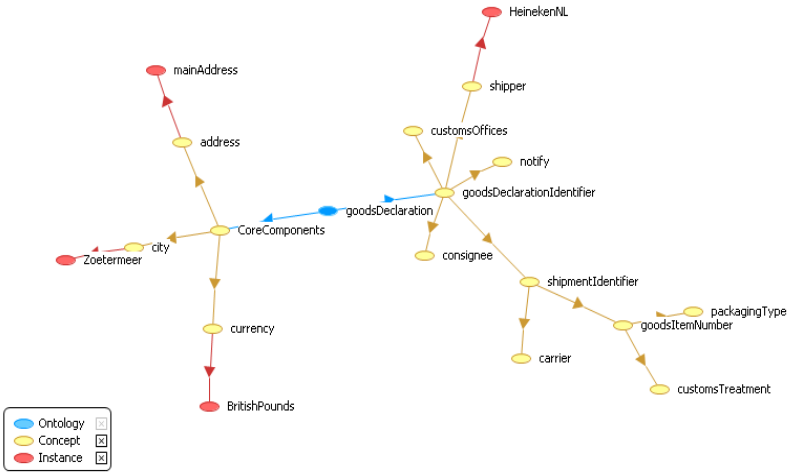

Fig. 1. ITAIDE ontology (modelled in WSMO Studio)

We tried to create several layers of ontology to fully accommodate the core component approach. The ontology, called ‘goodsDeclaration’, reflects the ITAIDE class diagram. Basically, it consists of two parts. The first part reflects the core components that are the basis for specifying the ITAIDE class diagram. As basic core components we constructed currency codes with a limited list of codes and city names. These basic concepts can be used to specify particular code values. Each instance of those concepts has a particular name and a code. Relations between instances like a city name like ‘Zoetermeer’ and its code are expressed by a value of the attribute ‘code’ of the concept ‘city name’. The association between city name and code can be made symmetric, meaning that a code has also a name. The concept address assigns the concept ‘city’ as the range of an attribute. Thus, an address can be constructed based on a particular city name, street and number and postal code. The postal code could be made such that a list of allowed postal codes per country is available.

The other part of the goodsDeclaration consists of an identifier with details specified by attributes and sub concepts. We have chosen to use an identifier as a basic concept for representing the classes in the ITAIDE class diagram, e.g. 'shipmentIdentifier' is a concept with particular attributes and 'packagingType' identifies the packages. The following examples show the specification of the concept 'shipmentIdentifier'.

```
Concept shipmentIdentifier subConceptOf
goodsDeclarationIdentifier
        departureDate impliesType _dateTime
        arrivalDate impliesType _dateTime
```

The concepts carrier, shipper, consignee, and notify refer to the core concepts. Each instance of these concepts represents a name, e.g. the instance 'Heineken' is the name of a shipper. Reference to the core concepts is by adding an attribute 'address'. In this particular example, the shipper Heineken is represented as:

```
instance HeinekenNL memberOf shipper
     address hasValue mainAddress
     shipperIdentifier hasValue "HE_NL"

instance mainAddress memberOf address
     city hasValue Zoetermeer
     postalCode hasValue "1111 HE"
     streetAndNumber hasValue "mainStreet_14"

instance Zoetermeer memberOf city
     code hasValue "ZTM"
```

Another relevant aspect of WSMO is 'axiom'. An axiom expresses integrity rules that are applicable to instances of concepts and their attributes. The ITAIDE class diagram does not contain integrity rules. However, one could envisage the following integrity rule (in wording): a TREC device can not be attached to two containers at the same time. To add this particular axiom, the 'goodsDeclaration' ontology needs to be extended with the concept 'TRECdevice' that can have an association with 'packagingType'. The TRECNumber itself is an attribute of 'packagingType'. In addition, to this axiom, one could state an axiom that TREC devices are only managed by trusted authorities with value TDA (TREC Device Authority). This axiom also needs further extension of the ITAIDE ontology (?x represents a variable).

```
axiom TRECDeviceManagement
     definedby
        ?x memberOf TRECDevice
             Implies ?x[issuingAuthority hasValue TDA]
```

