Antonia Albani

Theory-Driven Reverse Engineering of Organisations

Driven by the strong demand for reusable yet situational business solutions on the one side and the necessity to provide a stable, reliant foundation that allows to adapt the supporting information systems in a systematic way on the other side, the need for a closer link between relevant theories and successful practices for the design of enterprise information systems becomes evident. The aim of the reported research is to provide such a link by means of prescriptive guidelines for the class of problems concerning the reverse engineering of organisations. The reverse engineering of organisations aims at deriving at the ontological models of organisations, which build the basis for the design and engineering of information systems supporting the business needs. An ontological model as used in the presented research is defined as the highest-level constructional model of an organisation, which is fully independent of its implementation. The prescriptive guidelines for reverse engineering presented in this paper are derived from the Ψ-theory (the Greek letter Ψ is pronounced PSI, which stands for Performance in Social Interaction), the theory that underlies the notion of Enterprise Ontology. This theory regards organisations as social systems and sees IT systems as support for social actors in performing coordination-related activities and production-related activities. In this paper we focus specifically on recommendations based on the Ψ-theory concerning the coupling of two types of enterprise models in order to derive at ontological models of organisations. The first type of models are derived by applying the Design and Engineering Methodology for Organisations (DEMO) and the second type of models are derived by applying the Architecture of Integrated Information Systems (ARIS).

1 Introduction

In the field of enterprise engineering, i.e., the discipline of systematically developing solutions to organisational problems, including the development of enterprise information systems, several methodologies exist and are widely applied in practice to aid the enterprise engineer. However, most of them lack a *theoretical foundation*. In a sense this may not be a major disadvantage. If the appropriate methods and procedures are applied appropriately the results may be of great relevance for practitioners in order to better understand the construction of the enterprise. However, when such models build the basis for the development of enterprise information systems supporting the business needs, major problems may arise regarding the necessity to provide a stable and reliant foundation that allows adapting the supporting information systems in a systematic way. Additionally, it is often the case in enterprises that different types of models exist, which have been derived at by applying different methodologies. Based on the discrepancies of the available models, no stable basis is provided for the development of the supporting enterprise information systems.

Following this argumentation the need for a closer link between relevant theories and appropriate methodologies for the design and engineering of enterprises and their supporting

information systems becomes evident. The aim of our research is therefore to provide such a link by means of *recommendations based on theory guiding the reverse engineering process*. The theory we use in our research is the Ψ-theory (the Greek letter Ψ is pronounced PSI, and stands for Performance in Social Interaction) (Dietz 2006a), which is the starting point for profoundly understanding the organisation of an enterprise. The resulting essential enterprise models are called Enterprise Ontology (Dietz 2006a). The Enterprise Ontology provides the highest-level constructional models of an organisation – also called ontological models – which are completely independent of their implementations. The ontological models of an organisation build the basis for the design and engineering process of the supporting information systems. With the application of the guidelines, which we derive from the Ψ-theory, we show how far mainly practically proven approaches can be theoretically grounded aiming at providing a coherent, consistent, and comprehensible foundation for actual solution engineering in practice. Our research goes beyond existing work, where either too general or too specific recommendations are given – if prescriptive recommendations are provided at all. Existing work is mainly focusing on the definition of notations for enterprise modelling or on the development of artefacts (e.g., enterprise models), and not on the definition of guidelines.

In this paper we present results on how models, which are derived at by applying the Architecture of Integrated Information Systems (ARIS) (Scheer 2000), can be theoretically founded allowing for the coupling of models, which are derived at by applying the Design and Engineering Methodology for Organisations (DEMO) (Dietz 2006a). Since the DEMO methodology implements the Ψ-theory by means of constructs, models and methods, instantiating the design rules provided by the Ψ-theory, the resulting models are already theoretically sound. There is no need to further improve those models.

The necessity of coupling ARIS and DEMO models as presented in this paper arose from the need of two national service centres at the ministry of the national government in the Netherlands. Both centres have their own models for describing their part of the organisation, but since both types of models – ARIS as well as DEMO models – deal with content of the same organisation, it was expected that connecting them could be very beneficial for having one sound ontological model of the organisation part. The investigation for the coupling of both types of models in a consistent way was therefore aimed at.

The paper is structured as follows. In Sect. 2 we discuss the different types of available theories and illustrate why we have chosen the Ψ-theory as a theoretical base to derive at the presented guidelines. In Sect. 3 our notion of reverse engineering, design and engineering is given and the research question is formulated based on it. In Sect. 4 the Ψ-theory is introduced. The prescriptive guidelines are presented in Sect. 5. Those guidelines are based on the Ψ-theory and allow for the coupling of DEMO and ARIS models. Since the presented research was conducted in cooperation with a governmental agency in the Netherlands, an overview of the case study and the evaluation of the results are presented in Sect. 6 and information about the applicability of the approach is given in Sect. 7. The summary and an outlook on future research can be found in Sect. 8.

2 Related Work

Theories are discussed, developed and applied in many disciplines. In general, five relevant types of theories can be distinguished according to Gregor (2002): *Theory for Analysing and Describing, Theory for Understanding, Theory for Predicting, Theory for Explaining and Predicting, and the Theory for*

Design and Action. While *theories for analysing and describing* focus on the 'what is' in order to describe or classify specific characteristics of individuals, groups, situations, or events, *theories for understanding* explain 'how' and 'why' something occurred. *Theories for predicting* say 'what will be' aiming at predict outcomes from a set of explanatory factors, without necessarily understanding or explaining the causal connections between the dependent and independent variables (Gregor 2002). To many people, *theories for explaining and predicting* are the 'real' theories, since they address 'what is', 'how', 'why' and 'what will be'. This type of theory implies both prediction and understanding of the underlying causes as well as a good description of the theoretical construct (Gregor 2002). Authorities, such as Dubin (1978) advocate this type of theory. The last type of theory mentioned is *theory for design and action*. This is a normative or prescriptive type of theory. It gives guidelines and/or principles that can be followed in practice.

Since the information systems discipline is concerned with the effective design, delivery, use and impact of information technology in organisations and society (Avison and Fitzgerald 2002) and, as discussed in the introduction, guidelines are often missing, we focus on this type of theory for our research. The value of an information system design theory is to reduce developers' uncertainty by restricting the range of allowable system features and development activities to a more manageable set, thereby increasing the reliability of the development process and the likelihood of success (Walls et al. 1992). Design theory concerns both, how to undertake the building of an artefact (development process), and what the artefact should look like when built (design principles) (Gregor 2002). According to Gregor (2002) 'it must be able to articulate the principles instantiated in the method, tool, process, or design. It is the articulation, whether in natural language, diagrams or similar, that constitutes the design theory'. Walls et al. (1992) state that an information systems design theory has two distinctive characteristics: a theoretical base and an explicit guidance for practitioners. Gregor goes even a step further and says that an information system design theory should have well-defined constructs, definitions and propositions that both explain and predict phenomena (Gregor 2002).

Known examples of design theories for information systems are introduced e.g., by Walls et al. defining an information system design theory for Vigilant Executive Information Systems (Walls et al. 1992), and by Markus et al. (2002) defining a design theory for Systems that Support Emergent Knowledge Processes. The guidelines provided in information systems design theories known to us are mostly domain specific. We aim instead at providing basic guidelines for modelling an organisation and therefore facilitate the development of its supporting enterprise information systems.

We follow the recommendations of Walls et al. and build our guidelines on a theoretical base, which is provided by means of the Ψ-theory (Dietz 2006a). The Ψ-theory is a theory for understanding organisations, defining human communication as being the root of information and social action. The Ψ-theory defines elementary concepts and interdependencies for actions in organisations, focusing specifically on the coordination and production acts between actor roles. The emphasis is on essential acts, which are implementation independent acts. This emphasis helps in understanding the fundamental commitments subjects enter into or comply with regarding the products/services they bring about in cooperation.

