Value-oriented Solution Development – uncovering the rationale behind organizational components

João Pombinho

Department of Information Systems and Computer Science, Instituto Superior Técnico
Technical University of Lisbon, Portugal

jpombinho@acm.org

Abstract. Formally rationalizing complex system design and construction decisions is non-trivial, particularly when involving different systems participating in a value chain. Most methods used up to now to manage complexity are not based on a transversal, coherent and concise conceptual model. Particularly, there is a lack of an integrated perspective that can be generally and recursively applicable to value chains, organizations and sub-organizations of several types and sizes. In order to incorporate value systematically and deliberately promote system/subsystem alignment, it is necessary to integrate teleologic and ontologic perspectives of the system at hand.

We devised a set of artefacts that support, by design, a recursive definition of the contribution of different system components. Combining DEMO and e3Value provides traceability and allows defining the rationale behind each organizational artefact. This allows formal specification of the rationale behind value network creation and system/subsystem bonding. We believe that modelling this rationale systematically will improve reactive and proactive change management through increased self-awareness, improved scenario specification, objective evaluation and well-grounded system development decisions.

Keywords: Value-oriented Solution Development Process, Enterprise Engineering, DEMO, Value Model, e3Value, Purpose, Contribution Perspective

1 Introduction

The complexity of businesses and the high change pace of their environments, coupled with increasing ICT support, turn the gap between strategy and its implementation into a major challenge. Over time, the inability to manage this complex set of factors will inevitably lead to failure. Studies indicate as much as 90 percent of organizations fail in applying their strategies [1]. Misalignments between the *business* and its support systems is frequently appointed as a reason of these failures [2, 3].

Formally rationalizing complex system design and construction decisions is non-trivial, particularly when involving different systems participating in a value chain. Most methods used up to now to manage complexity are not based on a transversal, coherent and concise conceptual model. Particularly, there is a lack of an integrated

perspective that can be generally and recursively applicable to value chains, organizations and sub-organizations of several types and sizes. The thesis aims to provide such a perspective and answer how to formally incorporate purpose into system development activities. In this paper, we review state of the art analysis and devise a set of concepts and a method to address the service development process, which includes harnessing the so-called subjective areas of organizational modelling.

In order to address these issues, this research proposes modelling enterprises as service systems along three perspectives, namely: construction, function and contribution. With our approach we are able to clearly specify, integrating the teleological/function (black-box) and ontological/construction (white-box) perspectives, how each component of an enterprise system provides value to other components of the ecosystem. Further, the concept of value is incorporated in the system development process in a traceable way as requirements make their way towards implementation. In turn, this allows running in the reverse direction in a rational manner during a reengineering process. Particularly, it also allows re-evaluating the constraints used in the process of the initial introduction of the component into the organization as rationale support for change scenarios and identifying innovative alternatives.

In this paper we present and discuss an innovative value-oriented approach to System Design and Engineering. It combines existing work in related domains, such as Enterprise Engineering [5], Enterprise Architecture [6], Service Science [7], Value Modeling [8] and ICT development. In the following subsections, related work will be briefly presented for discussion context.

2 The problem at hand: Where is the Value?

During the process of changing an enterprise one needs to constantly be aware of value conditions imposed by stakeholders in the past, present and future. At times such conditions will be obvious just by looking at an organization component and at times they will not. These value conditions show how each and every part of an organization contributes to the purpose of the organization itself and/or to the purposes of other elements in the organization's environment (market). While engineering any organization one should be aware of all the value threads that influence the existence of the components target of change.

Current research approaches do not model value in a formal way that is relatable to the composition of the value providing system. Behaviour specifications should be traceable upstream to specifications of user needs and system objectives and downstream to specifications of subsystems. Also, it is not clear how to structurally model system/subsystem relations. This makes it impossible to define the rationale of system intervention systematically for improving analysis and future change handling.

The following four problem areas where isolated:

- **Value Definition**. Value is, by nature, dependent on the stakeholder and, thus, subjective. The problems in adequately naming and scoping of a service, known in the Service Design community, are a symptomatic of lack of formal principles. What is the core transaction through which a system pro-

vides value? For instance, should a library core transaction be named "Loan book" or "Provide (limited-time) access to (reading) content"?

