Service Innovation and Service Realisation with VDML and ServiceML

Arne J. Berre, Yannick Lew, Brian Elvesæter
Networked Systems and Services
SINTEF
Oslo, Norway
arne.j.berre@sintef.no, yannickl@ifi.uio.no,
brian.elvesater@sintef.no

Henk de Man
Research and Development
Cordys
AC Putten, The Netherlands
hdman@cordys.com

***Abstract*—It is shown how a service-oriented and model-based approach for Enterprise Architecture and Enterprise Engineering can provide agile support for the different abstraction and modeling levels from business model and service innovation to cloud-based service realisation. Business model innovation is supported with a basis in a business model framework with six views, where each view is supported by a corresponding diagram from the Value Delivery Modeling Language (VDML). Service innovation is supported by the Service Modeling Language (ServiceML), which shares the core collaboration models of VDML for role modeling and value networks, according to the five views of the AT-ONE service innovation method. Service realisation is supported by further transformation to SoaML and a model-based service execution platform. This approach presents a framework using the emerging OMG standard language VDML together with ServiceML for service design and engineering, relating value models, process models, user interface and interaction flow models, and service architectures and service contract models.**

Keywords— Service Innovation, Service Design, Service Modeling, Service-oriented Enterprise Architecture, VDML, SoaML, ServiceML, AT-ONE

I. INTRODUCTION

This paper presents a platform for service innovation with service realisation using domain specific languages such as the Value Delivery Modeling Language (VDML) [1] and the Service Modeling Language (ServiceML). The approach is viewed as part of a model-based enterprise architecture [2]. Enterprise goals and strategies are related to supporting initiatives for business model innovation and usage of the BMM (Business Motivation Model) [3] for the identification of business vision and goals, in conjunction with customer requirements, which are related to strategies supporting processes and services. Our focus is on the elaboration and specification of innovative services which can be further realised in an appropriate service oriented infrastructure, such as the Cordys cloud-based service platform [4].

This is provided through the NEFFICS platform [5] which combines an open innovation social media platform with a business modeling and operations platform. Together these provide a foundation for cloud-based open business model innovation, process innovation and service innovation for networked enterprises. *Business model innovation* is supported through a business model framework with six views. Each view is supported by a corresponding diagram from the Value Delivery Modeling Language which is an emerging standard proposed to the Object Management Group (OMG) focusing on value transfers in role collaboration and business activities. VDML has now been proposed for further standardisation within the OMG. The Business model provides the context for further Service innovation and Process innovation. *Process innovation* is supported by VDML activity diagrams with options for mappings to Business Process Model and Notation (BPMN) [6] and Case Management Model and Notation (CMMN) [7]. *Service innovation* is supported by ServiceML which shares the core collaboration models of VDML for role modeling and value networks. ServiceML extends the Business-SoaML language [8], which is based on the Service oriented architecture Modeling Language (SoaML) [9] for business architectures, in order to model the five views of the AT-ONE method [10] to support service innovation.

The remainder of this paper is structured as follows: in Section II, we provide a description of the functionalities offered by the NEFFICS platform. Section III introduces the enterprise innovation framework with the business model innovation cube with VDML and service innovation with ServiceML. Section IV introduces ServiceML and describes service innovation. Section V discusses VDML and ServiceML alignments. Section VI presents the business operations and execution environment of the NEFFICS platform in an enterprise setting. Section VII discusses our contribution with respect to related work. Finally, Section VIII concludes this paper and outlines further work.

II. A MODEL BASED ENTERPRISE ARCHITECTURE APPROACH

There are a number of Enterprise Architecture frameworks with a multi viewpoint approach for abstraction levels and perspectives (Zachman, RM-ODP, TOGAF and UPDM). The ASD (Agile Service Development) framework [11] is one that also includes user interaction as a separate perspective in addition to structure, function, coordination, information and

extra functional aspects. The abstraction levels goes from requirements to design, implementation and infrastructure. Fig. 1 illustrates a proposal for how different recent modeling languages can be used to support the various perspectives. Each of these comes with associated development practices. The suggested modeling languages are all existing standards from OMG, OASIS or W3C, or emerging languages in this context. These languages are well suited for their particular perspectives, but currently not designed to work harmoniously together, and sometimes also contain more concepts and complexity than needed for a certain abstraction level. It has earlier been suggested to start an effort to create a coherent set of co-working languages with a foundation in the conceptual model of existing languages, and this is followed up on here.

The agility emphasizes strong interactions with users and practitioners in dynamically adapting the models to emerging user needs. The following shows a number of modeling languages which are suitable for a direct language support for the various perspectives from the Agile Service Development framework [11].

The use of languages has been divided between the various abstraction levels from business architecture and business models for context and goals through requirements and design to implementation and infrastructure. In this paper we focus in particular on the use of the new VDML language and the link to the ServiceML language, as it has been developed for the NEFFICS platform [12]. Accompanying the NEFFICS platform, we have the public NEFFICS Methodology Wiki (neffics.modelbased.net) which provides descriptive guidelines in the form of practices for how to apply and adopt the models, methodologies and tools provided by the NEFFICS platform and frameworks.

Interaction can be supported with user interface mockups, as has been illustrated in [4]; with mockups that also can be a basis for further generation of actual user interfaces through exchange languages such as BMM. Structure can be supported with VDML. Function can be supported in an agile way by user stories in the spirit of agile requirements engineering [5] with a potential further refinement into use cases. Coordination can be supported by business level BPMN (Business Process Model and Notation) [6]. Information can be supported by term definitions and relationships through ontologies, and representations in various graphical and textual forms as in ODM (Ontology Definition Metamodel) [7]. Business objectives can be supported by the BMM (Business Motivation Metamodel) [3], which focuses on identification of business vision and goals related to strategies with supporting processes and services.

Service Innovation is about establishing new or improving existing services. This is particularly relevant in the context of NEFFICS because the platform is enabling services that span the boundaries of enterprises. The NEFFICS platform enables service innovation from the users' points of view. This is complementary to the abovementioned process innovation perspective.