Note that the concept 'location' for tracking and tracing purposes of a container is not included in the figure. We have included location in the ITAIDE ontology. 'Location' can be defined in different ways, e.g. as an 'address', 'city', and 'coordinate'. In case 'address' or 'city' is chosen as a location, an entry and departure date and time needs to be included to trace container movements. In case 'coordinate' is the means to locate a container, a coordinate may have to be transformed into an address or city, depending on requirements of the role interested in container

movements. Other enhancements to the ITAIDE ontology model are for instance core components like packaging types and including additional concepts and attributes like shown in the original class diagram. Also, the physical status of a container can be included in the ontology to produce temperature settings at certain dates and time.

Additional rules (axioms in WSMO) can be included in the ITAIDE ontology to safeguard that authorised opening of a container by removing a TREC device is only allowed at the final destination or the actual temperature setting must be between minimum and maximum allowed temperature values between place of departure and the final destination. Web services can be triggered in case one or more of such axioms are violated.

4.2 Process requirements

The process part of WSMO is expressed by goals/capability and choreography. WSMO is intended to construct a formal model of web services of a service provider that can be invoked to meet a goal of a customer. It implies that WSMO is intended to construct a formal model of each individual service provider in a service system and not a complete service system. This adheres to one of the design principles of WSMO: decoupling to reflect the nature of services. Per service provider, a capability and interface needs to be modelled. These can all be part of one WSMO specification, since one WSMO model can encompass more than one capability.

We have identified process requirements of the BeerLL for support of value chains. It is relatively easy to specify the goal of a customer of Heineken UK, but definition of Heineken's capabilities needs further consideration. Goal mediation relates to a value chain. The selection of a value chain by Heineken UK is based on the goal of a particular customer in terms of for instance delivery time, the customer status as perceived by Heineken UK (e.g. outstanding payments and returning customer), available stock in a warehouse versus required volume, and an allocation mechanism for assigning volumes of beer to meet simultaneous customer goals. Since value chain selection encompasses rules and access to other data, a capability needs to be formulated for those aspects that value chains have in common and are not derived from other services. Such a capability consists of transitions for including allocation services and accessing other relevant data. Variations of this standard capability are possible, e.g. Heineken may decide to specify particular capabilities for loyal customers. We will not specify this capability formally, since it is easy to define, but focus on the other process requirements of BeerLL. These need to be supported by the concept 'interface'.

5 Evaluation

This section gives an evaluation of the requirements of BeerLL to a modeling technique. Data -, process requirements, and autonomy are discussed.

5.1 Support of data requirements

We conclude that ontology as supported by WSMO is able to meet data requirements for service systems like BeerLL. The tools, WSMO Studio, do not yet fully support the functionality offered by ontology languages like OWL, since layers of ontology can not be supported by the tool. An issue is grounding of ontology to WSDL and XML Schema. If WSMO is considered for formally modeling a service system, the WSDLs need to be derived from choreography. This part of the grounding is not clearly described. Grounding to XML Schema in WSMO is the transformation of ontology to one XML Schema. However, the rules can not be represented in an XML Schema. Real life service systems like BeerLL therefore consist of an XML Schema for each interaction type, which means that an XML Schema needs to be derived from a choreography and ontology. Another way would generate an XML Schema and a set of rules in a rule language, which are referred to by an XML document. This needs further research.

5.2 Support of process requirements

We have identified three process requirements, that we will discuss seperately. The first process requirement is the support of business rules for selection of a value chain. These rules are partially implemented in allocation mechanisms, but additionally they have to be implemented on top of a WSMO model by using policy languages like WS-Policy [31]. They are not part of the model itself and it is not clearly defined how these policies can govern the execution of transition rules. They are said to be incidental details for service execution [31].

The second process requirement is that of interleaving of interactions that allows the reception of interactions in an arbitrary order. Basically, a formal method like WSMO offers a set of transition rules that can be executed only depending on preconditions of a transition rule, and thus support interleaving.