- **System (De)construction Support**. The construction of a system resulting from traditional development processes is a compiled structure that obscures the system/subsystem relations and their motivation. It is hard to separate a subsystem from its owner system, especially if it was modelled from a flat description of the operation of the organization, instead of a sequential bootstrap or an incremental design step.
- **System Intervention Rationale Modelling**. Forward traceability of a specification is the property that each part of the specification can be traced to specifications of lower-level components that implement it. It requires maintenance of links in the how-direction, which is down the aggregation hierarchy. Questions about the rationale behind past system interventions are commonly hard to answer. For instance: 1) When did the change take place? 2) What was its purpose? 3) What were the design principles, constructional principles, assumptions and constraints applied? Are they still valid?
- **Unidirectionality of the System Development Process**. Backward traceability of a specification is the property that for each part of the specification, it is specified why it is included. This refers to links in the why-direction, up the aggregation hierarchy. Backward traceability ensures that the impact of changing a component of a specification can be traced to changes in product behavior and in the service that the product delivers to its user [11]. The unidirectionality of the system development process limits the solution's value to the original functional request scope. Extra value that could be derived in bottom-up fashion, either at the original design time or in future interventions, is not addressed. This also implies the lack of capability to use existing solutions as structured and formal input to the development process.

From the problem areas, we have devised the following Research Question:

How to integrate the teleological and ontological perspectives of an enterprise model to support systematic value based system development decisions?

3 Approach and Methodology

The high-level approach was the following: formulate the problem, identify relevant state of the art in both practice and research communities, create a solution and apply it in practice, returning to the drawing board for refining/reformulating whenever necessary. To guide the aforementioned approach, Design Science Research was used. DSR aims at providing generic, innovative artefacts that solve important, relevant IS design problems. The cyclic nature of DSR adapts naturally to industry research setting: requirements from practical problems justify artefact design, which cannot be properly validated without returning to field testing. In this case, the candidate works at IT Demand Management (IT DM), a scenario described in [13]. The main functions are evaluating project proposals, devising solutions and respective business cases, evaluating Go/No Go and planning with project Delivery area.

We conceive solution development in a wide sense, not necessarily restricted to IT. Therefore, our research studies enterprises more generically as artificial [5, 15, 16], service systems [17] that are actors in value networks [8]. A distinction about the object of the design effort must be made right away, for clarification purposes: 1) the design of a particular to-be (sub)system and 2) the design of the system development process. In the context of our research, DSR applies to the second.

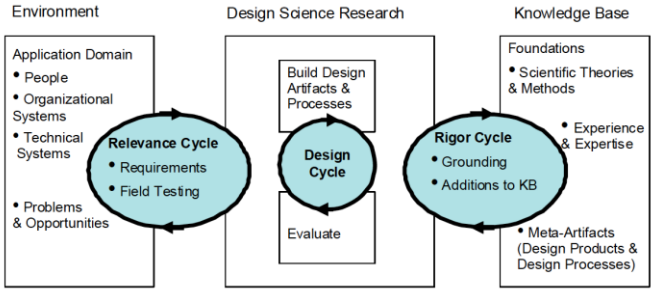

Figure 1. Design Science Research reference model [18]

The briefly described practical scenario plays the *Environment* role in a *Relevance Cycle*, presented in Figure 1. As for the *Knowledge Base*, the following main areas are included: General System Theory [16, 19]; Enterprise Engineering (DEMO – Design and Engineering Methodology for Organizations) [5]; Enterprise Architecture [20]; Service Modeling [21, 22]; Service Science (Service Science and SD-Logic) [7, 17]; Value Modelling (e3Value) [8]; Business Analysis (BABOK) [23]; Requirements Engineering (GORE) [24]; Value-based Software Engineering (VBSE) [25]; Benefits Management [26]; and ITIL Service Strategy [27]. A selection of these areas based on relevance to the solution will be presented in the next section.

In order to characterize our approach, we will define it along eight dimensions:

1. **Goal.** Solving an important, relevant IS design problem – see section Problem Statement. The target audience and beneficiaries of this thesis are the research community and enterprise engineering practitioners;
2. **Means.** To achieve the mentioned goal, a set of artefacts (see section on Contribution) was produced and adapted to operational processes;
3. **Genericity**.The artefacts were designed to solve a class of problems and not only a singular one. The combination of KB elements included in the design cycle applies to the scope of (artificial) service-based systems. The research began at the IT Management domain, and a general system development problem was formulated. Next, a generic solution was found and reapplied to IT by specialization and adaptation to operational processes;
4. **Solution requirements** were derived from the real-world scenario, generalized and its usefulness and validation validated in practice;
5. **Solution construction is grounded.** Soundness and consistency are rooted on DEMO's theory background and e3Value's formal ontology and procedural and evaluation rigor;
6. **Solution construction is transparent** – as it will be presented further, the ontological, white-box solution specification promotes its transparency;

7. **Solution process is iterative.** As mentioned on 2, the existence of a real-world scenario of artefact application induced an iterative process. Particularly, a value management maturity model was created [28];
8. **Solution utility is evaluated.** The validation techniques are argumentation, instantiation, case studies and results from a practical scenario of solution instantiation at a IT DM process. The nature of the research makes it difficult to evaluate, with challenges such as being an open process (so the effects may have many causes), having a partly subjective component, politics and the relatively long time necessary to see results. As the research approaches its final stages, the apparently positive results are expected to be confirmed by a formal evaluation, which include both quantitative (process metrics) [13] and qualitative analysis (ex: interviews with stakeholders).