The Ψ-theory focuses on the use of language to achieve agreement and mutual understanding. As defined by Dietz (2010) the Ψ-theory

bridges the gap between information systems engineering and the organisational sciences in providing answers to questions like 'how can the immense complexity of organisations be made intellectually manageable' or 'what is the common core of information, action and organisation?' The Ψ-theory contributes specifically to the explanation of how and why people cooperate, namely by entering into and complying with commitments. By applying the Ψ-theory one can disentangle the essential knowledge of the construction and the operation of the organisation of an enterprise and make communication in organisation becoming transparent.

Amongst others, the Ψ-theory (Dietz 2006a) finds its roots in Language Philosophy, in particular the Language Action Perspective (LAP) (Crawford 2006; Flores and Ludlow 1980; Goldkuhl and Lyytinen 1982) and in Systemic Ontology (Bunge 1979). A complete overview of the theory is available in the book (Dietz 2006a) and the papers (Dietz 2006b; Dietz and Albani 2005; Dietz and Hoogervorst 2007, 2008).

To our knowledge, one of the few enterprise engineering approaches, which are explicitly based on theory is the DEMO methodology (Dietz 2006a). DEMO is an approach that focuses on modelling the essential features of an organisation and is based on the Ψ-theory. Many other enterprise engineering approaches exist but most of them completely lack a theoretical foundation. Worth mentioning are e.g., the *ARIS Platform for Business Process Management* (Scheer 2000) and the *Business Process Re(Engineering)* approach (e.g., cf. Davenport 1993; Davenport and Short 1990; Hammer 1990; Hammer and Champy 1993), which has been established in the early 1990ies. Some approaches, such as the *Business Engineering* approach (Österle and Winter 2003) or the *Work System* approach of Alter (2006, 2009) claim to be founded in theory, namely the theory of hierarchical systems (e.g., cf. Mesarovic et al. 1970) and the systems theory (Checkland 1999) respectively, but these theoretical foundations seem not to be appropriate and/or sound. For an overview and comparison of the different enterprise engineering methodologies we refer e.g., to Ilnseher (2007).

As e.g., mentioned in Albani and Dietz (2011), a missing theoretical foundation may result in incoherent (i.e., the parts do not constitute an integral whole), inconsistent (i.e., contradictions and irregularities may exist), incomprehensive (i.e., not all relevant issues are dealt with), not concise (i.e., models do contain superfluous matters) and unessential models (i.e., the models do not only model the essence, according to the system category, but focus also on realisation and implementation issues).

The advantages of implementing the Ψ-theory as e.g., in the DEMO methodology, directed us to the hypotheses, that the Ψ-theory can lead us to general guidelines supporting the reverse engineering of organisations allowing for accomplishing enterprise information system engineering. Such guidelines may then help us to make enterprise engineering approaches, not based on theory but widely used in practice, theoretically sound, or to allow for the coupling of different types of models in a theoretically sound way. Gehlert et al. (2009) also stress the necessity for integrating theories into the design research process. As stated in Winter (2008) 'it is important to understand the artefact types of information system research not as separate concepts, but as an interdependent system. Chmielewicz's (1994) classification of research approaches in social sciences may serve as a foundation to explain such dependencies. Chmielewicz differentiates between ontology building, theory building, technology, and philosophy. The respective artefact types are concepts, cause-effect relations, means-ends relations, and normative statements. Illustrations of his taxonomy

usually use the pyramid metaphor: Applicable ontology and meta models constitute the foundation for formulating theories. Valid theories should constitute the foundation for the design of useful artefacts. A "technology", which has been systematically developed and is theory-based then constitutes the foundation for choosing desirable ends, i.e., for normative actions'. By linking theory with artefact construction, we aim at providing a coherent, consistent, and comprehensible foundation for actual solution engineering in practice.

3 Research Question and Notion of Reverse Engineering

Before detailing the research question, we would like to explain the notions of *reverse engineering* that we apply in this research. In any design situation, two distinct systems are involved, called the *using system* and the *object system*. The object system is the system to be developed. When deployed, it supports the using system. Next, in designing a system (of any kind) both the functional and the constructional perspective on systems are relevant (Dietz and Albani 2005). Figure 1 exhibits the basic steps and context of the process of developing the object system, collectively called the *Generic System Development Process* (Dietz 2008).

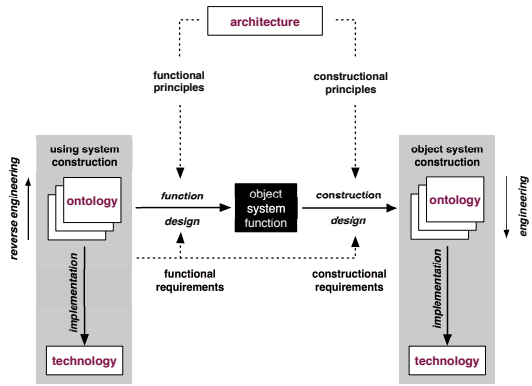

Figure 1: Generic System Development Process (Dietz 2008)

The using system could e.g., be an enterprise, and the object system some enterprise information system, supporting the business activities executed within the enterprise. The starting point is the need by the using system of a supporting system (the object system). Using systems include all stakeholders that contribute to the requirements, therefore include also the owners of the object system. By nature, the need for a supporting system stems from the construction of the using system. Ideally, the basis for the design phase is the *ontological model* of the using system (Dietz 2006a), which is the fully implementation independent constructional model of the enterprise. The process of reconstructing the ontological model of the using system from its implementation model is called *reverse engineering*. The *design phase* starts with designing the function of the object system, expressed in a black-box model of the system. This means that the specific function of the object system does not contain any information about the construction of the object system. There are two main inputs for this step. One is the set of functional requirements stemming from the needs by the using system. These requirements regard the required business services and are for a specific system. The other main input is the set of functional principles that apply to the design of the object system. Indeed, principles are also requirements, but general requirements. They are part of the architecture that is applicable to the class of systems to which the object system belongs (Note. We apply here the prescriptive notion of architecture, as proposed in Hoogervorst 2004). The next basic design step is the design of the construction of the object system. There are two main inputs for this design step, in addition to the designed function of the system, i.e., the black-box model that is arrived at in the first design step. One input is the constructional (often also called non-functional) requirements. The

other one is the set of constructional principles that apply to the design of the object system. They constitute the other part of the architecture that is applicable to the class of systems to which the object system belongs. A thorough analysis of the resulting white-box model must guarantee that building the information system is feasible, given the available technology. The actual process of designing is not one (large) function design step, followed by one (large) construction design step, but rather a sequence of (small) alternating analysis (function design) and synthesis (construction design) steps. After having designed a system, it has to be engineered. Engineering consists basically of producing a coherent and consistent ordered set of white-box models of the object system. The 'lowest' one is commonly called the implementation model. This model can straightforwardly be implemented on an appropriate technological platform. By implementation is understood the assignment of technological means to the constructional elements in the implementation model (Dietz 2008, p. 44). Once implemented properly, the system can be put into operation. The technology needed to implement the constructional elements is actor technology, coordination technology, and production technology. According to Dietz (2006a, pp. 75-77) the only possible actor technology is human beings, since they are the only 'things' that are able to enter into and comply with commitments. Also for immaterial production acts, like judging and deciding, the only available production technology is human beings. Of course, they can be supported by information systems that provide them with the necessary information. Regarding material production acts, like manufacturing things, the basic production technology consists of the manual skills of human being, however they can make use of a numerous of supporting systems. If we look at the communication technology, there is a wealth of technology to record and transmit spoken and written sentences, ranging from voice communication (e.g., face-to-face, telephone, or skype), to text communication (e.g., by postal mail or electronic documents and Internet) (Dietz 2006a, p. 76). If we take e.g., an implementation model, where the technology to be applied is information technology, then the implementation model is a computer program in a programming language that can be compiled or interpreted by the platform on which the object system is going to be operational. However, the 'highest' model, the ontological model or ontology of the object system, is fully independent of its implementation and only shows the essential features of the system.