The following shows a number of modeling languages that are suitable for representation of the different design and implementation perspectives. Interaction can be supported through the emerging IFML (Interaction Flow Modeling Language) [7], which provides a platform independent approach for the specification of user interface dialogues. Structure with roles can be supported with VDML [1] and the role models being related also to roles in SoaML for provisioning of services and further to the role of sensors in

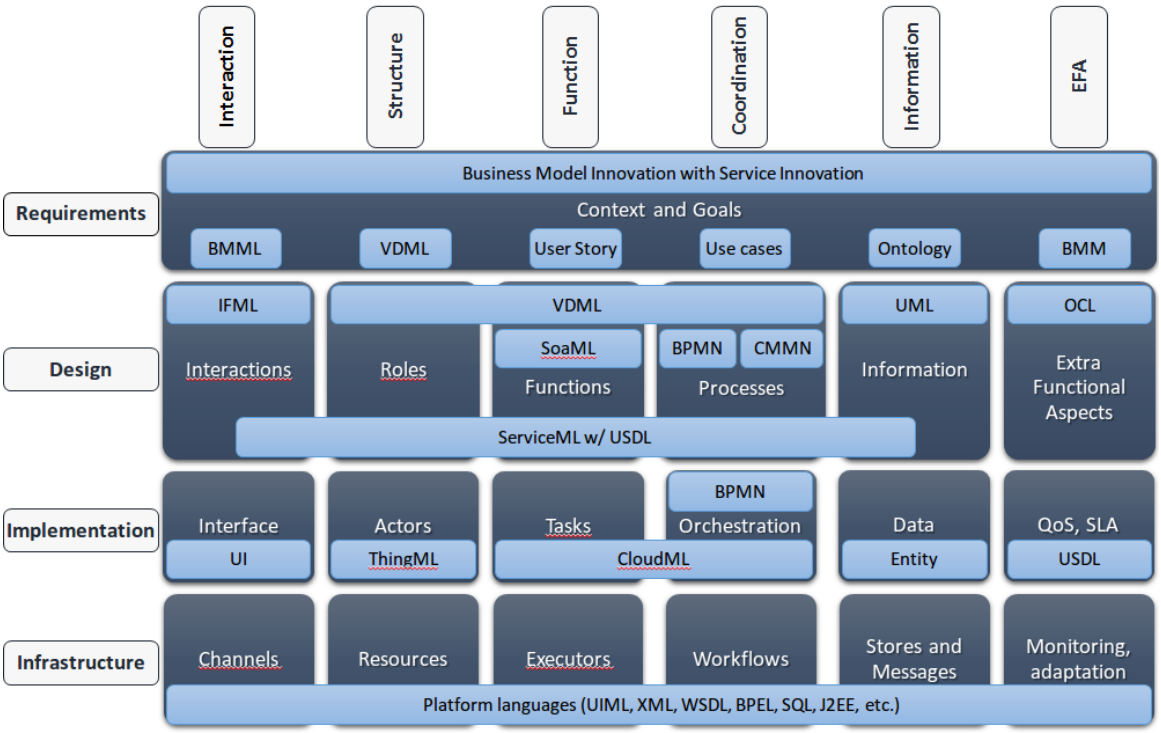

Fig. 1. A model based Enterprise Architecture framework

ThingML [8] which is a modeling language for embedded and distributed systems. Function with services can be supported by service descriptions in SoaML (Service oriented architecture Modeling Language) from OMG, and also be further supported by USDL (Universal Service Description Language) which is being studied in W3C [9]. This can be supported further with provisioning and deployment in Cloud environments using CloudML [8]. Coordination with processes can be supported by system level BPMN [7], and further mappings to BPEL and TOSCA [6] for implementation level support for services and cloud environments. Information can be supported by further modeling through UML enhanced class diagrams, and have further mappings to representations in databases and semantic technologies. Extra functional aspects can be supported by constraints specification in various constraint languages, such as OCL (Object Constraint Language) [7].

III. ENTERPRISE INNOVATION FRAMEWORK

Innovation in an enterprise setting occupies two major aspects which are namely innovation affecting the business model environment as well as the service platform being offered by the enterprise.

A. Business Model Innovation with VDML

A Business Operation Platform serves as the integrated component in the NEFFICS platform that supports modeling and model-based management of business and business operations. It is logical to extend modeling support in a business operations platform with support for model-based design and analysis of Business Models (BMs).

In NEFFICS, business models are considered to consist of six building blocks or dimensions, linked through relationships as a seventh dimension, and can conceptually be represented as a cube. The six dimensions of this "business model innovation cube (BMI Cube)" [13] [14] are: Customers, Value Propositions, Activities (the "value chain"), Capabilities (or Competences), Network Partners and Value Formulas.

This cube, together with diagrams for the different views, is shown in Fig. 2, where VDML provides a diagram for each dimension of a BM. Every business model item, as associated with a business model dimension, can also be represented on the corresponding VDML diagram, in such a way that navigation is supported from the business model diagram to the "underlying" VDML diagrams, whereby the business model item of concern is highlighted, and the VDML diagram represents how the business model item is related to other items. Thus VDML-based models provide a structured and detailed representation of business model items (according to the six dimensions) and their "relationships" (seventh dimension).

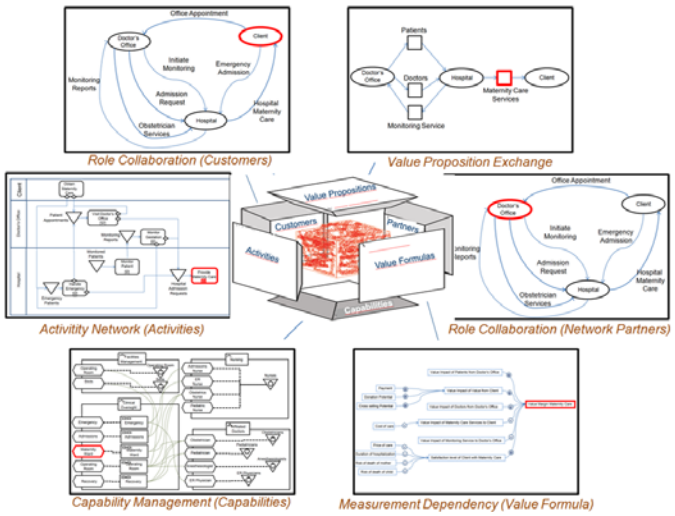

Fig. 2. BMI Cube with VDML diagrams

Note that this integration between the BMI Cube framework and VDML will bring advantages for managers and innovators, as well as for business designers (analysts, architects):

- It enables managers and innovators to abstract from business design and business architecture details. It will also, and eventually, provides them with a useful context to present measurements, ideas and challenges, that are related to underlying details.
- It will provide a useful way to guide business analysts and architects, whereby the various dimensions of a BM serve as "chapters" of business design "methodology". It will provide them with a broad understanding and context of the business and with a useful means to navigate business designs.

The BMI cube is intended to extend the VDML metamodel and notation with integrated business model metamodel and business model diagram as a result. Integration will involve integration of both model objects and notation (diagrams), with support for diagram-based navigation from business model diagrams to VDML diagrams (and vice versa), where appropriate.

