Enterprise ontology based development of information systems

Antonia Albani*

Institute of Information Management,
University of St. Gallen,
Müller-Friedberg-Strasse 8, 9000 St. Gallen, Switzerland

and

Information Systems Design,
Delft University of Technology,
Mekelweg 4, 2628 CD Delft, The Netherlands
Fax: +41 71 224 3296
E-mail: antonia.albani@unisg.ch
*Corresponding author

Jan L.G. Dietz

Information Systems Design,
Delft University of Technology,
Mekelweg 4, 2628 CD Delft, The Netherlands
Fax: +31 15 278 66 32
E-mail: j.l.g.dietz@tudelft.nl

Abstract: For the development of enterprise information systems, the utilisation of a suitable methodology is essential, providing necessary methods and techniques for modelling the business domain and for designing the supporting information systems. Several methodologies exist and are widely applied in practice nowadays, but most of them lack a theoretical foundation. In this paper, we demonstrate an information system development methodology based on the notions of *enterprise ontology* and *business components*, and explain it within the conceptual framework called the *generic system development process*. The methodology allows for reduction of complexity of domain models and for identification of stable business components.

Keywords: enterprise information systems; enterprise ontology; business components; generic system development process.

Reference to this paper should be made as follows: Albani, A. and Dietz, J.L.G. (2011) 'Enterprise ontology based development of information systems', *Int. J. Internet and Enterprise Management*, Vol. 7, No. 1, pp.41–63.

Biographical notes: Antonia Albani is a Senior Researcher at the University of St. Gallen, Switzerland. Prior to that, she was an Assistant Professor at the Delft University of Technology, The Netherlands. She holds a Master's Degree and a PhD in Computer Science and her main research interests are in modelling component-based inter-enterprise information systems. She is a Co-founder of the research network CIAO! (Cooperation &

Interoperability – Architecture & Ontology, www.ciaonetwork.org) and member of the CIAO! executive board. She worked in IT consulting and was CEO and co-founder of an Internet start-up in the area of business process outsourcing.

Jan L.G. Dietz is a Professor at the Delft University of Technology, The Netherlands. His research interests are in modelling, (re)designing and (re)engineering organisations. He has published over 200 papers as well as several books. He is the spiritual father of DEMO (Design & Engineering Methodology for Organisations, www.demo.nl), Co-founder and Chairman of the DEMO Knowledge Centre, and leads the national research programme Extensible Architecture Framework (www.xaf.nl). His current interests are in the emerging field of Enterprise Engineering. To proliferate these ideas, he set up an international network, called the CIAO! Network (Cooperation & Interoperability – Architecture & Ontology, www.ciaonetwork.org).

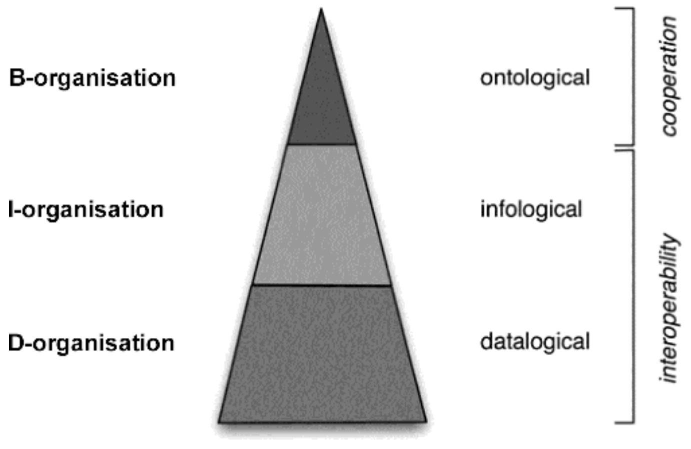
1 Introduction

Drastic changes in the competitive business landscape force companies to transform their organisational and managerial structures. While concentrating on core competencies, enterprises outsource secondary tasks to business partners, being confronted with an increasing need for cooperation and interoperability. Innovations in Information and Communication Technology (ICT) offer possibilities to increase the interoperability of information systems and to support and improve inter-enterprise cooperation. However, the design of intra- as well as inter-enterprise information systems and the deployment of modern ICT in implementing these systems do not always meet expectations. A major reason is the lack of an appropriate, deep understanding of enterprises and enterprise networks. The needed understanding cannot be drawn from the organisational and managerial sciences because of their predominant functional orientation. Functional knowledge is appropriate and sufficient for the use and control of enterprises, but in order to change them, knowledge about their construction and operation is needed. The first thing to be recognised is that enterprises are designed and engineered artefacts, and that suitable modelling methodologies need to be utilised in order to arrive at construction-oriented models.

Several enterprise modelling approaches exist and are widely applied in practice nowadays. Looking at business processes modelling techniques, being a relevant part of enterprise modelling, next to the traditional flow charts, there exist e.g., Petri Net (Jensen, 1997; van der Aalst and van Hee, 2001), Event Driven Process Chains (EPC) (Scheer, 1999) and Activity Diagrams (OMG, 2005). However, these techniques lack an appropriate understanding of the notion of business process for the purpose of (re)designing and (re)engineering business processing. In particular, they ignore that organisations are social systems, thereby reducing business processes to sequences of actions, conditions, and results. Consequently, these approaches do no justice to the deep structure of business processes (Dietz, 2006a), being that they are tree structures of transactions according to a universal transaction pattern.