The questions we are dealing with in the research presented in this paper concern the *reverse engineering* phase as shown in Fig. 1. The goal of our research is to provide guidance to modellers for arriving at the ontological model of an organisation by means of prescriptive guidelines posed for a specific class of problems. The research question can therefore be formulated as follows:

Which prescriptive guidelines support the reverse engineering process of an organisation, given the requirement that two types of enterprise models, namely DEMO and ARIS models, have to be coupled in order to derive at the ontological model of an organisation?

4 The Ψ-Theory

The goal of the Ψ-theory is to understand how the essence of an organisation can be extracted from its actual appearance. The Ψ-theory has been formulated in the tradition of Systemic Ontology, which is very close to mathematics and logic. Therefore the terms 'axiom' and 'theorem' have been chosen in the theory. An axiom has to be understood as a basic assumption about how things are that is evidently true but cannot be proven true in a formal way. A theorem has to be understood as a corollary of a set of axioms

(or other theorems). The Ψ-theory consists of 4 axioms and one theorem and is described in detail in Dietz (2006a). In this paper a short summary of the axioms and the theorem is provided.

4.1 Operation Axiom

The operation axiom states that actors perform two kinds of acts, *production acts* and *coordination acts*. By performing production acts (P-acts) the actors contribute by bringing about the function (by means of goods or services) of the enterprise to its environment. A production act has a definitive result, namely a *production fact*. By performing coordination acts (C-acts) the actors enter into and comply with commitments towards each other regarding the performance of production acts. The definitive result of a coordination act is a *coordination fact*. A subject in its fulfilment of an actor role is called an actor. Actor roles are elementary chunks of authority and responsibility. According to the operation axiom two worlds are distinguished, the *production world* (P-World) and the *coordination world* (C-World). A state of the P-world is a set of P-facts, and a state of the C-world is a set of C-facts. Figure 2 exhibits the operation axiom graphically.

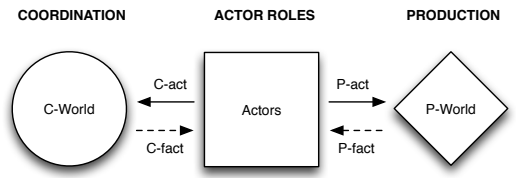

Figure 2: Operation Axiom (Dietz 2006a)

The symbol for coordination is a disk, the symbol for production is a diamond and the symbol for actor roles is a box. Plain arrows indicate the performance of a C-act or P-act respectively. The dashed arrows express that actors take account of the states of the worlds (P-world and C-world respectively) when being active.

4.2 Transaction Axiom

The transaction axiom deals with the question of how production and coordination acts are related to each other. It states that the relationship between coordination and production acts can be considered as paths through some generic coordination pattern. That means that coordination acts are performed in universal patterns. These patterns are called *transactions*. Transactions always involve two actor roles, the *initiator* and the *executor* actor role, and are aimed at achieving a particular result, namely the creation of a production fact. A transaction evolves in three phases (see Fig. 3), the *order phase*, the *execution phase* and the *result phase*.

In the order phase the initiator and the executor work to reach agreement about the intended result of the transaction, namely the production fact the executor is going to create. In the execution phase this result is actually created by the executor, and in the result phase the actors involved try to reach agreement about the result that has actually been produced. There exist three patterns of a transaction. The so-called *basic pattern* of a transaction consists of the coordination steps *request* (rq), *promise* (pm), *state* (st) and *accept* (ac). An example of applying the basic transaction pattern to the completion of a purchase order looks as follows:

(1) Customer *requests* Completer to *complete the purchase*

(2) Completer *promises* Customer to *complete the purchase*

(3) <actual completion of purchase>

(4) Completer *states* Customer to have *completed the purchase*

(5) Customer *accepts* from Completer that he has *completed the purchase* (received goods and performed payment correspond to customer's expectations)

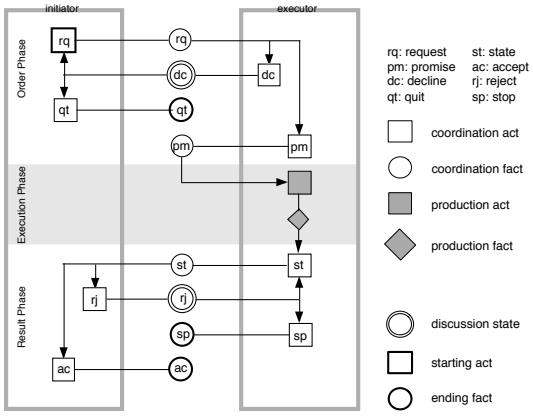

Figure 3: Standard Transaction Pattern (Dietz 2006a)

This typical scenario of the basic transaction pattern allows the completer (executor) only to promise the request and the customer (initiator) only to accept the completion of the purchase (P). But what happens in reality if e.g., the customer has not received the correct goods? He would then like to reject the state of the purchase and expect the completer to send him the right goods. Also the completer may want to decline the request to complete a purchase. This could e.g., be due to the request to deliver the purchased goods in a foreign country. That means that the two actors may dissent in two of the states, namely the requested state and the stated state. Allowing also to *decline* a request posed by the initiator and to reject a state made by the executor, results in the *standard transaction pattern*, shown in Fig. 3. In addition to the request, promise, state and accept acts, the standard transaction pattern includes also the decline (dc), quit (qt), reject (rj) and stop (sp) acts. The decline of a request or the rejection of a state may result in a discussion state (indicated in Fig. 3 with a double disk). As stated in Dietz (2006a), the corresponding actors need then to discuss in order to reach consensus.

In practice, it is also quite common that either the initiator or the executor may want to revoke an act. This is accommodated by the option to cancel any C-act in the basic transaction pattern. The resulting transaction pattern is the *complete transaction pattern*, which consists of the standard transaction pattern and four cancellation patterns, namely for the request, promise, state and accept. Details about the complete transaction pattern can be found in Dietz (2006a).

4.3 Composition Axiom

We have learned that the result of every successful transaction is the creation of a P-fact. The composition axiom provides an answer to the question of how such P-facts are interrelated. The composition axiom expresses that every transaction is enclosed in some other transaction, or is a customer transaction of the organisation under consideration, or is a self-activation transaction. The composition of the production facts for the completion of a purchase order is shown in Fig. 4.

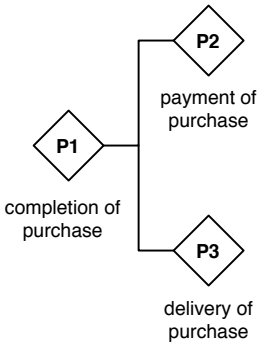

Figure 4: Composition Axiom according to (Dietz 2006a)

In order to be able to complete a purchase order for specific goods (P1: completion of purchase), it is necessary that the goods are paid (P2: payment of purchase) and that the goods are successfully delivered to the customer (P3: delivery of purchase). P1 is a customer transaction, whereas P2 and P3 are enclosed in another transaction, namely in the P1 transaction. Self-activation transactions

are used e.g., for periodic activities, as for all control activities.

As stated in Dietz (2006a), the composition axiom provides the basis for a well-founded definition of the notion of business processes, which states that a *business process* is a collection of causally related transaction types, such that the starting step is either a request performed by an actor role in the environment (external activation) or a request by an internal actor role to itself (self-activation). Every transaction type is represented by the complete transaction pattern.

4.4 Distinction Axiom

The last axiom, the distinction axiom, states that there are three distinct *human abilities* that play a role in the execution of production and coordination acts, namely the *performa*, *informa* and *forma* human abilities (see Fig. 5). How are these human abilities relevant for coordination acts on the one hand and production acts on the other hand?