B. Service Innovation with ServiceML

Service innovation provides a holistic approach to the development of innovative services by relying heavily on the way customers experience such services. It aims at improving the early stages of service innovation through the integration of design-thinking into a structured innovation process. Thinking about how existing services can be improved or how new ones can be created then becomes an integral part of the enterprise strategy with respect to the service portfolio planning and its relationship with customer demands. Thus, service innovation contributes to the implementation of a business plan that is geared towards the provision of innovative services.

The community of service designers tend to focus more on interactive workshop activities and informal "sticky notes" for their documentation, rather than using formal modeling languages. Based on the same "design-focused" approach, we make use of the AT-ONE method for service design and development. AT-ONE provides a valuable insight into service innovation by proposing to innovate a service in terms of five distinct lenses, namely (A)ctors, (T)ouchpoints, (O)fferings, (N)eeds and (E)xperiences with a solid foundation in the ways these five lenses work together to achieve specified service

design objectives. These diagrams form the basis of the *ServiceML* language which is further described in Section IV.

ServiceML comes with a number of creative techniques to foster the innovation of services. The requirements, software and service engineering communities could benefit well from integrating and adapting a number of service design practices into their practices portfolio, and the service design community would benefit from the possibility to more easily create prototype services for earlier and quicker user feedbacks. In particular, the creation of innovative services for the "future internet" mandates a fast and flexible design approach – including both a top-down and a bottom-up one.

IV. THE SERVICEML LANGUAGE

To support the service innovation capabilities being proposed in AT-ONE, the ServiceML language has been created. It is comprised of three different language packages [15], namely: (1) Business-SoaML which is a SoaML variant that is suited for business people while covering the architectural aspects of a service oriented architecture, (2) Light-USDL which provides interface descriptions for business-minded people and is based on the USDL (Unified Service Description Language) [16], and (3) Service Journey Maps which describe the service experiences from the customer's and service provider's points of views.

Fig. 3 shows the various ServiceML diagrams for the five lenses of the AT-ONE method, namely Actors, Touchpoints, Offerings, Needs, and Experiences.

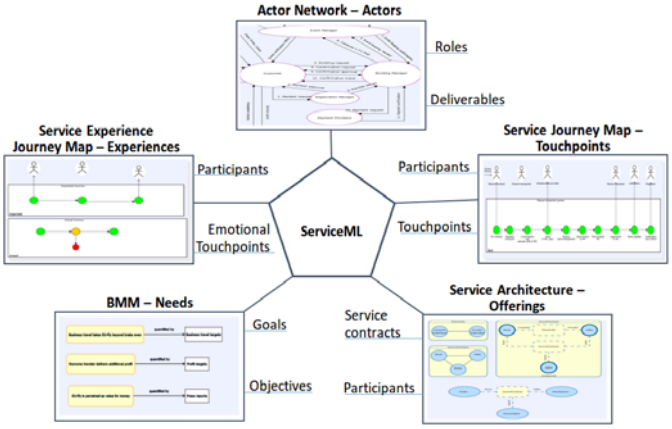

Fig. 3. ServiceML AT-ONE lenses

The ServiceML language is currently being applied to a few case studies with the cooperation of industry partners. The goal is to make use of ServiceML to create innovative services or improve existing ones. One such case study involves a hypothetical event booking company, called 'Concierge', which provides *ticket booking services* by acting as a middleman between event organizers and customers. The business environment parameters of Concierge are based on real-life settings and have been validated by our industry partners. The following sections provide an overview of the ServiceML AT-ONE models based on the Concierge example.

A. Actors with Role Collaboration and Value Networks

An actor network, in ServiceML, represents a collaboration of actors in a value network [17] consisting of roles as well as deliverable flows that are exchanged among them. Actors or participants are responsible to fulfil these roles and can represent real-life or logical entities. Deliverable flows are deliverables that are either tangible or intangible.

Fig. 4 illustrates the ServiceML actor or value network for Concierge. Five roles (represented by ellipses), namely Customer, Event Manager, Booking Manager, Registration Manager, and Payment Providers, are depicted along with a series of deliverables (represented by arrows) that are exchanged among them. Each deliverable flow can be labelled and can be annotated with a number to indicate the sequence of deliverable flow exchanges.

As shown in Fig. 4, a Customer typically starts the sequence by sending a "Member request" to the actor playing the Registration Manager role for Concierge. The latter verifies the request and issues a "Member approval" message back to the Customer. After performing the initial member registration, a Customer interacts with the Booking Manager when performing an event booking. The process continues based on the remaining deliverable flows that must be executed in the sequence such that the booking scenario is completed when payment for a booking order is confirmed and the customer receives a "Confirmation ticket" from the Booking Manager. Note that a deliverable flow without a sequence number indicates that it may be exchanged spontaneously – that is, it may occur without any prior event occurring. For instance, the Event Manager can, at any time, send an "Event notification (PEA)" message about latest events to a Customer without relying on any prior message interaction.

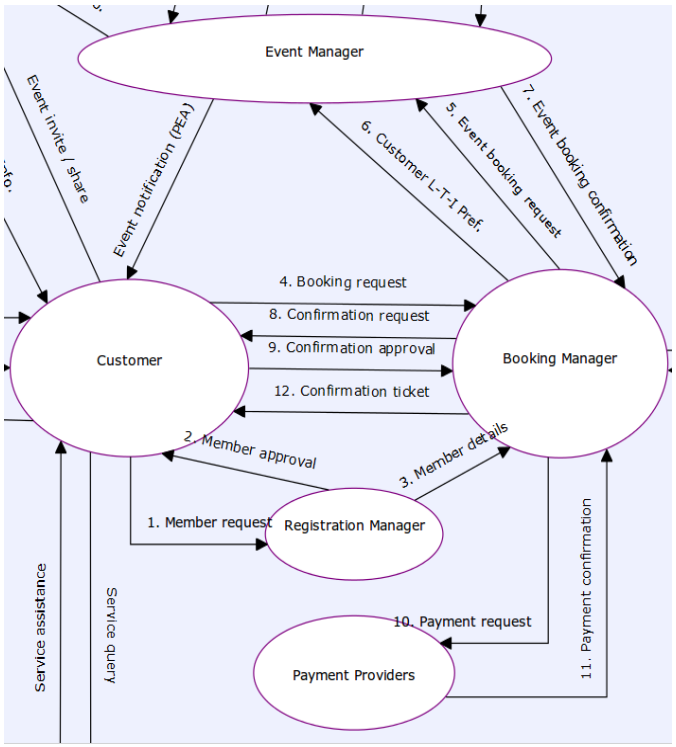

Fig. 4. ServiceML Actor network

The ServiceML actor network provides a valuable overview of all the essential stakeholders in an enterprise's network, along with their message interactions, in order to design and create services that revolve around these actors. As a matter of fact, the actor network serves as an input for the creation of the other ServiceML models.