4 State of the Art and Related Work

4.1 General System Theory and Service Science

The formal system definition we will use, from Enterprise Ontology [5], defines the following properties for a system: *composition* – a set of elements of some category; *environment* – a set of elements of the same category, disjoint from the composition; *production* – things produced by elements in the composition and delivered to the environment; and *structure* – a set of influence bonds among the elements in the composition, and between them and the elements in the environment.

The previous definition applies recursively to super and sub-systems with the following related activities: *system decomposition*, i.e., specifying a decomposition of a system at a certain aggregation level into subsystems. On the opposite direction, we have *system integration*, i.e., producing a system as an assembly of subsystems.

General systems theory provides a framework for understanding complex relations in configurations of operant and operand resources [19]. But service systems are not defined by the relations and interaction of resources alone. Some operant resource must act by providing the proposal, agreement, and determination of value-cocreation. In [17], the concept of *service system* is introduced, central to both service science (SS) and service-dominant (SD) logic.

4.2 DEMO and the GSDP

Unlike other approaches, DEMO makes a very strict distinction between teleology, concerning system function and behaviour – the black-box perspective – and ontology, about its construction and operation – the white-box perspective [14]. These perspectives are embodied in the Generic System Development Process (GSDP). It begins with the need by a system, the Using System (US), of a supporting system, called the Object System (OS). From the white-box model of the US, one determines the functional requirements for the OS (function design), formulated in terms of the construction and operation of the US. Next, specifications for the construction and operation of the OS are devised, in terms of a white-box model (construction design). Choices

are then made with each transition from the top-level white-box model towards the implementation model. However, nothing is prescribed about the rationale behind these choices. System design decisions, either implicit or explicit, remain solely in the minds of the participants in the process. The complexity can quickly cross the limits of unsupported human handling. It may then become short of impossible to know the rationale of past decisions, its impacts and dependencies in designing the to-be.

Aveiro took a step towards instantiating the GSDP by applying DEMO to specify the models of the sub-organizations responsible for handling change caused by exceptions. In the control sub-organization [29], the viability of a system is specified by a set of measures and respective viability norms that can be periodically checked against the operational status. If such norms are violated, a dysfunction handling mechanism is triggered. If the exception that causes the dysfunction to the norm is expected, solutions that have previously been identified in anticipation are applied and evaluated for solving the problem. If the cause is unexpected, an organizational engineering process (OEP) must be started, that occurs in the scope of another sub-organization, the G.O.D. organization [30], responsible for specifying and implementing change that will solve or circumvent the unexpected exception causing a dysfunction. The solution may be new organizational components (e.g., new norms, new actors, processes and rules, etc.) or just (re-)allocation of human or IT resources.

4.3 Value Modeling

A formal business model [31] is necessary to create a founded integration with constructional models. Value Modelling is increasingly recognized that the concept of value assists in improving stakeholder communication, particularly Business and IT [32]. All organizations have in common bringing about *value* to their *environment*, either directly or indirectly, so *value* is an unifying concept to consider in business modeling. The high relevance of the value perspective for software engineering has been shown in the Value-Based Software Engineering (VBSE) research discipline [25]. As our intended application scope is general artificial system development, we will begin by grounding on a more general approach. For our case, we elected e3Value [8] because of its formal ontology, practical application, coverage financial evaluation coverage and tool support.

e3Value is part of e3family, a set of ontological approaches for modelling networked value constellations. It is directed towards e-commerce and analyses the creation, exchange and consumption of economically valuable objects in a multi-actor network [31]. In e3Value, an Actor is perceived by his or her environment as an economically independent entity, exchanging Value Objects.

As we will see, our proposal is to apply e3Value in a way that improves system and subsystem value modelling: inside the boundaries of organizations, instead of e-commerce relations between formal organizations. e3Value models are directly created and do not directly address problem statements or requirements engineering. Therefore, a previous modelling step must be introduced – in this case, modelling the problem that a value chain is set up to solve for its end-customer.

4.4 Enterprise Architecture

Archimate is an Enterprise Architecture modelling language with broad practical application. In the corresponding architecture framework, three enterprise layers are distinguished: business, application and technology. In comparison with DEMO, the business layer of Archimate and the ontological layer of DEMO are considered. The contents of Archimate's application and technology layers are regarded as implementation and, thus, not directly modelled in DEMO. The business layer of Archimate relates to all three B-I-D layers of DEMO, without clear distinction [33]. These undesirable large degrees of modelling freedom result in incompleteness and incoherencies. DEMO, for instance, would provide both completeness by matching with the Transactional Pattern and isolating essential (ontological, i.e. B-) transactions. In turn, the elements of a DEMO model can be matched to an e3Value model [12, 34].