The third process requirement is the derivation of XML schema for each input or output formally specified in WSMO choreography. WSMO supports so-called grounding of in- and output of transitions to WSMO operations. Whereas WSMO considers an information space represented by ontology and transition rules consider the state of this information space, the relation between XML schema for web service operations and the WSMO information space is specified by transformation between XML schema elements and the ontology of the information space. A direct relation between XML schema elements and ontology is not required by WSMO, which allows WSMO to operate on top of existing web services specification. Thus, WSMO can be used independent of any web service specifications, which makes it difficult to derive those specifications out of a WSMO model.

5.3 Autonomy

One of the basic design principles of WSMO is to describe each resource separately to reflect the distributed nature of the web (decoupling). Each service provider offers

a service with its particular semantics and choreography. The semantics and choreography of a customer will be different. This inherent difference can be solved in various ways:

1. A mash up kind of way in which a customer includes the services of a service provider in its semantics and transitions, because there is a need to do business with that service provider (e.g. legal or economic need).
2. Runtime mediation of semantics and choreography of services. We have not yet found any reference that this type of mediation is supported by software. In practical situations, so-called intermediates like forwarders offer this functionality.
3. Mediation via a common model specifying semantics that potential customers and providers have in common and its related choreography in terms of what we would call collaboration protocols. These common models have to be organization independent. Common models are defined once and each organization can map its semantics and choreography to these common models.

Whereas the first solution requires many transformations, the third solution requires far less. Once the number of services and customers grows, and the number of different collaboration protocols decreases, a common model approach requires fewer transformations than a mash up approach. Further research is required to both approaches from a cost/benefit perspective.

6 Conclusions and further research

The evaluation shows that ontology is useful for modeling semantics in service systems, although grounding to technical solutions like WSDL and XML Schema needs further consideration. From a process perspective, a modeling technique like WSMO seems to be applicable, but needs to consider choreography and orchestration as separate issues. A graphical user interface, logical operators and 'time' may offer solutions to non-determinism. Operators are already defined by other formal methods like process algebras [24]. These basic control-flow patterns are also implemented by business process modelling languages like BPMn [23]. Since languages like BPMn and Petri Nets are supported by graphical tools and they support workflow patterns, these might be considered as formal methods for modeling service systems. Whereas BPMn 1.0 does not have a formal basis, Petri Nets have (see also [19]). A weakness of Petri Nets is its support of data modeling. It considers tokens, but is not concerned about the underlying semantic model of the tokens in the net. There are extensions to Petri Nets to include XML Schema [27]. Workflow control-patterns are specified using timed, coloured Petri Nets [32].

From an autonomy perspective, the most important requirement, different mediations approaches need further study. Cost/benefit analysis of these approaches also needs further study.

Additionally, we think concepts in relation to modeling techniques need further research. Existing modeling techniques encompass concepts, but those concepts don't seem sufficient to model service systems at business level.