B. Touchpoints with Service Journey Maps

A ServiceML Touchpoint model is based on the mapping of a customer journey [18] as experienced by an actor or participant of the service. A customer journey map depicts the typical flow of events or activities, encapsulated as a touchpoint, which a participant performs in a service execution as he/she encounters a physical or logical point of contact between a service provider and a consumer such as the service provider's website or newspaper advertisements. Instead of only representing the customer perspective in a customer journey map, we have adapted the latter to include viewpoints or journeys from other stakeholders forming part of the service provider's group. Our new model is called *Service Journey Map* and can show perspectives from both a service provider and a service user [19].

The ServiceML Touchpoint model for the Concierge example is shown in Fig. 5. Actors, which played roles previously identified in the ServiceML Actor model of Fig. 4, are represented by stick figures, while touchpoints or activities are represented by the green circles. An actor can either initiate a touchpoint (represented by open arrows) or be involved, through some inputs or outputs, in a touchpoint execution (represented by dotted lines). The touchpoint execution sequence is shown with the help of closed arrows linking one touchpoint to the next.

The ServiceML Touchpoint model of Fig. 5 depicts the member registration and event booking service journeys as experienced by a Concierge user belonging to the mass market category of consumers. Note that the level of touchpoint details depends on the nature of the services being modelled as well as the stakeholder viewpoints involved in the process. Typically, a service journey map can describe the set of touchpoints which make up one or more services that are derived from the list of services that are identified in the ServiceML Offerings model which is shown in Fig. 6. Moreover, the deliverable flows presented in the ServiceML Actor model of Fig. 4 serves as inputs to support the touchpoint execution scenario in Fig. 5.

Basically, Fig. 5 illustrates how a service consumer performs an event booking using Concierge's website by first having to register himself or herself as a member if required, and then, choosing an event (involves the Concierge Booking Manager to handle the ServiceML Actor deliverable flows numbered 4 to 9 as shown in Fig. 4), paying for the order booking (involves the Payment Provider to verify payment based on deliverable flows 10 and 11 in Fig. 4), and so on before providing feedback regarding the booking service to the Concierge customer helpdesk (not shown).

A service journey map helps a service provider to better understand how its services are being "consumed" by its customers, and, in so doing, be able to refine and optimize each touchpoint along the service journey. Therefore, a ServiceML Touchpoint model offers a great tool to engage stakeholders in providing service feedback so as to help improve/innovate the service journey that they are experiencing. We will later also see how user emotions can be introduced to identify "good" and "bad" touchpoints based on the degree of satisfaction when a user carries out a touchpoint activity thanks to a ServiceML Experience model.

C. Offerings with Service Architecture and Service Contracts

A ServiceML Offerings model provides an intuitive representation of services that have been agreed to be delivered by one stakeholder or participant to another. The participant offering the service is what we commonly refer to as being the "service provider", while the one receiving the service is called the "service consumer". We model the service offerings through the introduction of (1) a "Service Contract" concept which basically establishes a service delivery agreement between two participants and (2) a "Service Architecture" diagram which contains an overview of all participants and the service contracts binding them to each other.

Fig. 6 depicts a ServiceML Offerings model for our Concierge example. Service stakeholders or participants, which are derived from the ServiceML Actor model, are represented by ellipses while service contracts, which link one participant

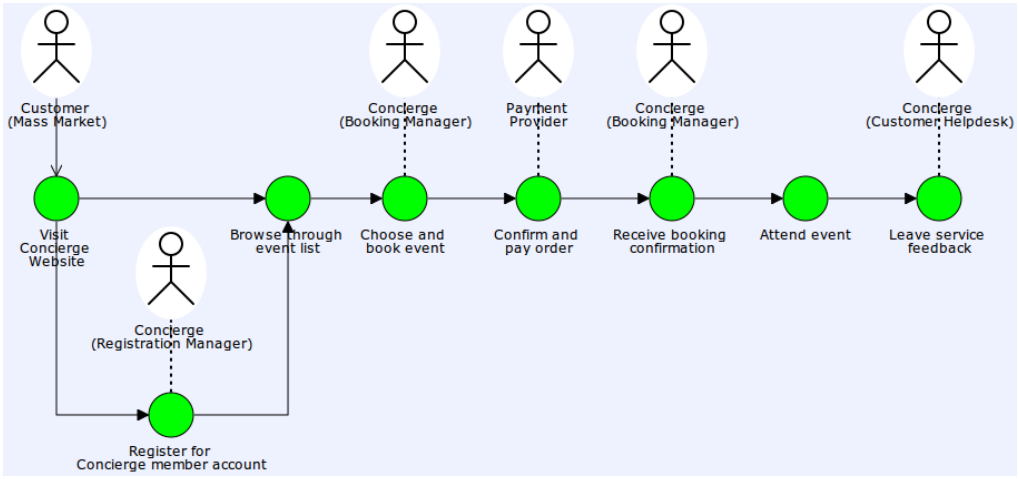

Fig. 5. ServiceML Touchpoints – Service Journey Map

to another, are modelled as the rounded, dotted rectangles. The dotted lines, connecting one participant to a service contract, can be labelled with the role name that the participant plays in the service contract agreement. Such a distinction is necessary since a participant can be involved in multiple service contracts by shifting the role he/she plays in the service interaction.

It is worth noting that each service contract is identified based on the deliverable flows that are exchanged among participants – the basic principle being that all participant interactions, involving one or more deliverable flows, should occur through *services or service contracts.* Thus, a service contract represents the implementation of one or more deliverable flows between two participants.

As Fig. 6 suggests, the overall layout of a ServiceML Offerings model looks similar to its related ServiceML Actor model. The five participants from the Actor model are again present in the Offerings model while five service contracts linking the participants describe the sets of deliverable flows that are exchanged between participants. For example, the "Member request" and "Member approval" deliverables now form part of the "Registration Service" between a customer and the registration manager with the customer playing the role of a "Customer" or "Consumer" and the registration manager taking up the role of a "Registration Agent" in the interaction. Similarly, the same "Customer" is involved in the "Event Booking Service", which takes care of the deliverable flows for event booking numbered 4, 8, 9, and 12 in Fig. 4, with the "Booking Manager" who plays the role of a "Booking Agent" in the service contract interaction.