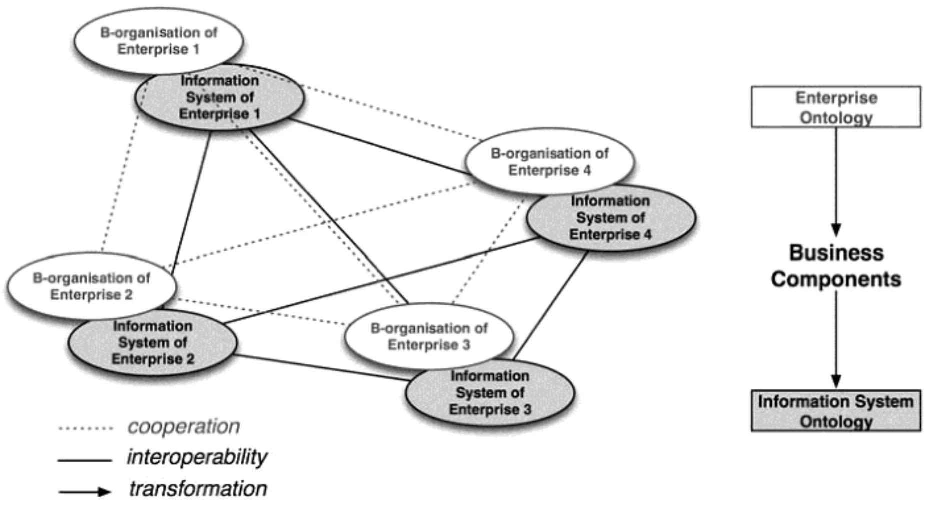
The enterprise ontology (Dietz, 2006b; Dietz and Habing, 2004) methodology is an approach that additionally distinguishes between essential (ontological), infological and datalogical production steps. The organisation of an enterprise is considered to consist of three aspect organisations: the B-organisation, the I-organisation, and the D-organisation (see Figure 1). In the D-organisation one is concerned with datalogical problems: the syntactic aspects of information (now mostly called data) and the operations on data and documents, like storing, copying and transporting. In the I-organisation one is concerned with infological problems: the semantic aspects of information and computational operations, like mathematical and logical reasoning operations. The notion of interoperability regards the dealing with datalogical and infological problems. Only in the B-organisation new, original facts are brought about, i.e., facts that change the business world. Examples of those facts are decisions and judgements. Next, for a long time, people (particularly IT-professionals) have thought that the function of communication in an organisation is to exchange information. Research in the Language-Action Perspective has taught however that communication is not only exchanging information, but that it also is a kind of action. It has taught that the essence of an organisation lies in the entering into and complying with commitments between social individuals (Denning and Medina-Mora, 1995; Goldkuhl and Lyytinen, 1982; van Reijswoud et al., 1999; Winograd and Flores, 1986). This holds for the B-organisation, the I-organisation and the D-organisation. In order to understand the operation of organisations, one has to understand how people coordinate their activities, namely by entering into and complying with commitments. The atomic coordination acts are acts like requesting and promising. These acts happen to occur in recurrent patterns, called transactions (Dietz, 2003b). Clearly distinguished from interoperability, cooperation is a notion that concerns the interaction between (people in) enterprises. We will restrict ourselves to cooperation at the level of the B-organisation, since the I-organisation and the D-organisation are supportive to it. Concluding, the issue of developing enterprise information system has to be studied on two levels, as illustrated in Figure 2.

Figure 1 Adapted version of the three aspect organisations



Source: Dietz (2006b, p.116)

Figure 2 Cooperation and interoperability


B-organisation of Enterprise 1
Information System of Enterprise 1
B-organisation of Enterprise 4
Information System of Enterprise 4
B-organisation of Enterprise 2
Information System of Enterprise 2
B-organisation of Enterprise 3
Information System of Enterprise 3
cooperation
interoperability
transformation
Enterprise Ontology
Business Components
Information System Ontology

The cooperation level is exactly the level on which the enterprise ontology provides the basis for investigating. Based on the enterprise ontology, one then can identify the business components that are going to support the business transactions, leading to the ontological model of the supporting information system. As defined by Barbier and Atkinson (2003), "a business component models and implements business logic, rules and constraints that are typical, recurrent and comprehensive notions characterising a domain or business area". A business component is self-contained, reusable and marketable and provides well-defined interaction points in order to facilitate the access and execution of the business logic provided. The derivation of a business component model is therefore the first step to developing information systems on a higher level of abstraction, which are understandable for business people defining the business requirements, using the solutions and deciding about future strategies. A business component model constitutes the basis for the implementation of information systems and their interoperability making the development process more flexible and adjustable to the business needs. The advocated way to proceed from the information system ontology to an implemented information system is discussed in Dietz (2005) and will not be elaborated in the current paper.