COORDINATION	ACTOR	PRODUCTION
exposing commitment (as performer) evoking commitment (as addressee)	performa	ontological production (deciding, judging)
expressing thought (formulating) educing thought (interpreting)	informa	infological production (reproducing, deducing, reasoning, computing, etc.)
uttering data (speaking, writing) perceiving data (listening, reading)	forma	datalogical production (storing, transmitting, copying, destroying etc.)

Figure 5: Distinction Axiom (Dietz 2006a)

The *forma ability* deals with the form aspects of communication and information. Applying this to coordination acts means that actors should have a way to utter and perceive information. Information should be expressed in a particular language or code scheme that both the initiator and the executor of a transaction understand. This is also known as syntactic (or significational) understanding. Applying the forma ability to production acts means that one is concerned with the form aspects of information in terms of e.g., information transmission copying and storage. These types of production acts are also known as *datalogical acts*. The *informa ability* concerns the content aspects of information and communication. That means that we are now dealing with communication and information while fully abstracting from the form aspects. In order to communicate, the initiator should formulate information in a way that the executor can interpret. In other words, the initiator and the executor should semantically be in agreement with each other and share the same thoughts. This is also called intellectual understanding. Concerning the informa ability in production means that information can be computed, reasoned or deduced. Those activities are known as *infological acts*. The *performa ability* states that new information and knowledge can be created through communication between the initiator and executor. Looking at coordination acts, this means that actors can expose and evoke commitments and it indicates social understanding between the initiator and executor. Looking at production acts the performa ability concerns the deciding, judging or creating new material things such as products. This is what we call *ontological acts*. Dietz considers the performa ability as the *essential* human ability for doing business of any kind. Transactions that contain datalogical acts are called *datalogical transactions* (D-transactions), if they contain infological acts than they are called *infological transactions* (I-transactions) and if they contain ontological acts they are called *ontological transactions* (B-transactions).

4.5 Organisation Theorem

The axioms presented above serve to achieve the overall goal of the Ψ-theory, namely to extract the essence of an organisation from its actual appearance. The organisation theorem states that the organisation of an enterprise is

a heterogeneous system that is constituted as the layered integration of three homogeneous systems: the B-organization (from Business), the I-organization (from Intellect), and the D-organization (from Document), see Fig. 6.

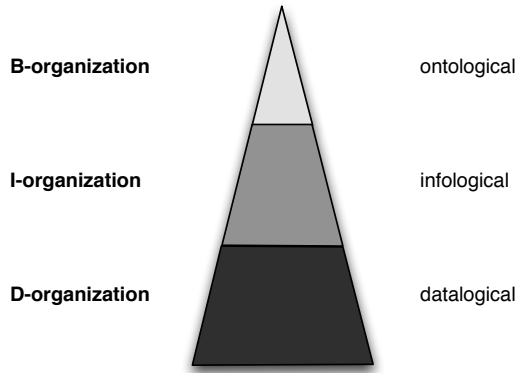

Figure 6: Organisation Theorem (Dietz 2006a)

The three homogeneous systems are all in the category of social systems. Concerning the coordination they are all similar, since the actors in all three systems are entering into and comply with commitments. They differ only in the kind of production acts they perform. Actors in the B-organization directly contribute to the business of the organization by performing B-transactions (executing ontological production acts). Actors in the I-organization support the actors in the B-organization by performing e.g., computational acts for changing information and knowledge, but do not bring about any new business specific production facts. I-actors perform I-transactions. The D-organization is concerned with the documentation of information (storing, copying, destroying) taking into account the form of information. Actors in the D-organization support the actors in the I-organization by executing D-transactions.

5 Prescriptive Guidelines for Reverse Engineering

The Ψ-theory allows us to define some principles for reverse engineering. These principles can directly be deduced from the axioms and the theorem of the Ψ-theory and can be applied in any enterprise engineering methodology to model the ontological model of an enterprise. They can be summarised as follows:

- Organisational units assignments in enterprise models should be in accordance with the actor role as defined in the operation and the transaction axiom of the Ψ-theory.
- The notion of process steps as defined in the transaction axiom of the Ψ-theory should be reflected in the process logic of the enterprise models. When e.g., applying the basic transaction pattern, the resulting process steps would be request, promise, execute, state and accept.
- The relationships between the different transactions should be modelled in accordance with the composition axiom of the Ψ-theory (customer, enclosed or self-activating transaction).
- The different aspect organisations (B-, I- and D-organisation) should be modelled in accordance with the distinction axiom of the Ψ-theory.

Based on these principles specific guidelines are derived in considering the goal of eliminating inconsistencies when coupling two types of enterprise engineering models, namely DEMO and ARIS models. With inconsistency we mean the internal contradiction and the missing logical coherence among models or parts of models. In order to arrive at an integrated ontological model of the using system, a case study at a governmental agency in the Netherlands was conducted. The main focus of the case study was to elaborate guidelines for the specific situation at hand. From this case study, we were able to derive general prescriptive guidelines for any situation where DEMO models and ARIS models need to be coupled while eliminating inconsistencies. Since ARIS is not based on a theory, whereas DEMO is, the main guidelines describe how applying the

reverse engineering principles based on the Ψ-theory can substantiate ARIS models. The resulting guidelines are formulated as follows:

1. Find related DEMO Actor Transaction Diagrams and ARIS Functional Decomposition Diagrams
2. From each DEMO Actor Transaction Diagram generate a tree
   a. Use the designation of the DEMO Actor Transaction Diagram to name the Functional Decomposition Diagram (root node in the tree)
   b. Put every DEMO transaction initiated by the customer of the organisation under consideration into a branch of the tree
   c. Put every transaction initiated by the organisation under consideration into a node of the corresponding branch of the tree
3. If the trees are not identical, make the union of the trees
4. Make the tree ontological
   a. Remove non-ontological nodes, i.e., remove infological and datalogical nodes
   b. Add missing ontological nodes
   c. Improve labels to accentuate their ontological meaning
   d. Provide transaction numbers to unnumbered nodes
   e. Build up the Transaction Result Table
5. Trim the tree, i.e., make sure the resulting tree fulfils the required business goal
6. Reverse engineer the tree into a DEMO Actor Transaction Diagram
7. For the root node in the tree create an Event Driven Process Chain
   a. For each node in the tree create an Event Driven Process Chain fragment according to the standard transaction pattern
   b. For each Event Driven Process Chain fragment of a sub-node in the tree create a relationship to the Event Driven Process Chain fragment of the super-node according to the composition axiom
8. Assign organisational units to the process steps
   a. Ensure that the role that makes the request is the same role that accepts the request
   b. Ensure that the role that promises the result is the role that executes and states
   c. Ensure that the organisational unit assigned to the roles is authorised to take on those roles

The presented guidelines are prescriptive guidelines that lead to exactly one ARIS model based on the corresponding DEMO model. We can also call them hard guidelines, since they lead to unambiguous results, since they are based on the Ψ-theory as follows:

The guidelines 1 and 2 relate to the composition axiom, which expresses that every transaction is enclosed in some other transaction, or is a customer transaction of the organisation under consideration, or is a self-activation transaction. This allows for the designation of the initial tree. Guidelines 3 to 5 relate to the organisation theorem. Since we aim at producing an ontological model of the organisation infological and datalogical transactions (I- and D- transactions) are removed from the initial tree and missing ontological transactions (B-transactions) are added. The resulting tree is the basis for the coupling of the two types of models. In guideline 6 the tree is used to create the DEMO Actor Transaction Diagram and in the guidelines 7 and 8 the tree is used to create an Event Driven Process Chain. Since DEMO is based on the Ψ-theory, no further explanation for the creation of the Actor Transaction Diagram is given here (details about how to construct such a model can be found in Dietz 2006a, pp. 160-171). However, for the creation of the Event Driven Process Chain the guidelines

7 and 8 are necessary. Guideline 7 and 8 are based on the transaction and on the composition axioms. By applying those axioms to each node (i.e., each transaction) in the tree, all relevant coordination and production acts are modelled and set in relation to each other (see guideline 7). Additionally, since every transaction involves two actor roles, the initiator and the executor (as defined in the transaction axiom), organisational units can easily be assigned to the single process steps as described in guideline 8.