Note that the service contract identification provides an initial understanding regarding the high-level services that must be provided between a pair of participants. By further refining the service descriptions for each service contract, new and,

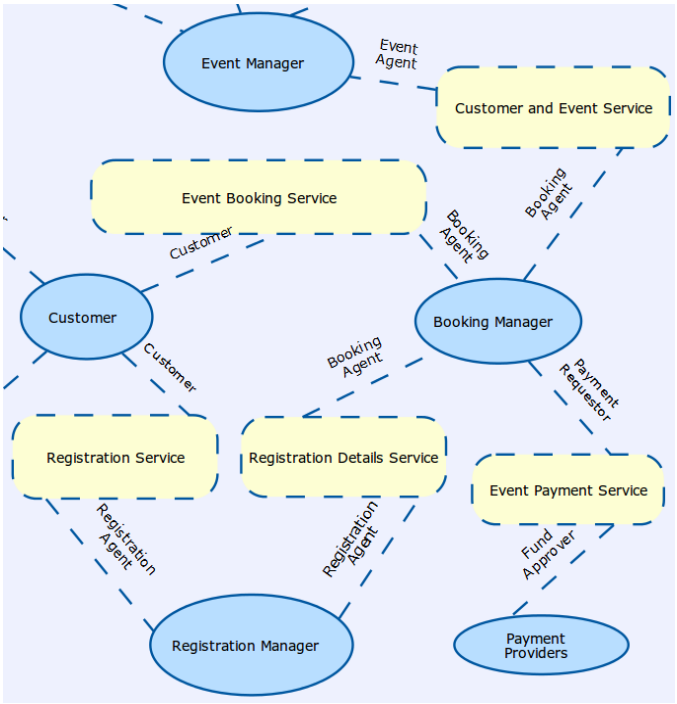

Fig. 6. ServiceML Offerings Model

perhaps smaller service units can be constructed as a basis for defining service interfaces through a *SoaML implementation* of the ServiceML Offerings' Service Architecture diagram.

Consequently, the ServiceML Offerings model presently provides an important tool for service identification and alignment with respect to the needs of customers and the enterprise resources required to implement these services while at the same time making sure that enterprise goals are being matched with those customer needs. The latter is discussed as part of the ServiceML Needs model which is described in the next section.

D. Needs with Goals and Objectives

The ServiceML Needs model is based on the Business Motivation Model (BMM) [3] and provides a description of customer needs with respect to the services being offered by a service provider. As such, customer needs are crafted based on functional and non-functional user requirements that typically constitute the underlying platform for building software systems in our present computing era.

Through the goal-objective approach proposed by the BMM, we model functional requirements as "goals" that customers desire to be reached while non-functional requirements are expressed as quantifiable "objectives" that are necessary to be attained in order to meet a specified goal. Therefore, a goal can have *multiple* objectives. This is clearly a worthwhile combination, from both the service provider's and customer's points of view, as we explain through the Concierge example shown in Fig. 7.

The goal-objective philosophy of a ServiceML Needs model is represented as a series of rounded rectangles for goals and classic rectangles for objectives linked together with open arrows. As shown in Fig. 7, a non-exhaustive list containing two goals, along with three objectives associated with each one, are described for our Concierge example. For instance, the bottom goal specifies a customer need to have an event booking service to handle event booking transactions. This goal is coupled with three distinct objectives which are related to non-functional aspects involving the security and performance requirements of the event booking web page.

It is interesting to note that the goal description is written using an "agile" approach for the specification of *user stories* that are linked to functional requirements. Likewise, the objectives are described based on the "who", "what", and "why" of their respective goals, and are quantifiable by mentioning appropriate metrics necessary to attain each objective.

Essentially, a goal-objective pair provides a medium to match customer needs with service offerings proposed by a service provider as shown in the ServiceML Offerings model of Fig. 6. By making sure that each customer need is catered for through the provision of at least a service contract and that each service offering can be matched with one or more customer needs, unnecessary or duplicate services can be eliminated and existing ones can be fine-tuned to better meet the demands of consumers.

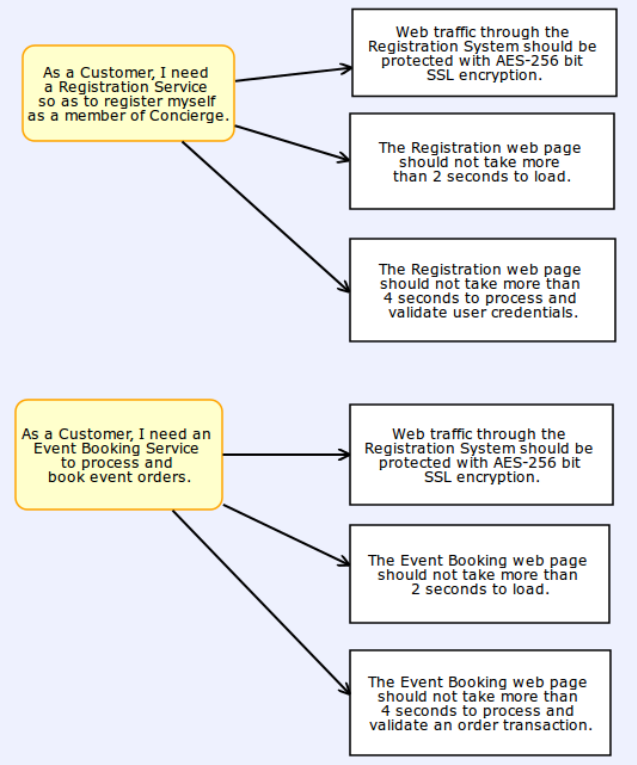

Fig. 7. ServiceML Needs Model

E. Experiences with Emotional Service Journey Map

Services are experiential in nature and service experiences can be designed and staged to achieve optimal results. An emotional service journey map or ServiceML Service Experience Journey model is basically a traditional service journey map, but with the addition of touchpoints that now portray the level of satisfaction a user experiences through his/her *emotional attachment* towards these touchpoints.

As pointed out in the *IV. B. Touchpoints with Service Journey Maps'* section, user experiences can be used to identify touchpoints that are considered to provide either a positive or negative experience to users based on their emotional attachments towards specific touchpoints. For example, if a consumer feels that performing a touchpoint activity is boring or too problematic, the latter can receive a negative feedback from that consumer.

On the other hand, a touchpoint can be attributed a good feedback if a service user enjoys performing it or feels a strong, positive emotional connection with the touchpoint. Note that service experience journey maps can be effectively performed by a consumer or a group of consumers, but the emotional rating for each touchpoint have to be weighted in and averaged based on all consumer feedbacks obtained.

Four levels of emotional experience for service touchpoints have been defined, namely: "Bad (Poor)" (represented by red circles), "Average" (represented by yellow circles), "Good (Normal)" (represented by green circles), and "Outstanding" (represented by blue circles). As shown by the ServiceML Service Experience Journey model for our Concierge's customer registration and event booking example in Fig. 8, mass-market customers have identified two "outstanding" touchpoints, three "good" ones, one "average" touchpoint, and two "bad" ones. For instance, customers have rated the registration touchpoint as providing an outstanding service experience while registering for a new customer account. However, browsing through the list of events and selecting one of them have proved to be difficult and have, thus, been assigned the red touchpoints for bad service encounters.