References

1. D. Fensel, M. Kerrigan, M. Zaremba (eds.), Implementing Semantic Web Services – the SESA framework, Springer-Verlag, 2008.
2. UN/CEFACT - United Nations Centre for Trade Facilitation and Electronic Business, UN/CEFACT's Modelling Method – Base Module version 1, 2006.
3. J. Heineke, M. Davis, The emergence of service operations management as an academic discipline, Journal of Operations Management 25 (2007) 364–374.
4. J. Spohrer. and S.K. Kwam, Service Science, Management, Engineering and Design (SSMED) – An emerging Discipline – Outline and references, International Journal on Information Systems in the Service Sector, May 2009.
5. ISO 15000-5:2006, Core Component Technical Specification, draft version 2.2.
6. W.J. Hofman, EDI, Web Service and ebXML, communication in organisation networks, UTN, 2003 (in Dutch).
7. T. Erl, Service-Oriented Architecture – concepts, technology, and design, Prentice Hall, 2005.
8. OWL-S, Semantic Markup for Web Services, W3C member submission, 2004.
9. J.Scicluna, C.Abela, and M.Montebello**,** Visual Modelling of OWL-S Services, Proceedings of the IADIS International Conference WWW/Internet, Madrid, Spain, October 2004.
10. SAWSDL, Semantic Annotations for WSDL and XML Schema, W3C Recommendation, 2007.
11. K. Sivashanmugam, KunalVerma, AmitSheth, John A. Miller, Adding Semantic to Web Service Standards, ICWS 2003.
12. C. Feier, D. Roman, A Polleres, J. Domingue, M. Stollberg, and D. Fensel, Towards Intelligent Web Services: The Web Service Modelling Ontology (WSMO), in Proceedings of the International Conference of Intelligent Computing (ICIC'05), 2005.
13. J. Davies, R. Studer, and P. Warren, Semantic Web technologies – trends and research in ontology-based systems, John Wiley & Sons, 2006.
14. E. Börger: "High Level System Design and Analysis Using Abstract State Machines", Proceedings of the International Workshop on Current Trends in Applied Formal Method: Applied Formal Methods, p.1-43, October 07-09, 1998
15. J. Scicluna, A. Polleres, and D. Roman (editors), D14v0.2 Ontology-based choreography and orchestration of WSMO services, WSMO final draft February 3rd 2006.
16. Dublin Core Metadata Element Set, Version 1.1, 2008-01-14, www.dublincore.org.
17. X. Wang, T. Vitvar, V. Peristeras, A. Mocan, S. Goudos and K.Tarabanis, WSMO-PA: Formal specification of Public Administration Service model on Semantic Web Service Ontology, Proceedings of the 40th Annual Hawaii International Conference on System Sciences, 2007.
18. H.M.W. Verbeek, A.J. Pretorius, W.M.P. van der Aalst ... [et al.], Visualizing state spaces with Petri Nets, Eindhoven : Technische Universiteit Eindhoven, 2007.
19. Aalst, W.M.P. van der, Beisiegel, M., Hee, K.M. van, König, D., Stahl, C. (2007). A SOA-based architecture framework. International Journal of Business Process Integration and Management, 2(2), 91-101.
20. http://www.douane.nl/zakelijk/ accijnzen/en/accijnzen-07.html.
21. http://www.zurich.ibm.com/csc/ process/securetradelane.html.
22. http://formalmethods.wikia. com/wiki/Formal_methods.
23. Business Process Modelling Notation (BPMN), version 1.2, january 2009.
24. E. Brinksma, On the design of extended LOTOS – a specification language for Open Distributed Systems, 1988.
25. D24.2v0.1. WSMO Grounding, WSMO Working Draft 27 April 2007.

26. R. van der Toorn, Component-Based Software design with Petri Nets – an approach based on inheritance of behaviour, Ph.D. thesis, Eindhoven University of Technology, 2004.
27. http://www.yawl-system.com/.
28. http://en.wikipedia.org/wiki/Service_system.
29. Lankhorst M., et.al., Enterprise Architecture at work, Springer, 2005.
30. Lovelock, Wirtz, "Services Marketing: People, Technology, Strategy," 6/e, Upper Saddle River NJ: Prentice Hall, 2007.
31. Vitvar T., Zaremba M., Moran M., Zaremba M., and Fensel D., SESA: Emerging Technology for Service-Centric Environments, IEEE Software, vol 24, no. 6, November/December 2007.
32. Russel N., Hofstede A.H.M. ter, Aalst W.M.P. van der, Mulyar N., Workflow control-patterns – a revised view, *BPM Center Report BPM-06-22* , BPMcenter.org, 2006.
33. Tan Y.H., Hofman W.J., Gordijn J., Hulstijn J., A framework for the design of service systems, *The Science of Service Systems*, Springer, 2010 (to appear).
34. Quartel D., Steen M.W.A., Pokraev S., Sinderen M. van, A condecptual framework for service modelling, EDOC 2006, 319-330.