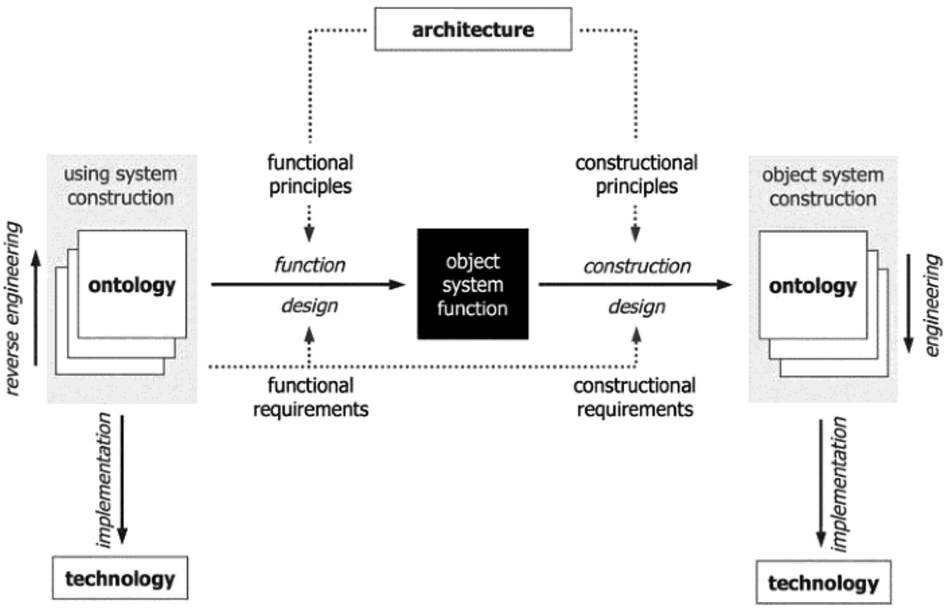
Instead, we will show how the notions of enterprise ontology and business components are incorporated into a methodology suitable for the development of intra and inter-organisational systems. The methodology will be presented the conceptual framework called generic system development process. This is a conceptual framework that elucidates what designing and engineering of enterprise information systems is all about. In order to describe the process steps we will use an example from the domain of strategic sourcing in enterprise networks throughout the paper. With this paper we contribute to bridging the gap between business and IT, since there is still a strong need for modelling, design and development methods mapping high-level business requirements to software technology, as e.g., discussed in Stojanovic and Dahanayake (2005).

The outline of the rest of the paper is as follows. In Section 2, the generic system development process is presented and discussed. It is a conceptual framework for discussing the design, engineering, and implementation of systems of any kind. Section 3 deals with the notion of enterprise ontology and the particular methodology Design & Engineering Methodology for Organisations (DEMO), which we apply to build ontological models of enterprises, including enterprise networks. As explained, such a model constitutes the starting point for the functional design of business components. This topic is elaborated in Section 4. A specific, formal approach to identifying business components is introduced, called the BCI-3D method (a three dimensional graphical method for Business Components Identification). Section 5 and 6 contain evaluations and conclusions of the research presented.

2 Designing and engineering information systems

In designing a system (of any kind) both the functional and the constructional perspective on systems are relevant (Dietz and Albani, 2005). Taking the functional perspective on a system means that one is interested in its (external) function and behaviour. The corresponding type of model, for expressing function and behaviour, is the black-box model. In contrast, taking the constructional perspective means being interested in the (internal) construction and operation of the system. The corresponding type of model is the white-box model. 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. Figure 3 exhibits the basic steps and context of the process of developing the object system, collectively called the generic system development process (Dietz, 2008).

Figure 3 Generic system development process



Source: Dietz (2008)

For the area of inter-enterprise cooperation, the using system would be an enterprise network, and the object system could be some inter-enterprise information system, supporting the business activities executed within and between enterprises being part of the enterprise network. The starting point is the need by the using system of a supporting system (the object system). By nature, this need stems from the construction of the using system, so the design of the information system starts from a white-box model of the using system, i.e., a model of its construction and operation (Dietz and Albani, 2005). Ideally, the basis for the design phase is the ontological model of the using system (Dietz, 2006b), which is the fully implementation independent constructional model, in our case of the enterprise network, since it only shows the essence of the using system. Building such a model will be elaborated in Section 3.

The design phase starts with designing the function of the object system, expressed in a black-box model of the 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. The other main input is the set of functional principles that apply to the design of the object system. 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)). Based on the insight that the design of a system is the design of components and their relationships (Churchman, 1971), the concept of business components seems very well suited for the functional design of an information system. A business component provides a set of services out of a given business domain through well-defined interfaces and hides its implementation (Fellner and Turowski, 2000).

The next basic design step is the design of the construction of the object system, expressed in a white-box model 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. Designing the construction of a business component would mean to model its internal view that, when implemented, would bring about the behaviour as specified in the function design. A thorough analysis of the resulting white-box model must guarantee that building the information system is feasible, given the available technology.

According to Alexander (1960), 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 and synthesis steps. In an analysis step, a better understanding is achieved of the requirements of the object system (including those enforced by the functional principles). This improved understanding results in extending or improving the black-box model of the object system. In a synthesis step, a better understanding is achieved of how the system could be built. This results in extending or improving the specifications regarding the construction and operation of the object system. The iterative nature of designing (which we left out from Figure 3 for the sake of simplicity) is well known: the final result of every design process is (or should be) a balanced compromise between reasonable functional requirements and feasible constructional specifications.

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. For example, the implementation model of an information system is the source code in some programming language. The 'highest' model is called the ontological model or ontology of the system. This model is fully independent of its implementation and only shows the essential features of the system.

To exemplify the usability of the generic system development process, focusing on the design steps as introduced above, the domain of Strategic Supply Network Development (SSND) (Albani et al., 2003, 2004, 2007) is used as an example in the following sections.