The touchpoint implications, in terms of changes to be made, for a service provider based on a service experience journey are summarized in table I. From a service provider's point of view, careful analysis of a service experience journey map can provide visual clues regarding the level of satisfaction that customers have towards service offerings. If a touchpoint or group of touchpoints receives negative feedback, the service provider must act by bringing forward changes to the service journey. This endeavour will ultimately have an impact on the other ServiceML diagrams. Therefore, it is appropriate to perform a change impact analysis on the various ServiceML models to determine the best approach when bringing in changes. This change impact analysis, along with the addition of assessment criteria that are necessary to determine the

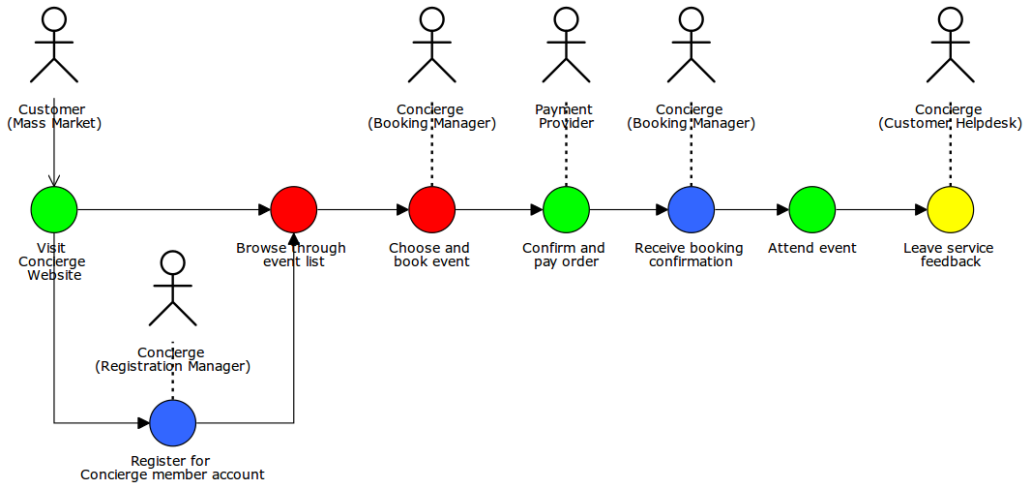

Fig. 8. ServiceML Service Experience Journey Map with Emotions

weight and impact of a negative feedback on touchpoints against the overall "success" of a service journey map, constitute further areas of research.

TABLE I. TOUCHPOINT IMPLICATIONS BASED ON USER EXPERIENCES

Emotion	Experience	Service Touchpoint Impact
Bad	Negative	Offers major problems to the user. Must be evaluated and improved.
Average	Slightly Negative	Offers a few minor problems to the user. Should be evaluated and improved.
Good	Positive	Offers a reasonably good service experience to the user. Can be evaluated and further improved if required.
Outstanding	Outstanding	Offers a service experience that surpasses user expectations. Can be evaluated and may provide a model to improve other touchpoints.

V. VDML AND SERVICEML ALIGNMENT

Service concepts in ServiceML have been aligned with VDML. This section provides a brief overview of the mapping of core VDML to ServiceML elements as shown in Fig. 9. Five core elements are described including 'collaboration', 'role', 'activity', 'value', and 'value proposition'.

A collaboration in VDML is a collection of several participants (in business networks, communities, and organizational units) interacting with each other for business purposes. In ServiceML, the notion of a collaboration is required to perform service journeys and to fulfil service contracts in a service architecture.

A VDML role performs activities defined in a service collaboration. Roles can be assigned to VDML participants in order to specify which actor or group of actors should fulfil the role and perform the set of associated activities. Similarly, ServiceML roles express consumer needs, provide/consume service offerings, design/express service journeys, and provide/receive value propositions.

A VDML activity is work performed within the context of an established service collaboration. A participant executes the activity and, in so doing, helps to create or consume 'value' as defined in VDML. From the ServiceML side, service journeys are translated into a set of activities to be carried out by participants.

A VDML assignment represents a role binding mechanism to be performed in the context of a ServiceML service journey.

A VDML top-level collaboration is a ServiceML service architecture. If the collaboration is between two roles, then a simple service contract is created; otherwise, a compound service contract (containing multiple, simple service contracts) is produced for collaborations involving more than two roles.

A VDML deliverable flow is used to identify ServiceML service contracts – the source and targets of the deliverable flows are mapped to roles involved in the collaboration.

Note that other less notable alignment concepts have not been mentioned, but these equally play an important part in any VDML to ServiceML and vice-versa conversion process.