Representing the motivation has value in itself but, in order to reap the full benefits of addressing the motivation layer, it must go further into *engineering* the business itself. Regarding the connection with business modelling, other approaches combine Archimate and Business Modelling Canvas, a simplified version of the Business Model Ontology [9]. However, none of this approaches uses the concept of *intention*, a characteristic of the so called third wave of approaches, like DEMO [33].

4.5 Business Modeling

In a system's lifecycle, the concept of Value Proposal [9] is realized during operation, when it is more tangible. However, the way it is achieved and supported must be addressed at the scope of system development. The Business Model Canvas (BMC) [35] offers a systematic approach to developing business models. While widespread success as an innovative approach to business model generation is recognized, the fact that it is a top-down and relatively informal approach, lacking constructional depth even from a business perspective has been noted [36]. Accordingly, the potential benefits of combining with a more formal approach, specifically e3Value, have been recognized by Osterwalder and a comparison is presented in [37].

5 Solution - Value-Oriented Solution Development Process

5.1 The System Development Continuum

In order to create a referential where to position the studied approaches we devised the System Development Continuum (SDC). The SDC, presented in Figure 2, comprises teleology and ontology areas, with the relative positioning representing conceptual distance to either from the approach at hand. Objectivity plays a central role as it can help characterize the scale: the teleology is concerned with what particular system users (subjects) do with it, whereas ontology addresses what it is and can be agreed upon by the maximum number of independent observers.

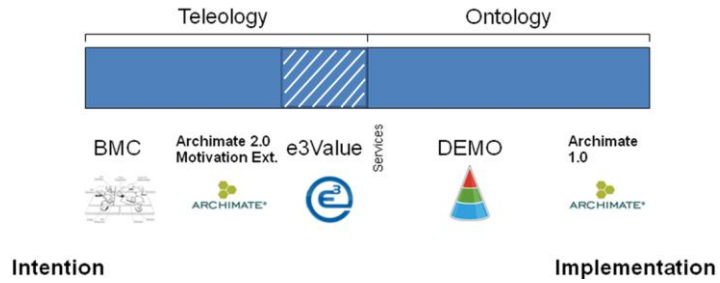

Figure 2. Positioning of reference approaches along the SDC

The objective/subjective relation is not one-to-one with ontology/teleology [38]. Indeed, the pursuit of objectiveness can be extended from the ontology through teleology areas (extension represented as striped area in Figure 2). Further, the formulation of a transaction result always has some degree of subjectiveness to it.

Our approach to the challenges mentioned fundamentally differs from the state of the art by following three main lines:

1. Distinguishing three perspectives of a system: construction (white-box), function (black-box) and *contribution* (environment relations);
2. Formally defining the *value* of a system as a relation with other system;
3. Defining purpose by decomposing a system into its components and recursively creating value models that represent decomposition rationale.

The approach followed in the thesis involves 1) distinguishing the three mentioned perspectives and 2) articulating the concepts of each perspective so that an end-to-end, integrated, model is provided. To this end, we devised a conceptual framework (cf. Figure 3) that supports recursive contribution definition by design. We also provide a method to perform value-based alignment between the three named perspectives of a system. Specifying the value in a contribution perspective allows improved specification of the rationale behind value network establishment and system/subsystem bonding. We can now specify how each component of a system S contributes (provides value) not only to the purpose of S but to other purposes present in the value chains S participates in. The most distinguishing feature is using the concept of value proposal of a system to express the motivation that drives its development and integration within existing systems. We propose to explicitly build and manage the value system behind a given change rationale/business case. This enables transporting to operation environment instead of being a solely a design-time artefact.

5.2 The Framework

Systems provide services that are valued by users as solutions to their problems. They form networks of actors, each responsible for certain solution components. We have created a four layer framework represented in Figure 3 and briefly described next.

The Purpose layer, i.e., Problem/Solution is specified by modelling means-ends relations as described in [25]. The essentially serve to establish causal bonding between consecutive nodes in a solution chain. In the Value layer, the value generation mecha-

nisms are specified – it is essential to identify the variables influencing and their relation, i.e., the structure of the value system. The value system comprises a set of system construction viability equations. Viability conditions [39] are then established over specific scenarios of the value equation. Next, services complying with the value equations are procured in the market and/or developed internally, in this case restarting the problem/solution cycle at level N+1. As solutions are trivially found in the market or trivially developed internally, system (actor role) specification is closed bottom-up, thereby completing actionable solutions (by instantiating actor roles with concrete subjects). Next we will detail the main aspects of this approach.