The result of applying steps 1-6 to a very simple example case, the shop case, is shown in Fig. 7. Left of Fig. 7, the DEMO Actor Transaction Diagram (ATD) is shown with the corresponding Transaction Result Table (TRT) listing the result types of the transaction types visible in the ATD. The ontological transactions of the shop case are the completion of the purchase order (T01), which is composed of the transactions for the delivery (T03) and the payment (T02) of the purchase order. For details concerning the notation of the DEMO models we refer to Dietz (2006a). Right of Fig. 7 shows an ARIS Functional Decomposition Diagram (FDD) for the shop. When applying the guidelines 1-6 the ARIS FDD results, shown in the middle of Fig. 7. It can be seen that the function 'availability checking' has disappeared. This is due to step 4a of the guidelines, prescribing that non-ontological nodes need to be removed. Since 'availability checking' is an infological transaction (or function as called in the FDD), it has been removed. Additionally, it can be seen that transaction numbers are given to the single nodes in the tree and that the results of the transactions/functions are described in detail in the TRT listed below the FDD.

Figure 8 shows an extract of the resulting Event Driven Process Chain (EPC) of applying steps 7 and 8 of the guidelines to the shop example case of Fig. 7. We took the node 'T01 completion' and created the EPC fragment according to the standard transaction pattern. The corresponding EPC fragment is listed on the right of Fig. 8, whereas the acts and facts of the standard transaction pattern are shown on the left of Fig. 8. The actor role responsible for the execution of the 'T01 completion' transaction/function is the 'A01 completer'. The requests for completing the purchase order come from the 'customer' actor role. For the 'execute purchase P' activity, the relationship with the sub-nodes of the tree (T02 payment and T03 delivery) is made by means of a semantic refinement. This helps to keep a clear view of the whole process and allows for implementation of the composition axiom.

When applying the prescriptive guidelines introduced above to a real case, an enormous reduction in complexity can be reached. In both diagram types only the ontological, and therefore essential functions/activities are modelled leaving out all infological and datalogical ones. Additionally, a clarification of the functionality listed in the FDD is achieved by means of clearly describing its resulting types within a TRT. The application of steps 7 and 8 of the prescriptive guidelines results in giving a transparent structure to the EPC. Having based the EPC e.g., on the standard transaction pattern, implies that the model is complete, containing all essential activities. Additionally, tacitly performed C-acts have been traced and made explicit in the EPC. By assigning clear responsibilities to the activities by means of actor roles, a complete implementation independent model results. Following the guidelines it becomes also evident where to use e.g., the ARIS concept of 'semantical refinement', namely exactly there where the composition axiom holds.

6 Case Study and Result Evaluation

The presented research was conducted in conjunction with the department 'Rijkswater-

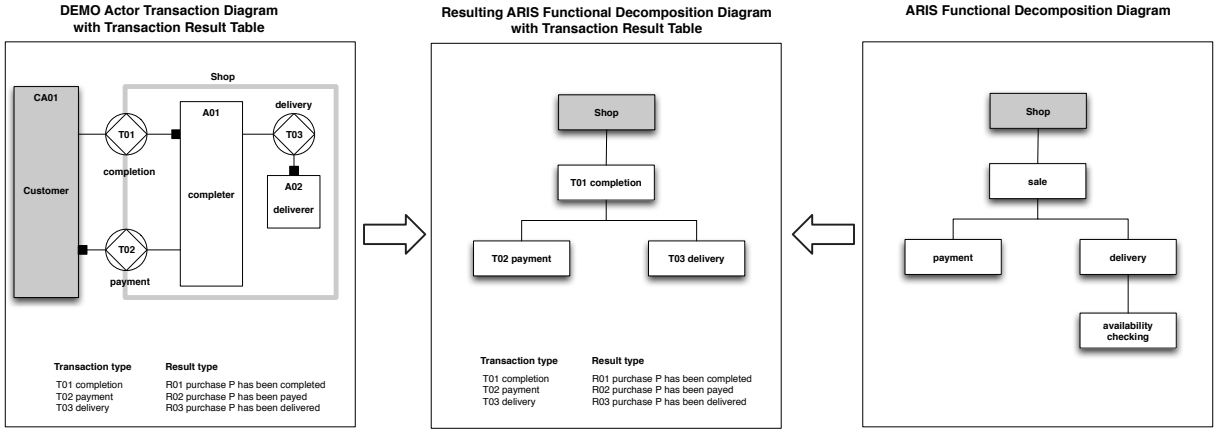

Figure 7: Result of applying steps 1-6 of the prescriptive guidelines to a simple example case

staat (RWS)' of 'The Ministerie van Verkeer en Waterstaat (V&W)', a ministry of the national government in the Netherlands. RWS is concerned with the practical execution of the ministry's directives, mainly the construction and maintenance of roads and waterways. The details of the conducted research can be found in Strijdhaftig (2008). Two of the national service centres within RWS were of importance for the research project. One was the Data and ICT service centre with the Enterprise Architecture group making use of service provision models based on the DEMO methodology. The other was the Management Organisation of Processes and Systems, making daily use of the work processes uniform within the whole organisation. These models are produced using ARIS. Before starting the project, an analysis was executed concerning the benefit that would result from the investigation of coupling both types of models. As outlined in Nagel (1990) several methods for the estimation of the benefit of IT projects exist. In the said evaluation, a cost analysis was not in the focus. The focus was set on the quality of the models. The possibility to drop one of the two types of models in order to continue with only one model type was investigated. Due to the very specific expression power of both model types and the extensive use of these specific model qualities in the single service centres, the analysis resulted in the necessity to maintain both types of models. Because of the said success it is expected that both model types will continue to exist alongside each other for the foreseeable future. But since both types of models deal with content of the same organisation, it was expected that connecting them could be very beneficial for both national service centres since quality improvements can be achieved and consistency problems can be solved. The investigation for the coupling of both types of models in a consistent way was therefore necessary. Looking only at the notations and types of the different ARIS and DEMO models, one would assume that a simple coupling of the two would be no problem, see Tab. 1.

A simple coupling, as assumed by RWS to be feasible, turned out not to be possible due to the discrepancies in modelling. The main reason was that ARIS is not based on theory (as many other enterprise modelling methodologies) and that no rules exist in order to guide the application of the ARIS methodology. The ARIS models were therefore unstructured, leaving a lot of freedom for interpretation, not complete and containing many (nonontological) activities. Many methodologies, like ARIS, lack a thorough and precise underlying theory for understanding. As a ser-