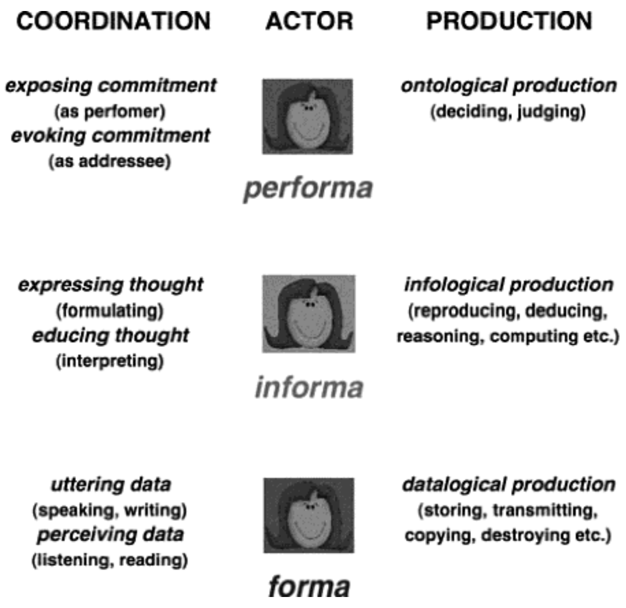
3 Building the ontological model of an enterprise

As we mentioned already in Section 1, a crucial factor in modelling a system is the appropriateness of the applied meta model for the system category (Bunge, 1979) to which the system belongs. It should be recognised particularly that organisations belong to the category of social systems. Moreover, an ontological model should fulfil the quality criteria as introduced in Albani and Dietz (2006) and Dietz (2006b, p.8) and listed below:

- coherent (i.e., the parts constitute an integral whole)
- consistent (i.e., there are no contradictions or irregularities)
- comprehensive (i.e., all relevant issues are dealt with)
- concise (i.e. the model does not contain superfluous matters)
- essential (i.e., it shows only the essence, according to the system category, and independent of the realisation and the implementation of the system).

A business domain model that satisfies all of these requirements is called an enterprise ontology. As mentioned in Section 1, several domain modelling techniques exist – e.g., Petri Nets (Jensen, 1997; van der Aalst and van Hee, 2001), EPC (Scheer, 1999), Activity Diagrams (OMG, 2005) or the traditional flow charts – propagating the business specific aspect of enterprise modelling, but most of them do not satisfy all of the quality criteria mentioned, particularly the last one. The enterprise ontology (Dietz, 2006b) methodology is a promising approach that offers a solution for the mismatch between social perspectives and technical perspectives by explicitly focusing on business specific communication patterns, where social individuals, i.e., human beings, achieve changes in the (object) world by means of communicative acts (Denning and Medina-Mora, 1995; Goldkuhl and Lyytinen, 1982; van Reijswoud et al., 1999; Winograd and Flores, 1986). In addition, and equally important, it distinguishes between essential (ontological), informational (infological) and documental (datalogical) actions, as shown in Figure 4.

Figure 4 The three distinct human capabilities



Source: Dietz (2006b, p.105)

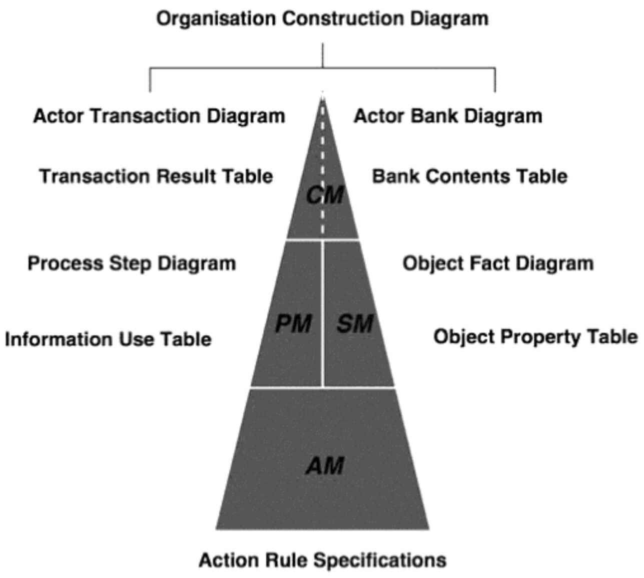
As is explained in Dietz (2003a, 2003b, 2006b), an enterprise – or a network of enterprises – consists of social individuals who perform two kinds of acts: production acts (resulting in production facts) and coordination acts (resulting in coordination facts). The transaction axiom aggregates these acts/facts into the universal pattern of the (business) transaction. Consequently, two worlds are distinguished in which the acts of the social individuals have effect: the production world (P-world) and the coordination world (C-world).

Regarding coordination acts, the forma ability concerns the form aspects, the informa ability concerns the content aspects, and the performa ability concerns the being engaged in commitments. The first abstraction DEMO makes in arriving at the ontological model of an enterprise consists of taking only into account the performa ability in coordination, thus leaving out how C-acts are actually performed on the informa and the forma level (see Figure 4), while at the same time grouping them into transactions. It results in an enormous reduction of complexity, on the average estimated to be over 70% in terms of the amount of documentation (Dietz, 2006a).

Regarding production acts, the forma ability concerns the datalogical production, the informa ability concerns the infological production, and the performa ability concerns the ontological production. The performa ability is the essential human ability for doing business, of any kind. The second abstraction DEMO makes in arriving at the ontological model of an enterprise consists of taking only into account the performa ability in production (thus the B-organisation), leaving out the transactions on the informa and the forma level (thus in the I-organisation, and the D-organisation). This results in a second enormous reduction of complexity, on the average estimated also to be over 70% in terms of the amount of documentation (Dietz, 2006a).

The complete ontological model of an organisation consists of four aspect models (see Figure 5). The Construction Model (CM) specifies the composition, the environment and the structure of the organisation. It contains identified transaction types, which are executed by associated actor roles, as well as the information links from actor roles to production and coordination banks, which are the conceptual stores of production and coordination facts respectively. The Process Model (PM) details each single transaction type of the CM by means of the universal transaction pattern. Next, it contains the causal and conditional relationships between transactions. Business processes thus are tree structures of transactions. The Action Model (AM) specifies the business rules that serve as guidelines for the actors in dealing with business events, i.e., occurrences of coordination facts. The State Model (SM) specifies the object classes, fact types and ontological coexistence rules in the production world. Diagrams and tables are used to express the information relevant to each model. The Actor Transaction Diagram and the Actor Bank Diagram together constitute the Organisation Construction Diagram.