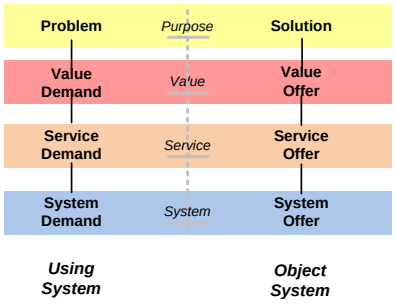

Figure 3. Value-oriented System Development Framework

5.3 Defining Value

We consider the combining the essential modeling of the business organization, in DEMO, with its value model, in e3Value, as a crucial step towards formal value definition. Both approaches were compared and related in [40] but no detailed ontological matching was provided nor apparently intended. Their alignment is critical to the *contribution perspective*, as defined in [38], since it relates a production fact and the value object it pertains to.

The service value concept is based on e3Value by relating the buyer/seller dual-party semantics to DEMO's Using System and Object System. Both DEMO and e3Value are centred in *transactions* between actors. *Actors* are the active elements of both social systems and value networks. In DEMO, an actor is a *subject* fulfilling an *actor role* in a *transaction type*. The *initiator* and *executor* actor roles are bound by their common interest in bringing about a *production result*. In e3Value, both actors (*provider* and *requester*) are bound by the willingness to share value objects with the concept of reciprocity. The *transaction* concept represents the relationship between actors by associating *value ports* of different directions. A unitary DEMO transaction relates to a *value exchange* in e3Value. A *value transaction* involves at least two, according to the principle of *economic reciprocity* - the supplier is only willing to exchange objects via all ports of its value interface, or none at all.

As said, the critical dimension to the alignment is between *production fact* (DEMO's Transaction Result) and *value object's* exchange, as the production of each service transaction determines its effective contribution to the value chains it

participates in. The principle of *economic reciprocity* makes it necessary for DEMO's transactions to have a *counterpart*. A *value object* is specifiable as the combination of an access right to some resource and a transformation enabled by this resource [41]. In the classical DEMO Library example, the requested production facts are:

- *R01 - book loan (**possession of a physical book copy**) has started*
- *R02 – book loan was paid (**money**)*

In this simple case, it is trivial to relate the production fact and value object. However, it can be hard in the cases where the value-orientation is not present in the transaction definition and it is necessary to make value explicit. For instance, the implicit value objects of R01 and R02 are now explicit, in bold, inside brackets.

Decomposing value exchanges down to these primitives explicitly connects the value model of the system with its construction. In addition to economic reciprocity and full transactional support, the alignment with a constructional model allows checking for VO completeness. VOs are either brought from outside the system or must be derivable from existing VOs through a certain value activity. Also, coherency and alignment can be enforced, e.g., that are no isolated actors and no isolated value transactions (every actor role has at least a value transaction and *vice versa*).

5.4 Defining Contribution: a matter of perspective

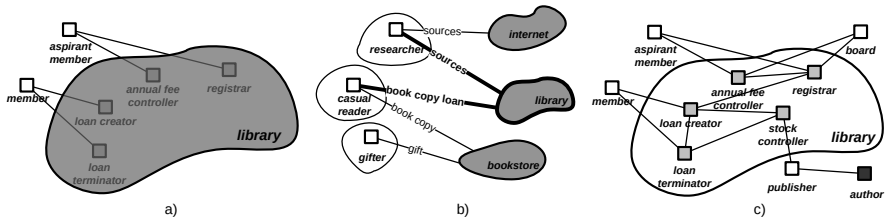

Figure 4. Construction, Function and Contribution perspectives

We defined an additional perspective [38] that allows distinguish the strict notion of function from contribution and modelling the later with explicit links to the respective stakeholders, i.e., systems in the environment. In Figure 4 a), we can see the partial functional perspective of the member of a Library (including aspirant member, which is an earlier mandatory state). Figure 4 b) shows the contribution perspective, which comprises the systems boundary and the relations with its using systems. The construction perspective of the Library system is shown in Figure 4 c). The grey nodes are part of the composition, the white nodes are the Library system's environment and the black node (author) is not part of the system.

5.5 Relativity & Recursivity

Porter's Value Chain concept differentiates core and support business processes [42]. It must, however, be noted that these are relative concepts. Moreover, they are applicable to a constellation of systems (organizations) interoperating together.

The Liquid Enterprise is one of nine Qualities of Being defined as strategic for enterprises of the future in the Future Internet Enterprise Systems (FInES) Research Roadmap 2025 [43] by the European Commission. The concept of Liquid Enterprise refers to the blurring of the enterprise boundaries, making it hard to distinguish the 'inside' and the 'outside', the employees and the partners, the competitors and the collaborators and new forms of labour and collaboration (e.g., the 'workeprenuer') flourish. Every component of a business can potentially be a business by itself. One of our most distinctive contributions is freeing the concept of service system from aligning with the boundaries of a firm as there is no real reason to have that limitation during design steps. To support this, it is necessary to have the theory to decompose a system and chain the elements together with explicit value-oriented rationale.