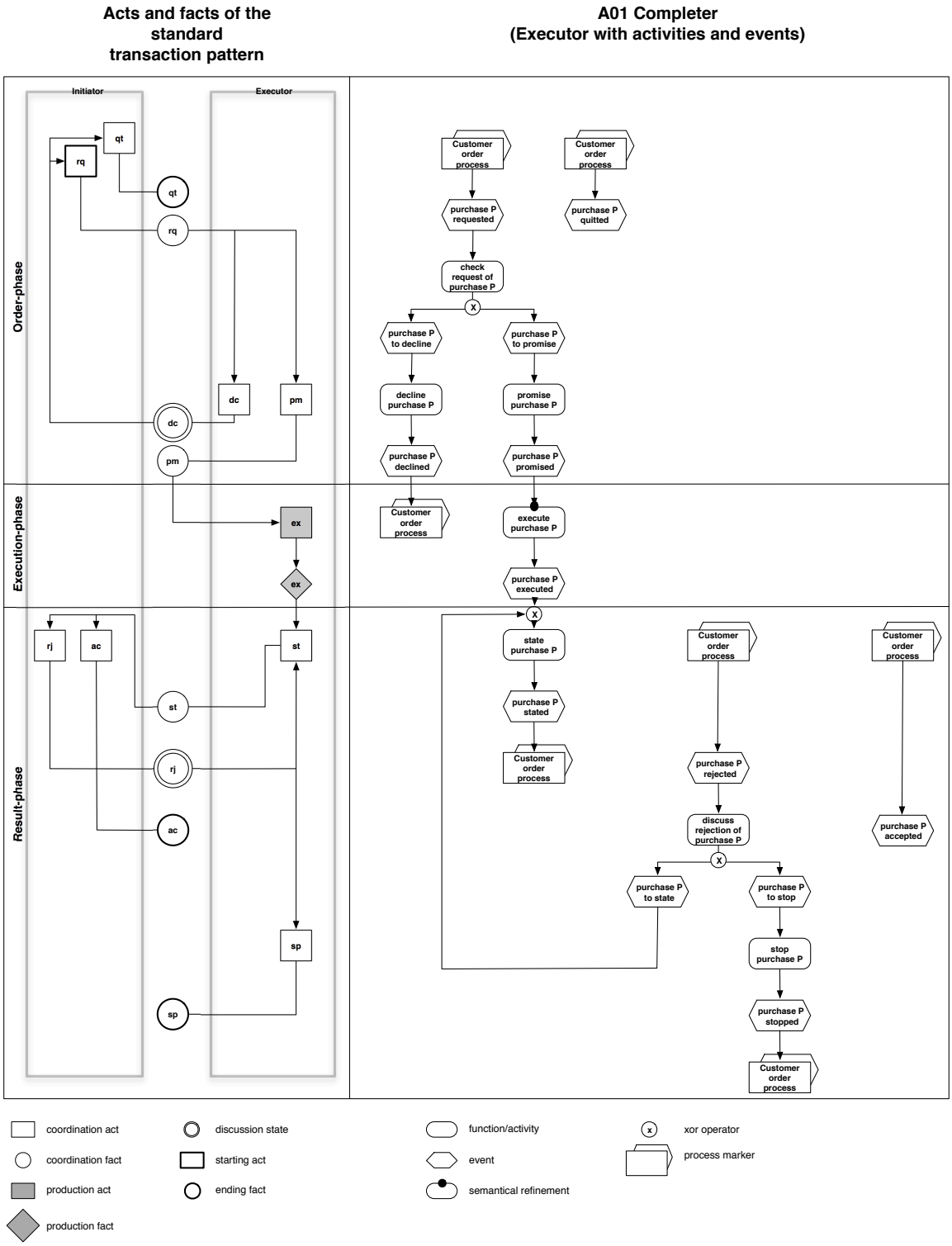

Figure 8: Result of applying steps 7-8 of the prescriptive guidelines to a simple example case

ious consequence, it remains unclear what the basic notions are exactly. Therefore, Tab. 1 is actually a kind of best guess.

The lack of an underlying theory for ARIS led to the investigation of deriving guidelines for the coupling of DEMO and ARIS models based on the Ψ-theory. The resulting guidelines, described in detail in Strijdhaftig (2008), are specific for the situation at RWS. That means they include several rules fitting to the specific ARIS models at RWS. But the research at RWS allowed us to abstract from the situational specific guidelines and gain insights for more general guidelines allowing for the coupling of any DEMO and ARIS model by eliminating inconsistencies between the two types of models.

The prescriptive guidelines have been applied to real case models at RWS and a profound evaluation has taken place. The evaluation focused on 1) the opportunities and barriers when coupling the DEMO and ARIS models at RWS, 2) the guidelines through which the coupling of the two types of models was established and 3) further actions to be taken when applying the guidelines to all relevant models at RWS. The evaluation aspects were addressed in a session using a Group Decision Support System (GDSS) with eleven participants; among those were DEMO specialists, ARIS specialists, professionals of both service centres, and people in the position to influence the course of the project. The use of GDSS was important in order to have an evaluation with the following characteristics:

Anonymity All inputs to the GDSS are anonym. That allows for an evaluation of ideas independent of the person who brings it up.

Parallel input and communication Participants can enter their comments and ideas at the same time, which results into a drastic shortening of discussion time.

Group memory All inputs are stored and can be retrieved later in order to finalise the evaluation.

Visual consensus The degree of agreement and disagreement can be visualised in real-time.

Immediate reporting The system can generate reports of the meeting immediately after the meeting is concluded.

The details of the questions, answers and discussions of the session can be read in Strijdhaftig (2008). In this paper we summarise only the most relevant results concerning the applied guidelines, opportunities and barriers relevant for the coupling. Questions were posed concerning the guidelines 2, 4, 7 and 8 (see guidelines on page 14), which are the most relevant guidelines of our theory. Just to summarise: guideline 2 concerns the generation of the FDD based on the DEMO transactions; guideline 4 prescribes to make the tree ontological in removing infological and datalogical functions; guideline 7 explains how to create EPC fragments based on the standard transaction pattern (7a) and how they are combined to an EPC by means of the composition axiom (7b). How to assign organisational units to process steps is prescribed in guideline 8. The conclusion that can be drawn from the evaluation is that the acceptability of guidelines 2, 7b and 8 was very high. Guideline 4 and 7a were disputed significantly despite its overall acceptability. Of course the application of the standard transaction pattern in every EPC fragment results in quite detailed processes. However, the clear structure of the EPC resulting from the application of the guidelines was obviously seen as one of the most relevant contributions of the project. But one of the main impacts to reach such a clear structure is of course the notion of the transaction pattern. This was also recognised by the participants of the evaluation session and therefore they accepted guideline 7a after some dispute. The discussion about guideline 4 is also absolutely

Table 1: DEMO and ARIS: Models and types correlations (Adapted from Strijdhaftig 2008)

ARIS	DEMO	Reasoning
Organisation View	Construction Model	The organisation view in ARIS and the construction model of DEMO hold information about who is responsible in the organisation. While the construction model shows implementation independent roles and transactions, the organisation view shows departments, positions, roles and persons.
Function View	Construction Model	The function view in ARIS specifies the activities that are performed as part of a task. The construction model of DEMO specifies the transactions that take place in the organisation. A transaction in DEMO is a sequence of acts and events arranged according to the transaction pattern.
Data View	State Model	In ARIS the data view models the interrelationships between relevant information entities (according to the Entity Relationship Diagram), whereas in DEMO the state model specifies the object types and the state space (i.e., the set of allowable states) of both the production world and the coordination world of the enterprise. Otherwise said it contains the conceptual model of all objects and facts that are either produced or used.
Control View	Process and Action Model	The control view in ARIS shows a process as a string of alternating activities and events connected by control flow relationships. This is similar to the transaction pattern of the process model in DEMO. Both alternate between action and result and both show the conditional and causal relationships between activities. The difference is that the transaction pattern has a specific set of activities and events and the control view allows arbitrary activities and events. The control view also contains the guidelines that should be followed when transitioning from state to state. In DEMO these guidelines are specified in the action model.
Function	Act	An act is an atomic unit of activity, of which the effect is the creation of a factum. In the Ψ-theory, two kinds of acts are distinguished: coordination acts and production acts. An act is performed by an actor. Since the act is an atomic unit of activity in DEMO and the function is the unit of activity in ARIS, DEMO acts are comparable to ARIS functions.
Event	Event	Events are unique and are defined as the occurrence of a transition. A transition is a change of state of a world. Since a DEMO event is the occurrence of a fact that results from an act, and an ARIS event is the result of a function, the DEMO event is comparable to the ARIS event. In DEMO however, facta and events are labelled differently from each other. A factum is labelled as something that happened at a certain point in time, whereas the event is the occurrence of the transition. In ARIS the event is labelled as something that happened at a certain point in time combined with the condition under which it is relevant for further action.
None	Fact	A fact is an elementary state of affairs in a world. A fact is a result of an act. A fact starts to exist at the moment that the act is performed. There is no ARIS equivalent of the DEMO fact type.
Functions and Events	Transaction	A transaction in DEMO is a sequence of acts and events arranged according to the transaction pattern. It goes off in three consecutive phases: the order phase, the execution phase and the result phase. Therefore, if directly converted, a transaction would map to a sequence of functions and events in ARIS.
Organisational Unit	Actor Role	The actor role in DEMO is the unit of authority, responsibility, and competence. The ARIS organisational unit can substitute the DEMO actor role, but the organisational unit cannot always be substituted by an actor role. This is because the actor role is not directly equivalent to entities like departments and managers.