Figure 5 The four aspect models



Source: Dietz (2006b, p.141)

Based on this method, the ontology for the SSND case has been constructed. Space limitations prohibit us to provide a more extensive account of how the models in the figures below are developed. Also, we will not present and discuss the Action Model. The basic idea of the SSND example is the identification of suppliers, located not only in tier-1 but also in the subsequent tiers, which are able to deliver specific components of a product to the Original Equipment Manufacturer (OEM) for constructing a specific product. This is established in sending out an offering request for a specific product to the tier-1 suppliers that execute a bill-of-material explosion in order to decide which products need to be requested from their own suppliers. This repeats until the request has reached the last tier. The information is then aggregated and transferred to the initial tier.

Figure 6 exhibits the Organisation Construction Diagram of the SSND case and Table 1 exhibits the corresponding Transaction Result Table. Collectively, they represent the Construction Model.

Figure 6 Organisation construction diagram of the SSND case

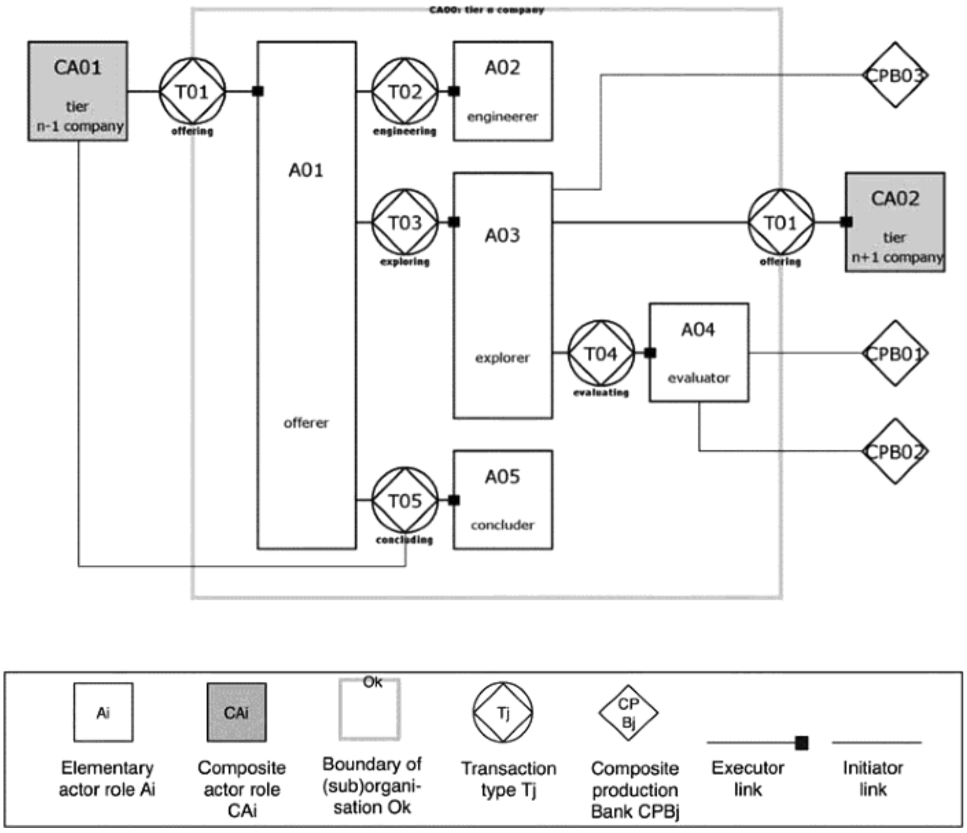


Table 1 Transaction Result Table of the SSND case

Transaction type	*Resulting P-event type*
T01 offering	PE01 supply contract C is offered
T02 engineering	PE02 the BoM of assembly A is determined
T03 exploring	PE03 supply contract C is a potential contract
T04 evaluating	PE04 supply contract C is evaluated
T05 concluding	PE05 supply contract C is concluded

The top or starting transaction type is the offering transaction (T01). Instances of the offering transaction are initiated by the environmental actor role CA01, which is a company in tier $n - 1$ and executed by the offerer (A01) in the tier-n company. CA01 asks the direct supplier (CA00) for an offer regarding the supply of a particular product P.

In order to make such an offer, the offerer (A01) first initiates an engineering transaction (T02), in order to get the bill of material of the requested product P. This is a list of (first-level) components of P, produced by the engineerer (A02). Next, the offerer (A01) asks the explorer (A03) for every such product component to get offers from companies that are able to supply the component. So, a number of exploring transactions (T03) may be carried out within one offering transaction (T01), namely as many as there are components of the product P which are not produced by the tier n company itself.

In order to execute each of these transactions, the explorer (A03) has to ask companies for an offer regarding the supply of a component of the product P. Since this is identical to starting another offering transaction (T01), we model this as initiating an offering transaction (T01). Now however, the executor of the offering transaction (T01) is a company in tier $n + 1$. Consequently, the model that is shown in Figure 6 must be understood as to be applicable recursively for every tier until the products to be supplied are elementary, i.e., non-decomposable. Note that, because of the being recursive, an offer (the result of a T01) comprises the complete bill of material of the concerned component of a product P.