5.6 Value-oriented System Development Process

We define a solution to a problem as the production of a determined result, which generally involves investment of resources (time, money, effort, etc.) by the Object System (OS) and generates value for some stakeholder, the Using System (US). By asking the solution requester to define the construction of the US and its value model, additional insight can be derived from its specification – changing the problem or dissolve it altogether. However, the entry point of the GSDP, i.e., the origin of the system development request, is not sufficiently clear in the original model.

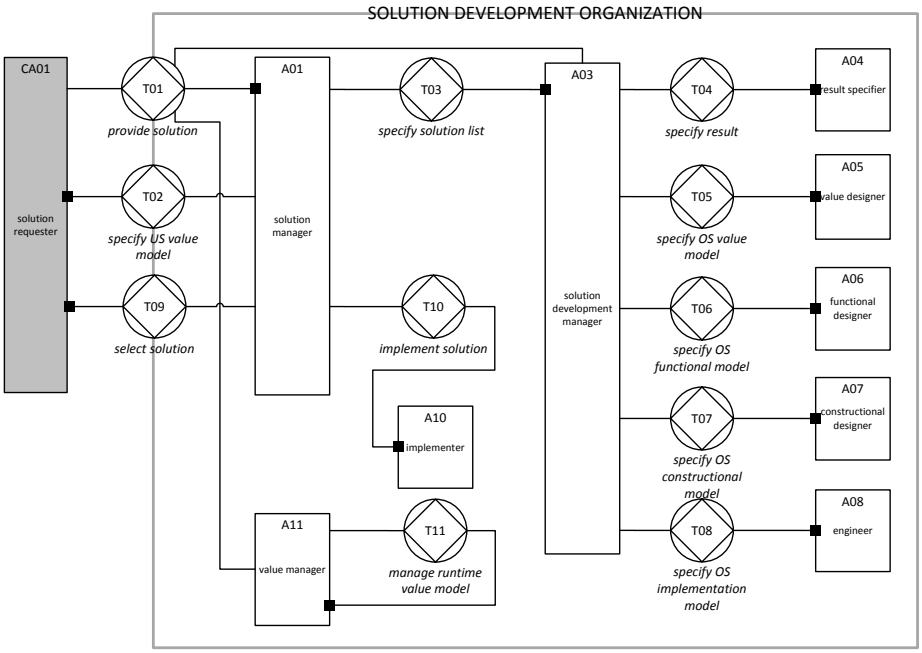

Figure 5. Solution Development Organization – ATD

To overcome this issue, we defined the Solution Development Organization (SDO), presented in Figure 5. In our view, the description originally provided for the GSDP was not ontologically complete and some adjustments were in order to obtain a coherent model of the SDO. Particularly, we defined a recurrent provide solution transaction (N+1) as a new solution development cycle where the current OS assumes the role of US and a new OS is being developed so that its function serves the construction of the US. This transaction is represented by the link between A03 and T01 and is crucial for explicit multi-cycle solution development, i.e., function/construction alternation. Conceptually, the design cycle is closed by the profitability sheet versus viability equation. This cycle always involves iterating through Problem/Solution Model, Value Model, Function Model and Construction Model.

The process begins with an external request to provide a solution. The solution manager asks the solution requester to specify the Using System value model, which is critical to identify rational solutions. The solution manager then requests that the solution development manager specifies a solution list to produce the result requested, in the context of the US value model. The *specify results* transaction is the creative step of the process, where different ways of producing the required result (solution) are identified. For each result, value and functional models are specified in sequence.

Next, the constructional model is built, where transactions and actors are specified. If there is a dependency in producing the result, then another solution development process is triggered, with the solution development manager requesting a solution for that problem. The current OS is repositioned, assuming the role of US in the new development cycle. Such a request would be made by the level 1 solution provider to level 2 providers. For each crossing of these levels, a new GSDP iteration takes place. Along each single thread of a solution chain, the alternation between each pair of levels is described by Dietz and Hoogervorst as function/construction alternation [5]. A set of such iterations is commonly performed implicitly inside a single GSDP, and thereby kept from being adequately modelled by the explicit application of functional, constructional and architectural principles.

When the set of known solutions is considered satisfying by the solution manager, it requires that the solution requester elects a solution from the presented alternatives. The elected solution is implemented and its value proposal is periodically monitored by the value manager. If an inconsistency is found, the *provide solution* transaction is invoked to address the gap, presented as an economic viability problem.

The SDO supports reconstruction of the development process rationale by recording the alternatives considered and their simulations and evaluations. It is, thus, an crucial instrument for backward traceability of specifications to the reasons why they were chosen over other specifications [11].