Processing Input	Information Bank	The information bank in DEMO is a conceptual store of coordination or production facts. The information bank may only be accessed by actor roles with the proper authorisation. The equivalent of those in ARIS is the processing input since it contains information necessary to execute a function.
Processor	None	Implementation issues such as systems and applications are not provided in DEMO.
Input/Output	Object Class	An object in DEMO is an identifiable individual thing. If the form of an object conforms to a type T, then there is a concept C that is an instance of T. An object class is the extension of a type. Object classes are related to each other in various ways. Whether something is being used as input or output is not the focus in DEMO.

understandable, since the distinction between ontological, infological and datalogical transactions is a complete new way of thinking and needs some time to get familiar with. But the immense reduction in complexity that arose from looking only at ontological transactions convinced people to accept guideline 4 as an important guideline. From the evaluation it resulted also that main opportunities of applying the guidelines are improvements in quality and transparency of the ARIS models. Main barriers of the application of the guidelines are the high costs and the time needed to adapt existing ARIS models as well as the necessity of having people to be trained in the DEMO methodology.

7 Applicability of the Approach

As said above, the prescriptive guidelines have been applied to real case models at RWS and the profound evaluation showed that an improvement in quality and transparency of the models, e.g., by means of a reduction in complexity, could be reached. With this example, we showed again, which advantages result from applying the axioms and the theorem of the Ψ-theory to a design and engineering methodology as e.g., ARIS. The further application of the prescriptive guidelines as introduced above to additional ARIS models still need to be investigated. However, over the last years the successful application of a methodology implementing the Ψ-theory, as e.g., the DEMO methodology, has been shown in several large-scale projects as well as in several smaller ones. Examples of large-scale projects – just to mention a few – are RWS, Dutch Telecom, Rotterdam Police Force, or Air France and KLM. Smaller projects have been executed e.g., at the Conciliation Board of Consumers in the Netherlands or Alcatel-Lucent in Switzerland. A summary of several projects can be found e.g., in Mulder (2006) or Op't Land (2008). The purpose of the different projects is diverse. At RWS the DEMO models have been used to support the line of reasoning when taking strategic decisions regarding the construction, management, development and maintenance of the main infrastructure networks in the Netherlands (Proper and Op't Land 2010). In the case of the Conciliation Board of Consumers DEMO was applied as a means for business process optimisation and information system development (Reijswoud et al. 1999). Additional work on finding key concepts to link agile enterprises with agile automated information systems resulted in combining DEMO and Normalised Systems (Krouwel and Op't Land 2011). At the Command and Control Support Centre of the Dutch Ministry of Defence DEMO and Normalised Systems were then used to build an information system (Op't Land et al. 2011). From these results and because Normalised Systems have a continuous link from enterprise modelling to software development and a short feedback loop from system development to enterprise modelling, DEMO and Normal-

ised systems are considered as key enabler for agile enterprise engineering. In the case of Air France and KLM, the finding and testing of method components for deciding on and implementing organisational splits and mergers was investigated. The DEMO Construction Model appeared to be the first neutral and shared language for describing the essence of the business. Also the results needed for decision-making (a) were experienced as a necessary and sufficient validation of operational integrity and (b) were delivered fast, yielding a high Return On Modelling Effort (Op't Land et al. 2009).

Seeing all the advantages reached in the said projects it is expected that the same advantages result also when applying other methodologies implementing the Ψ-theory, as e.g., ARIS as introduced in this paper, to real-world projects.

8 Summary and Outlook

The Ψ-theory underlies the notion of Enterprise Ontology. DEMO implementing the Ψ-theory is one of the few enterprise engineering methodologies based on theory. In this paper we first elaborated on the notion of reverse engineering, design and engineering and presented then the principles for reverse engineering an organisation, derived from the Ψ-theory. In addition we presented guidelines for a specific requirement, which was dealing with the elimination of inconsistencies when coupling two types of enterprise engineering models, namely DEMO and ARIS models. The research goes beyond existing work, where either too general or too specific recommendations are given – if prescriptive procedural recommendations are provided at all. Much of the existing work on enterprise information systems development is focusing on problem (or solution) representation ('enterprise modelling') and not on problem solution principles and guidelines.

Based on the presented principles and the prescriptive guidelines, a case study was presented where the coupling of ARIS and DEMO models at Rijkswaterstaat, a Department of the 'Ministerie van Verkeer en Waterstaat' of the national government in the Netherlands, has been conducted and evaluated. The evaluation focused on the opportunities and barriers and the feasibility and value of applying the guidelines. By applying the presented guidelines, a better transparency could be achieved and the ARIS models could be improved drastically having e.g., the transaction structure made explicit. Additionally, a significant reduction in complexity could be reached while focusing on ontological transactions, leaving out any infological or datalogical transactions. Of course, barriers of the presented approach are the time needed and the costs that arise in order to adapt all existing models. DEMO provides a new way of modelling organisations, however, the way of thinking and looking at organisations, as expressed with the Ψ-theory, needs to be understood in depth. Only with a deep knowledge of the DEMO methodology and its underlying theory the desired results as e.g., complexity reduction or high transparency in communication can be reached. DEMO provides very specific concepts to reach these goals and therefore to extract the essence of an organisation from its actual appearance. Examples are the transaction concept, which is defined in the transaction axiom of the Ψ-theory, or the concept of distinguishing between performa, informa and forma human abilities, defined in the distinction axiom of the Ψ-theory. The specific concepts defined in the Ψ-theory distinguish DEMO from other enterprise engineering methodologies. However, single concepts, as e.g., transactions, can also be found in other enterprise engineering methodologies, even though in different ways as in DEMO. E.g., Ferstl and Sinz (1995) use transactions in their Semantic Object Model (SOM) approach to model business processes.

The concept of encapsulating tightly coupled tasks by an object and loosely coupling the tasks of different objects via transactions is a key feature of the object-oriented characteristic of the SOM methodology and clearly differs from the transaction concept of DEMO.

The uniqueness of DEMO has not only advantages. The DEMO methodology is used and debated by only a small community of researchers and practitioners so far. The exchange with other communities however, is absolutely necessary and desired in order to improve DEMO and make it widely accepted. One of the biggest disadvantages of DEMO is the notation. The notation as provided by the DEMO models is not self-explaining and differs quite a bit from known modelling notations. It therefore needs an additional investment for reaching the desired understanding and being able e.g., to apply the guidelines suggested in this paper for improving existing models. Further, the continuity from the implementation independent models to the implementation specific models is still under development. This restricts the application of DEMO to specific problem areas. DEMO models are e.g., widely used by management, since it allows for understanding the essence of an organisation and taking strategic decisions.

Having presented the principles for the reverse engineering phase, and guidelines for one specific class of problems, namely the coupling of DEMO and ARIS models, additional guidelines need to be derived for different enterprise engineering types of models. The main focus of the future research is in generalising the guidelines defined for the different types of models allowing for a sound and systematic reverse engineering of organisations. Additionally, the focus of the future work should not only be on guidelines for ontological transactions, but also on guidelines to allow for the integration of the infological as well as the datalogical transactions.