Every offer from the companies in tier $n + 1$ is evaluated in a T04 transaction by the evaluator (A04) in tier n. So, there is an evaluating transaction (T04) for every 'output' of an offering transaction (T01), whereby each company can have its own evaluation rules. The result of an evaluating transaction (T04) is a graded offer for some component of product P. So, what the explorer (A03) delivers back to the offerer (A01) is a set of graded offers for every component of product P. Next, the offerer (A01) asks the concluder (A05), for every component of product P, to select the best offer. The result is a set of concluded offers, one for every component of product P. This set is delivered to the offerer (A01). Lastly, the offerer (A01) delivers a contract offer to the tier $n - 1$ company (CA01) for supplying product P, together with the set of concluded offers for delivering the components of product P. Because of the recursive character of the whole model, this offer includes the complete bill of material of product P, regardless its depth.

The Organisation Construction Diagram in Figure 6 contains three external production banks. Bank CPB01 contains the data about a company that are relevant for the evaluation of offers. Bank CPB02 contains the different evaluation methods that can be applied. In every instance of the evaluating transaction (T04), one of these methods is applied. CPB03 contains identifiers of all companies that may be addressed for an offer. The dashed lines represent information, i.e., access, links to these banks. Lastly, in the transaction result table (see Table 1), the supply of a product by a (supplying) company to a (customer) company is conceived as a contract.

Figure 7 exhibits the Process Step Diagram of the SSDN case. Due to visualisation reasons only shortcuts are listed for the single process steps. For the real names see Table 2. The Process Step Diagram is based on the universal transaction pattern, although only the basic pattern is exhibited (request, promise, execute, state, accept) (Dietz, 2003b). It shows how the distinct transaction types are related. From the state promised offering (T01/pm) a number of exploring transactions (T03) (possibly none) and a number of concluding transactions (T05) (possibly none) are initiated, namely for every first-level component of a product. This is expressed by the cardinality range 0 ... *k*. Likewise, from the state promise exploration (T03/pm), a number of offering transactions (T01) and a number of evaluating transactions (T04) are initiated, namely for every offer or contract regarding a first-level component of a product. The dashed arrows, from an accept state (e.g., T02/ac) to some other transaction state, represent waiting conditions. So, for example, the performance of an exploration request (T03/rq) has to wait for the being performed of the corresponding engineering accept (T02/ac).