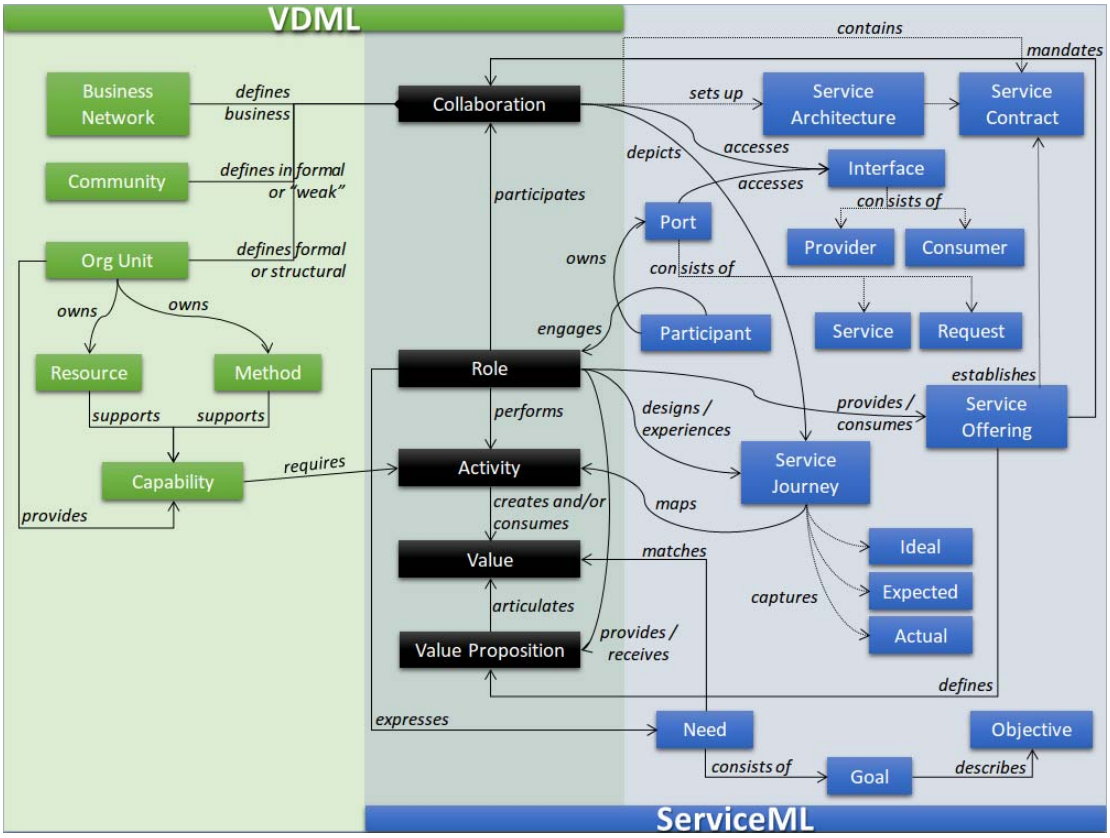

Fig. 9. VDML and ServiceML alignment

VI. From ServiceML to SoaML and Service Realisation

ServiceML can be adapted to work in an enterprise engineering setting consisting of business and IT elements designed to work in a coherent, integrated fashion. Beginning with enterprise goals and strategy, ServiceML provides models for the alignment of customer needs and service offerings. Enterprise resources, such as actors, assets, and IT support, can be analysed to work optimally based on the enterprise's actor network as well as the enterprise capability required to support service offerings. Finally, enterprise operations and execution rely on the proper running of services based on users' service experiences as detailed in a service journey map and a service experience journey map. Fig. 10 provides an overview of the ServiceML elements and their usage in an enterprise environment.

Enterprises adopting ServiceML for service development can either follow a top-down or bottom-up engineering process. The top-down method envisages a *constructive* approach by first identifying and then matching customer needs with service offerings. In this phase, enterprise planning is done to establish the goals and strategy of the business based on customer needs and services required to fulfil them. After that, actor identification takes place to build up the enterprise network required to support the provision of service offerings. Enterprise resources can be accounted for to provide the required capability to implement service offerings. Once the list of services has been finalized, the customer experience for each service journey can be evaluated and refined based on continuous customer feedback as part of the enterprise operations and execution duties. The *constructive* approach is more suited to be employed by new market players which do not yet have a solid service portfolio at their disposal.

On the other hand, the bottom-up method provides an *improvement* approach to service development since it is meant to be used by established enterprises which have set up a service portfolio and have customers using their services already. Therefore, using this approach, an enterprise analyses the customer experience for each service journey before studying its actor network and the set of customer needs matching its service offerings.

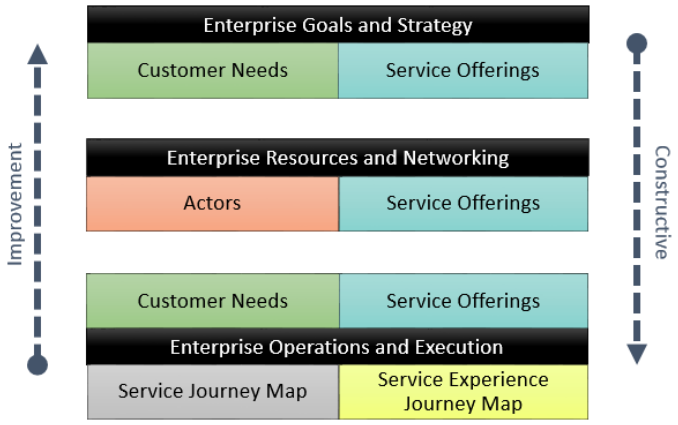

Fig. 10. ServiceML-enabled Enterprise Engineering

The primary objective is to optimize the functioning of the models by improving existing processes or creating new ones, if required, in order to support the innovative endeavour of the enterprise towards delivering more value to all stakeholders.

The ServiceML models are used as a foundation for further mappings and transformations towards service realisation with SoaML models, but also with CMMN and BPMN models, with a final realisation in a service oriented architecture. Based on Fig. 11, the NEFFICS platform provides different possibilities for the further support of service realisation. The models can be transformed to realisation models. The Business operations platform integrated with the NEFFICS platform provides support for BPMN based business process execution and also support for CMMN based case process management.

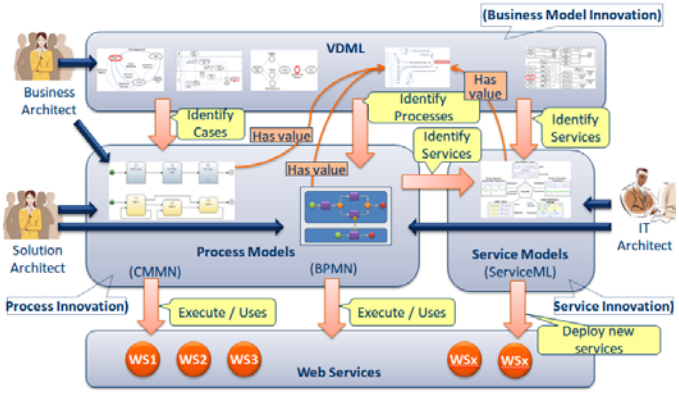

Fig. 11. ServiceML to SoaML and Service Realisation

Information models are supported by an Entity modeller with database persistence support, and user interface models are supported by an interactive designer with further realisations using XForms or by integration with external user interface tools. For the further realisation on cloud based platforms in particular, it is possible to make use of cloud modeling languages for configuration, orchestration and deployment, such as Tosca and CloudML.

VII. Contribution with Respect to Related Work

The Business model innovation approach with VDML is providing modeling support for the business model innovation cube [13, 20]. Compared to other visual approaches for business model innovation, like the Osterwalder Business model canvas [21], this provides an integrated model based representation for each of the building blocks well suited for further analysis and simulation.

The VDML language approach is based on a unification of concepts from a number of business modeling approaches like Value Networks [17], REA (Resource Event Agent) [22], e3Value [23] and the concept of role collaboration modeling also used in SoaML [9].

The contribution of VDML is that it brings together a number of foundational concepts from these approaches, and can serve as a starting point for further elaboration within each of these, as well as a bridge for further service model mapping and business model automation.