9 Acknowledgements

This project was supported by the Swiss National Science Foundation (SNSF).

References

Albani A., Dietz J. L. (2011) Enterprise Ontology Based Development of Information Systems. In: International Journal of Internet and Enterprise Management, Special Issue on Enterprise Systems Modeling and Simulation 7(1), pp. 41–63

Alter S. (2006) The Work System Method. Work System Press, Larkspur

Alter S. (2009) Work System Basics. http://www.stevenalter.com/work-system-basics-2/. Last Access: 12.11.2013

Avison D., Fitzgerald G. (2002) Information Systems Development, Methodologies, Techniques and Tools, 3rd ed. McGraw-Hill, London

Bunge M. (1979) Treatise on Basic Philosophy, A World of Systems Vol. 4. D. Reidel Publishing Company, Dordrecht, The Netherlands

Checkland P. (1999) Systems Thinking, Systems Practice (Includes a 30-year retrospective). John Wiley & Sons., Chichester, UK

Chmielewicz K. (1994) Forschungskonzeptionen der Wirtschaftswissenschaft. C.E. Poeschel, Stuttgart

Crawford D. (ed.) Communications of the ACM – Two Decades of the Language-Action Perspective. 5 Vol. 49. ACM, pp. 44–77

Davenport T. H. (1993) Process Innovation: Reengineering Work Through Information Technology. Harvard Business School Press, Boston, MA, USA

Davenport T. H., Short J. E. (1990) The New Industrial Engineering: Information Technology and Business Process Redesign. In: Sloan Management Review 31(4), pp. 1–31

Dietz J. L. (2006a) Enterprise Ontology – Theory and Methodology. Springer, Berlin, Heidelberg

Dietz J. L. (2006b) The Deep Structure of Business Processes. In: Communications of the ACM 49(5), pp. 58–64

Dietz J. L. (2008) Architecture – Building Strategy into Design. Academic Service, Amersfoort, The Netherlands

Dietz J. L. (2010) Is it Phi Tau Psi or Bullshit? Delft University of Technology, Delft, The Netherlands

Dietz J. L., Albani A. (2005) Basic notions regarding business processes and supporting information systems. In: Requirements Engineering Journal 10(3), pp. 175–183

Dietz J. L., Hoogervorst J. (2007) Enterprise Ontology and Enterprise Architecture, how to let them evolve into effective complementary notions. In: GEAO Journal of Enterprise Architecture 2(1)

Dietz J. L., Hoogervorst J. (2008) Enterprise ontology in enterprise engineering. In: Proceedings of the 2008 ACM symposium on Applied computing. Fortaleza, Ceara, Brazil, pp. 572–579

Dubin R. (1978) Theory Building. Free Press, London

Ferstl O. K., Sinz E. J. (1995) Der Ansatz des Semantischen Objektmodells (SOM) zur Modellierung von Geschäftsprozessen. In: Wirtschaftsinformatik 37(3), pp. 209–220

Flores F., Ludlow J. (1980) Doing and speaking in the office. In: Fick G., Jr. S. R. H. (eds.) Decision Support Systems, Issues and Challenges. Pergamon Press, New York, pp. 95–118

Gehlert A., Schermann M., Pohl K., Krcmar H. (2009) Towards a research method for theory driven design research. In: Hansen H. R., Karagiannis D., Fill H. G. (eds.) Business Services: Konzepte, Technologien, Anwendungen; Proceedings der 9. Internationalen Tagung Wirtschaftsinformatik. Österreichische Computer Gesellschaft, Vienna, Austria, pp. 441–450

Goldkuhl G., Lyytinen K. A. (1982) Language action view of information systems. In: Ginzberg M., Ross C. (eds.) 3rd International conference on information systems. TIMS/SIMS/ACM

Gregor S. (2002) Design Theory in Information Systems. In: Australasian Journal of Information Systems 10(1), pp. 14–22

Hammer M. (1990) Reengineering Work: Don't Automate, Obliterate. In: Harvard Business Review 68(4), pp. 104–112

Hammer M., Champy J. (1993) Reengineering the Corporation – A Manifesto for Business Revolution. Harper Collins Publishers, New York

Hoogervorst J. A. (2004) Enterprise Architecture: enabling integration, agility, and change. In: Journal of Cooperative Information Systems 13(3), pp. 213–233

Ilnseher J. (2007) Trends in Enterprise Modelling Methodologies – Overview and Comparison (in German). University of Augsburg

Krouwel M. R., Op't Land M. (2011) Combining DEMO and Normalized Systems for Developing Agile Enterprise Information Systems. In: Albani A., Dietz J. L., Verelst J. (eds.) 1st Enterprise Engineering Working Conference (EEWC) Vol. LNBIP 79. Springer, Antwerp, Belgium, pp. 31–45

Markus M. L., Majchrazak A., Gasser L. (2002) A design theory for systems that support emergent knowledge processes. In: MIS Quarterly 26(3), pp. 179–212

Mesarovic M., Macko D., Takahara Y. (1970) Theory of Hierarchical, Multilevel Systems. Academic Press, New York

Mulder H. (2006) Rapid Enterprise Design. PhD, Delft University of Technology, The Netherlands

Nagel K. (1990) Nutzen der Informationsverarbeitung Vol. 2. Oldenbourg, München, Wien

Op't Land M. (2008) Applying Architecture and Ontology to the Splitting and Allying

of Enterprises. PhD, Delft University of Technology, The Netherlands

Op't Land M., Zwitzer H., Ensink P., Lebel Q. (2009) Towards a Fast Enterprise Ontology Based Method for Post Merger Integration. In: 24th Annual ACM Symposium on Applied Computing (ACM SAC). Honolulu, Hawaii, U.S.A., pp. 245–252

Op't Land M., Krouwel M. R., van Dipten E., Verelst J. (2011) Exploring Normalized Systems Potential for Dutch MoD's Agility (A Proof of Concept on Flexibility, Time-to-market, Productivity and Quality). In: Harmsen F., Grahlmann K., Proper E. (eds.) The 3rd Practice-driven Research on Enterprise Transformation working conference (PRET). Springer, Luxembourg, pp. 110–121

Österle H., Winter R. (2003) Business Engineering. In: Österle H., Winter R. (eds.) Business Engineering – Auf dem Weg zum Unternehmen des Informationszeitalters. Springer, Berlin

Proper E., Op't Land M. (2010) Lines in the Water – The Line of Reasoning in an Enterprise Engineering Case Study from the Public Sector. In: Harmsen F., Proper E., Schalkwijk F., Barjis J. A., Overbeek S. (eds.) The 2nd Practice-driven Research on Enterprise Transformation working conference (PRET). Springer, Delft, pp. 193–216

Reijswoud V. E., Mulder H., Dietz J. L. (1999) Communicative Action Based Business Process and Information Systems Modelling with DEMO. In: Information Systems Journal 9(2), pp. 117–138

Scheer A.-W. (2000) ARIS – Business Process Modeling, 3rd ed. Springer, Berlin

Strijdhaftig D. (2008) On the coupling of architectures – Leveraging DEMO theory within the ARIS framework. MA thesis, Delft University of Technology, The Netherlands

Walls J. G., Widmeyer G. R., El Sawy O. A. (1992) Building an Information System Design Theory for Vigilant EIS. In: Information Systems Research 3(1), pp. 36–59

Winter R. (2008) Design Science Research in Europe. In: European Journal of Information Systems 17(5), pp. 470–475

Antonia Albani

Institute of Information Management
University of St. Gallen
Müller-Friedberg-Strasse 8
CH-9000 St.Gallen
Switzerland
antonia.albani@unisg.ch