6 Contributions and next steps

Following Design Science Research methodology, the research goal is to create a set of innovative artefacts capable of describing the purpose of a system through value specification and use it to address change in Enterprise Engineering.

A 4 layer framework (System, Service, Value and Purpose) and an ontology that builds on DEMO, for the System Layer, and e3Value, for the Value Layer, was created. Additionally, there is also an implied method resulting from the combination of the integration ontology and DEMO's GSDP which was made explicit by creating an ontological model of the Solution Development Organization. Finally, a Protégé-based prototype was used for supporting a preliminary instantiation (between two systems, not a full network) for validating the ontology matching. In summary:

- **Constructs**
  - By studying, analysing and integrating theories and concepts from different areas, such as systemics, economics, enterprise engineering and enterprise architecture, a set of constructs was created, formed by inherited constructs from the knowledge base and new concepts that were necessary to perform the integration;
- **Model**
  - **Value-oriented Solution Development Framework** – Improve problem definition and elicitation by using the concepts of value and system positioning in a value-providing chain, subsystem value generation and the dependencies that form in the value chain;
  - **Integration of DEMO and e3Value** – Provide an implementation of an integration ontology of e3Value and DEMO ontologies to systematically specify the relation of two systems from the contribution and construction perspectives and their alignment;
  - **System Development Organization** – Ontological specification;
- **Method**
  - **Value-oriented Solution Development Process** - Improve system design and engineering change evaluation and decision rationale in terms of the application of design principles, constructional principles, assumptions and constraints in a structured way that is relevant and explicitly included in the resulting model;
- **Instantiation**
  - Demonstrate the usefulness of the approach by using it in modelling and evaluating the value context of a system in a case study.

Until this moment, the Bibliographic Research, Problem Definition and Research phases of the Work Plan were concluded. Obtaining Results and Dissertation Writing phases are currently ongoing.

References

1. Kaplan, R.S. and D.P. Norton, *Strategy Maps : Converting Intangible Assets Into Tangible Outcomes*. 2004, Boston, Mass.: Harvard Business School Press.
2. Henderson, J.C. and N. Venkatraman, *Strategic alignment: leveraging information technology for transforming organizations.* 1993. **32**(1): p. 4-16.
3. Laudon, K.C. and J.P. Laudon, *Management Information Systems: Managing the Digital Firm*. 2011: Prentice Hall.
4. Boehm, B. and A. Jain, *An Initial Theory of Value-Based Software Engineering*. 2005.

5. Dietz, J.L.G., *Enterprise Ontology: Theory and Methodology*. 2006: Springer.
6. Lankhorst, M., *ArchiMate Language Primer*. 2004.
7. Vargo, S.L., R.F. Lusch, and M.A. Akaka, *Advancing Service Science with Service-Dominant Logic: Clarifications and Conceptual Development*, in *Handbook of Service Science*. 2010, Springer.
8. Gordijn, J., *Value-based requirements Engineering: Exploring innovatie e-commerce ideas*. 2002, Vrije Universiteit Amsterdam: Amsterdam.
9. Osterwalder, A., *The Business Model Ontology - a proposition in a design science approach*. 2004, Universite de Lausanne.
10. Oliver, R.W., *Real-Time Strategy: What Is Strategy, Anyway?* Journal of Business Strategy, 2001. **22**(6): p. 7-10.
11. Wieringa, R.J., *Requirements Engineering: Frameworks for Understanding*. 2006, Amsterdam: Wiley.
12. Pombinho, J., D. Aveiro, and J. Tribolet. *Towards Objective Business Modeling in Enterprise Engineering - Defining Function, Value and Purpose*. in *2nd Enterprise Engineering Working Conference*. 2012. Delft, The Netherlands: Springer.
13. Pombinho, J., D. Aveiro, and J. Tribolet, *The role of IT Demand Management on Business IT Alignment - the case of ZON Multimedia*, in *PRET 2013*, F. Harmsen and E. Proper, Editors. 2013, Springer: Luxembourg.
14. Dietz, J.L.G., *Architecture - Building strategy into design*. 2008, The Hague, The Netherlands: Netherlands Architecture Forum, Academic Service - SDU.
15. Simon, H., *The Sciences of the Artificial*. 3 ed. 1996, Cambridge, MA: MIT Press.
16. Skyttner, L., *General Systems Theory: Problems, Perspectives, Practice*. 2nd ed. 2005, Singapore: World Scientific Publishing Co. Pte. Ltd.
17. Maglio, P.P., S.L. Vargo, N. Caswell, and J. Spohrer, *The service system is the basic abstraction of the service science.* Information Systems and eBusiness Management, 2009.
18. Hevner, A.R., *A Three Cycle View of Design Science Research.* Scandinavian Journal of Information Systems, 2007. **19**(2): p. 87-92.
19. von Bertalanffy, L., *General system theory: foundations, development, applications*. 1973: G. Braziller.
20. The Open Group, *Archimate 2.0 Specification*. 2012: Van Haren Publishing.
21. Bell, M., *Service-Oriented Modeling: Service Analysis, Design and Architecture*. 2008, New Jersey: John Wiley & Sons.
22. Terlouw, L., *Modularization and Specification of Service-Oriented Systems*. 2011, Delft Technical University: Delft, The Netherlands.
23. International Institute of Business Analysis, *The Guide to the Business Analysis Body of Knowledge Version 2.0*. 2009.
24. Lamsweerde, A.V. *Goal-Oriented Requirements Engineering: A Guided Tour*. in *International Symposium on Requirements Engineering*. 2001. Toronto.
25. Boehm, B., *Value-Based Software Engineering.* SIGSOFT Software Engineering Notes, 2003. **28**(2): p. 4.
26. Ward, J. and E. Daniel, *Benefits Management: How to Increase the Business Value of Your IT Projects*. 2012: Wiley.
27. Great Britain Office of Government Commerce, *Service Strategy: ITIL*. 2 ed. 2007: Stationery Office.