ServiceML is related to a multi view approach for service modeling like USDL[16], which focuses on different views with modules like legal, pricing, functional, technical, interaction and service level agreement aspects of services. ServiceML is, however, focusing more on the front end of service design where various informal service description approaches exist as it has emerged from the service design community [18, 19]. ServiceML also aims at taking advantage of existing metamodels and modeling language approaches like BMM [3] and Business SoaML [8], and has also created a metamodel for service journey models with inspiration from [18]. ServiceML bridges the realisation with SoaML, where further aspects of USDL and service realisation fit in.

The contribution of ServiceML is by serving as a bridge from early business model innovation and business model analysis to support the activities from service innovation through service design and service engineering.

VIII. CONCLUSION AND FUTURE WORK

This paper has presented a platform for Service Innovation and Service realisation with VDML and ServiceML. The platform is currently being tested and validated in different pilot case scenarios in the domains of services related to manufacturing (Vlastuin [24]) and health care as well as retail.

Future work is focusing on reflecting the experiences from the platform usage to potential updates to the current VDML standardisation proposal. The current modeling approach also provides a basis for further development of corresponding executable models and simulation models.

ACKNOWLEDGMENT

The research leading to these results has received funding from the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement n° 258076 NEFFICS and the Norwegian program CSI, Center for Service Innovation. The authors would like to thank members of the NEFFICS project and the CSI program for their contributions.

REFERENCES

[1] OMG, "Value Delivery Modeling Language (VDML), Revised submission for November 12, 2012," Object Management Group (OMG), OMG Document bmi/2012-11-06, November 2012, http://www.omg.org/cgi-bin/doc?bmi/12-11-06.pdf (Restricted to OMG members), also available at http://neffics.eu/

[2] A.J. Berre, H. de Man, Y. Lew, B.Elvesæter, B. M. Ursin-Holm, "Open Business Model, Process and Service Innovation with VDML and ServiceML," in M. Zelm, M.v.Sinderen, L. Ferraira Peres, G. Doumeingts (Eds), Enterprise Interoperability, Proceedings of the Workshops of the Fifth International IFIP Working Conference, IWEI 2013, Enschede, Wiley.

[3] OMG, "Business Motivation Model, Version 1.1," Object Management Group (OMG), Document formal/2010-05-01, May 2010.

[4] Cordys cloud based service platform,
Available at http://www.cordys.com/cloud_provisioning.

[5] Networked Enterprise transFormation and resource management in Future internet enabled Innovation CloudS (NEFFICS platform), http://neffics.eu/

[6] OMG, "Business Process Model and Notation (BPMN), Version 2.0," Object Management Group (OMG), Document format/2011-01-03, January 2011, http://www.omg.org/spec/BPMN/2.0/PDF.

[7] OMG, "Case Management Model and Notation (CMMN) Specification 1.0, Revised Submission," Object Management Group (OMG), OMG Document bmi/2012-11-05, November 2012, http://www.omg.org/cgi-bin/doc?bmi/12-11-05.pdf (Restricted to OMG members).

[8] T. Chang, A.-J. Berre, C. Carrez, and B. Elvesæter, "Business-SoaML: Service Identification and Specification from a Business Perspective," in Enterprise Interoperability V, Springer, 2012, pp. 379-389. http://dx.doi.org/10.1007/978-1-4471-2819-9_33.

[9] OMG, "Service oriented architecture Modeling Language (SoaML) Specification, Version 1.0.1," Object Management Group (OMG), Document formal/2012-05-10, May 2012.

[10] S. Clatworthy, "Innovations in service experiences: The AT-ONE method," in Proc. of the 6th International Conference on Design and Emotion, Hong Kong, 2008.

[11] Lankhorst, M., "Agile Service Development," Springer, 2012, ISBN 3642281877.

[12] A. Berre, "An Agile Model-based Framework for Service Innovation for the Future Internet," MDWE' 2012, Berlin, July 2012, in M. Grossniklaus, M. Wimmer (Eds): ICWE: 2012 Workshops, LNCS 7703, pp 1-4, Springer, 2012.

[13] P. Lindgren, R. Jørgensen, Y. Taran, and K. F. Saghaug, "Baseline for Networked Innovation Models, Version 1.0," NEFFICS (FP7-ICT-258076, Collaborative Project, 2010-2013), Deliverable D4.1, 11 May 2011, http://neffics.eu/

[14] P. Lindgren, Y. Taran, and R. Jørgensen, "Open Business, Networked Process, Service and Product Innovation Models, Version 1.0," NEFFICS (FP7-ICT-258076, Collaborative Project, 2010-2013), Deliverable D4.2, 20 September 2011, http://neffics.eu/

[15] C. Carrez, B. Elvesæter, and A.-J. Berre, "Unified and Extended Models for Networked Processes and Services (v2), Version 1.0," NEFFICS (FP7-ICT-258076, Collaborative Project, 2010-2013), Deliverable D5.6, 2012. http://neffics.eu/

[16] USDL, "Unified Service Description Language 3. 0 (USDL)," SAP AG, 2011,
http://internet-of-services.de/fileadmin/IOS/user_upload/pdf/USDL-3.0-M5-Archive.zip.

[17] V. Allee, "A Value Network Approach for Modeling and Measuring Intangibles," White paper, November 2002, http://www.vernaallee.com.

[18] B. D. Temkin, "Mapping The Customer Journey," Forrester, 5 February 2010,
http://quaero.csgi.com/writable/files/mapping_customer_journey.pdf.

[19] J. Schneider and M. Stickdorn, "This is service design thinking," Hoboken, New Jersey, John Wiley & Sons, Inc., 2011.A. Berre, H.d.Man, P. Lindgren, "Business Model Innovation with the NEFFICS platform and VDML," NGEBIS'13, Second International Workshop on New Generation Enterprise and Business Innovation, CAISE'2013, Valencia, June 2013.

[20] A. Berre, H.d.Man, P. Lindgren, "Business Model Innovation with the NEFFICS platform and VDML," NGEBIS'13, Second International Workshop on New Generation Enterprise and Business Innovation, CAISE'2013, Valencia, June 2013, CEUR-WS proceedings.

[21] A. Osterwalder, Y. Pigneur, "Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers," Wiley, 2012.

[22] Geerts, G. L. and W. E. McCarthy, "An Ontological Analysis of the Primitives of the Extended REA Enterprise Information Architecture," The International Journal of Accounting Information Systems, March 2002.

[23] Gordijn, J. and Akkermans, H., "Value based requirements engineering: Exploring innovative ecommerce ideas," In Requirements Engineering Journal, Vol. 8(2):114-134, 2003, http://e3value.few.vu.nl/docs/bibtex/pdf/Gordijn2003e3value.pdf.

[24] J. Lentjes, "Second Release of the Virtual Extended Factory," NEFFICS (FP7-ICT-258076, Collaborative Project, 2010-2013), Deliverable D1.4, 2012,http://neffics.eu/