Figure 7 Process Step Diagram of the SSND case

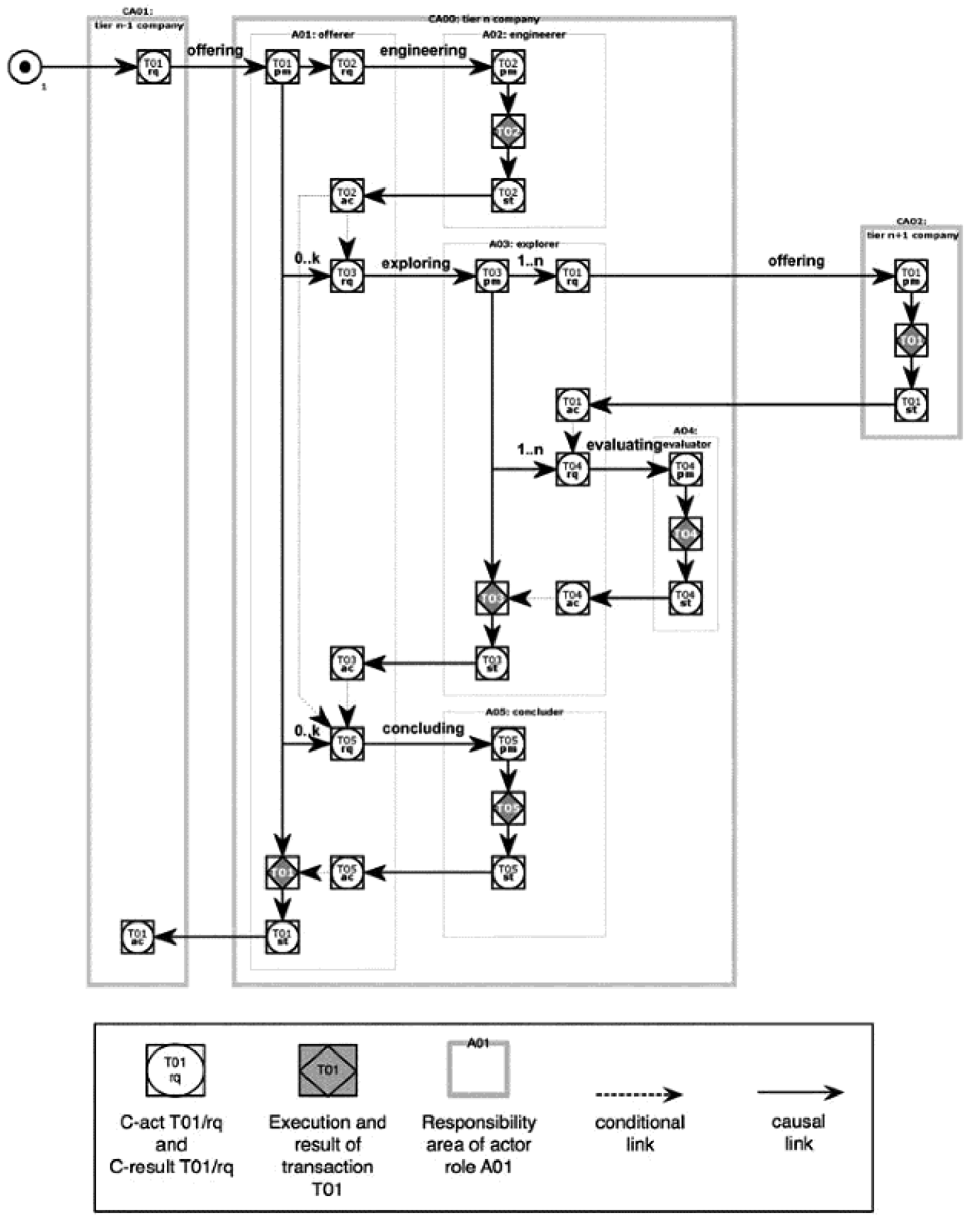


Table 2 Assignment of process step names to shortcuts

Process steps names	*Shortcuts*
Request offering	T01/rq
Promise offering	T01/pm
Produce offering	T01/ex
State offering	T01/st
Accept offering	T01/ac
Request engineering	T02/rq
Promise engineering	T02/pm
Produce BoM explosion	T02/ex
State engineering	T02/st
Accept engineering	T02/ac
Request exploration	T03/rq
Promise exploration	T03/pm
Produce contract	T03/ex
State exploration	T03/st
Accept exploration	T03/ac
Request evaluation	T04/rq
Promise evaluation	T04/pm
Produce evaluation	T04/ex
State evaluation	T04/st
Accept evaluation	T04/ac
Request conclusion	T05/rq
Promise conclusion	T05/pm
Produce concluded contract	T05/ex
State conclusion	T05/st
Accept conclusion	T05/ac

Figure 8 exhibits the Object Fact Diagram and Table 3 the Object Property Table. Together they constitute the State Model of the example case.

The Object Fact Diagram is the ontological variant of the Object Role Model (ORM) diagram (Halpin, 2001). Diamonds represent unary fact types that are the result of transactions, also called production fact types. They correspond with the transaction results in Table 1. A roundangle around a fact type or a role defines a concept in an extensional way, i.e., by specifying the object class that is its extension. For example, the roundangle around the production fact type “C is evaluated” defines the concept of evaluated contract. Lastly, the roundangle around the role ‘A’ of the fact type “P is a part of A” defines all assemblies, i.e., all products that do have parts. Properties are binary fact types that happen to be pure mathematical functions, of which the range is a set of, usually ordered, values, called a scale. Instead of including them in an Object Fact Diagram they can be more conveniently represented in an Object Property Table (Table 3). The information items as defined in the State Model, including the derived fact types, constitute all information that is needed to let a supply network for a particular product be operational.

Figure 8 Object fact diagram of the SSND case

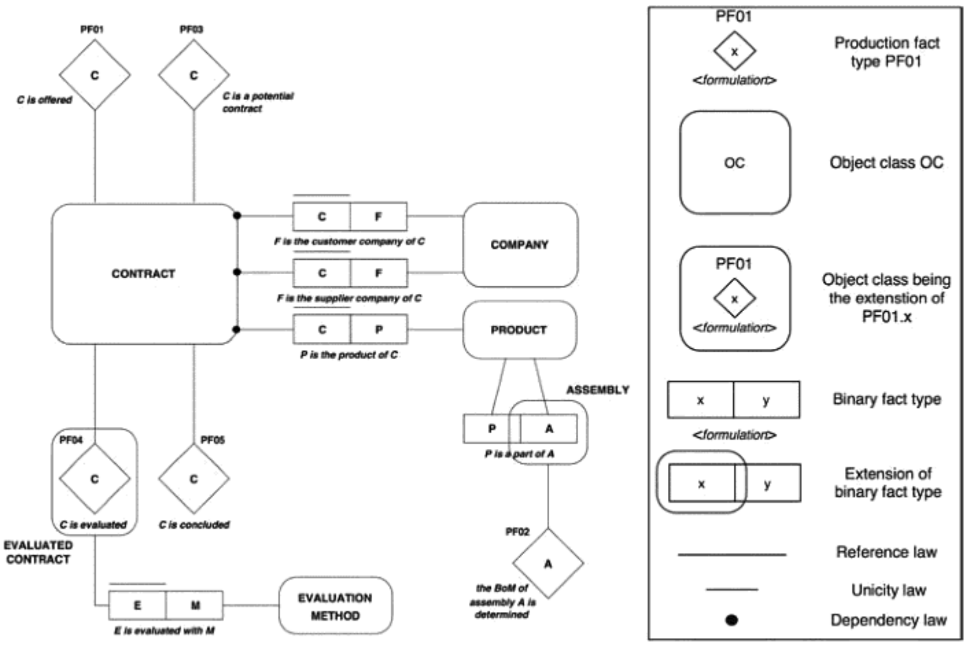


Table 3 Object property table of the SSND case

Property type	*Object class*	*Scale*
< company information >	COMPANY	< aggregated data >
< contract terms >	CONTRACT	< aggregated data >
evaluation mark	CONTRACT	NUMBER

4 Functional design of business components

As the ontological models of the construction and operation of networked enterprises provide an abstract view, while reaching an enormous reduction in complexity, the same should hold for the models describing the supporting information systems. The goal of such models is to provide a transparent view of the essential elements of the increasingly complex information systems to business people in order to better understand the functionality, define requirements and decide about future strategies. The concept of business component is a very promising one for reaching that goal. In general, the concept of component and service oriented development has been proposed for building complex but adaptive and agile enterprise information systems that provide effective inter-enterprise and intra-enterprise integration (Stojanovic and Dahanayake, 2005, p.vi). The identification of reusable and marketable business components and their services is therefore a primary research problem that needs to be addressed. Today, there is still little research initiative, see e.g., Jang et al. (2003), Levi and Arsanjani (2002), Réquilé-Romanczuk et al. (2005). Vitharana et al. (2003) recognises that more formal methodologies are needed to make the component based software development paradigm into an effective development tool. With our formal method used for the identification of business components, introduced later in this section, we directly contribute to this research area.