28. Pombinho, J., D. Aveiro, and J. Tribolet. *Value-oriented Solution Development Process - uncovering the rationale behind organization components*. in *3rd Enterprise Engineering Working Conference*. 2013. Luxembourg, Luxembourg: Springer.
29. Aveiro, D., A.R. Silva, and J. Tribolet. *Control Organization: A DEMO Based Specification and Extension*. in *First Enterprise Engineering Working Conference, EEWC 2011*. 2011. Antwerp, Belgium: Springer-Verlag Berlin Heidelberg.
30. Aveiro, D., A.R. Silva, and J. Tribolet. *Extending the Design and Engineering Methodology for Organizations with the Generation Operationalization and Discontinuation Organization*. in *5th International Conference on Design Science Research in Information Systems and Technology*. 2010. St. Gallen, Switzerland.
31. Kundisch, D., T. John, J.r. Honnacker, and C. Meier. *Approaches for Business Model Representation: An Overview*. in *Multikonferenz Wirtschaftsinformatik 2012*. 2012.
32. Cameron, B., S. Leaver, and B. Worthington, *Value-Based Communication Boosts Business' Perception Of IT*. 2009, Forrester Research.
33. Ettema, R. and J.L.G. Dietz, *ArchiMate and DEMO – Mates to Date?*, in *Advances in Enterprise Engineering III*, A. Albani, J. Barjis, and J.L.G. Dietz, Editors. 2009, Springer Berlin Heidelberg. p. 172-186.
34. Pombinho, J. and J. Tribolet, *Modeling the Value of a System's Production – Matching DEMO and e3Value*, in *6th International Workshop on Value Modeling and Business Ontology*. 2012: Vienna, Austria.
35. Osterwalder, A. and Y. Pigneur, *Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers*. 2009: Self-Published.
36. Rosenberg, A., M.v. Rosing, G. Chase, R. Omar, and J. Taylor, *Applying Real-World BPM in an SAP Environment*. 2011: SAP Press.
37. Gordijn, J., A. Osterwalder, and Y. Pigneur, *Comparing two Business Model Ontologies for Designing eBusiness Models and Value Constellations*, in *18th Bled eConference - eIntegration in Action*. 2005: Bled, Slovenia.
38. Op 't Land, M. and J. Pombinho. *Strengthening the Foundations Underlying the Enterprise Engineering Manifesto*. in *2nd Enterprise Engineering Working Conference*. 2012. Delft, The Netherlands: Springer.
39. Aveiro, D., *G.O.D. (Generation, Operationalization & Discontinuation) and Control (sub)organizations: a DEMO-based approach for continuous real-time management of organizational change caused by exceptions*. 2010, UTL: Lisboa.
40. Weigand, H. and W.-J.V.D. Heuvel. *A Conceptual Architecture for Pragmatic Web Services*. in *First International Conference on the Pragmatic Web*. 2006. Stuttgart, Germany.
41. Weigand, H., P. Johannesson, B. Andersson, M. Bergholtz, A. Edirisuriya, T. Ilayperuma, E. Dubois, and K. Pohl. *On the Notion of Value Object*. in *CAISE 2006*. 2006. Luxembourg: Springer Berlin / Heidelberg.
42. Porter, M.E., *Competitive Advantage: Creating and Sustaining Superior Performance*. 1998: Free Press.
43. Missikoff, M., Y. Charabilidis, R. Goncalves, and K. Popplewell, *Future Internet Enterprise Systems - Research Roadmap 2025 - version 2.0*. 2012, European Comission.