Since the identification of business components is strongly dependent on the underlying business models, it is crucial to use appropriate and high-quality business models. Here is where the enterprise ontology, as presented in Section 3, contributes strongly. Additionally, in order to reach an adequate clustering of business functionality within business components the underlying functional design principles (shown in Figure 3), which constrain the design of the function of the supporting information system, need to reflect (strategic) business goals. In the current literature, however, most design principles arise from technical aspects. Inter-component coupling, intra-component cohesion, number of components, size of components and component complexity are technical features, which are often mentioned in the context of component identification (Jain et al., 2001; Kim and Chang, 2004; Vitharana et al., 2003) focusing on improving the performance of information systems. Since these are technical features, which do not immediately arise from business goals, it is not guaranteed that the mapping of business models to business components based on these features is adequate for the required business goals. In Jain et al. (2001), Vitharana et al. (2003) truly managerial goals are additionally proposed, such as cost effectiveness, ease of assembly, reusability or maintainability. Those goals do satisfy our understanding of design principles because they do constrain the function or the construction of the supporting information system. Looking at those requirements, some are dominantly functional and others are dominantly constructional. Cost effectiveness e.g., clearly regards constructional design. We therefore put this design principle in the construction phase of the information system and do not use it for our formal business components identification approach. On the other hand, maintainability is a design issue that is highly facilitated by component-based design. Software reusability improves maintainability, quality, portability and productivity (Apte et al., 1990). Reusability instead is achieved directly by the functional design principle of clustering coherent domain functionality within a business component, which is the first design principle in our business identification approach. The same holds for the ease of assembly goal. Ease of assembly can e.g., be achieved by reducing the interfaces between components. It means that this management goal can directly be mapped to the functional design principle called loosely coupling of components, which defines to which extent a component is coupled with other components. We use this as a second design principle in the business components identification method and later on also as a constructional design principle. Along with the managerial goals defined in the literature, we gained knowledge about additional managerial goals from real-world business cases (see e.g., Skroch and Turowski (2008)).The three main managerial goals we identified are shortly summarised next. The first goal was the simplification of information systems through segregation and consolidation due to a complex functional mix-up inside one company and between subsidiary companies. The second goal was the coupling of services in order to reduce communication and therefore performance problems. And the third goal was the central management of related information, in order to reduce the high costs of solving update inconsistencies of redundant data. While the first goal can be mapped to the first functional principle, clustering coherent domain functionality within a business component, and the second goal can be mapped to the second functional principle, loosely coupling of components, the third goal needs to be mapped to a new functional design principle, namely the management of related information objects within one business component. Below the resulting functional design principles are summarised, formulated in the common 'must-form':

- Within a business component, coherent domain functionality must be clustered
- Business components must be coupled loosely
- Related information objects must be managed within one business component.

These design principles build the basis for our formal business components identification method, called three dimensional method for Business Components Identification (BCI-3D). BCI-3D aims at grouping business tasks and their corresponding information objects (of the enterprise models introduced in the previous section) into business components (Albani et al., 2005, 2008). In order to provide optimal grouping satisfying the design principles, an optimisation problem needs to be solved. As described by Simon (1996, p.116) optimisation methods, most highly developed in statistical decision theory and management science, are acquiring growing importance also in engineering design theory. He argues that in the real world we only rarely have a method for finding the optimum. This holds also for our component identification problem. Since no computational power would get an optimal result in polynomial time (Wang et al., 2005, p.234) we need to look for alternatives that satisfy our functional design principles and that are found after only moderate search. As defined by Simon (1996, p.127), the process of searching for alternatives can be viewed as a process for seeking a problem solution. According to the method definition given by March and Smith (1995), we use a genetic algorithm as a method for finding a solution to the business components identification problem. As input we take the ontological models describing the business domain and map it to a weighted graph. The nodes in the graph are of two types. The ones represent process steps gained from the DEMO process step diagrams and the others represent objects and fact types gained from the DEMO object fact diagrams. The relationships between all the nodes represent the relationships modelled in the mentioned models and the weights define the different types of relationships, e.g., relationship between two process steps or relationship between two objects. The mapping of the DEMO models to a weighted graph is straightforward and is needed in order to be able to implement graph partitioning algorithms for grouping business tasks and their corresponding information objects into business components. While executing the genetic algorithm, a business component model is generated, representing the solution space. For details about the optimisation algorithm we refer to Albani et al. (2008), Albani and Dietz (2006, 2008).

Applying the BCI-3D method on the DEMO models introduced in section 3 results in the following clustering (see Figure 9). Two business components can be identified immediately. While looking at the process steps and information objects clustered within the components one can identify the business functionality of the two business components, one containing the business tasks related to product management and one containing the business tasks related to contract management. From Figure 9 the services provided and required by each component can be derived. We distinguish between two types of services: inter-component services and *intracomponent* services.

Inter-component services are services, which are required by another component in order to provide a specific functionality. The inter-component services are apparent in Figure 9 as the lines connecting two process steps, each located in a different component. E.g., the line connecting the T01/pm and T05/rq defines an inter-component service. Which component requires or provides that service becomes clear when looking at the process flow. Since the T01/pm calls the process step T05/rq the component holding the T05/rq provides that service. In order to visualise all services provided by a component, we added a provided service element (composed of a circle attached to a line, following the UML 2.0 notation) to each corresponding process step (see Figure 9). Additionally, to each process step, which requires a service from another component, a required service element is added (composed of a semicircle attached to a line). For the example just mentioned, the service provided by the Contract Manager component relates to the conclusion of the contract, and is therefore called ProduceConcludedContract. The components with their required and provided services are shown in Figure 10.

Figure 9 Process steps clustered within the two identified business components

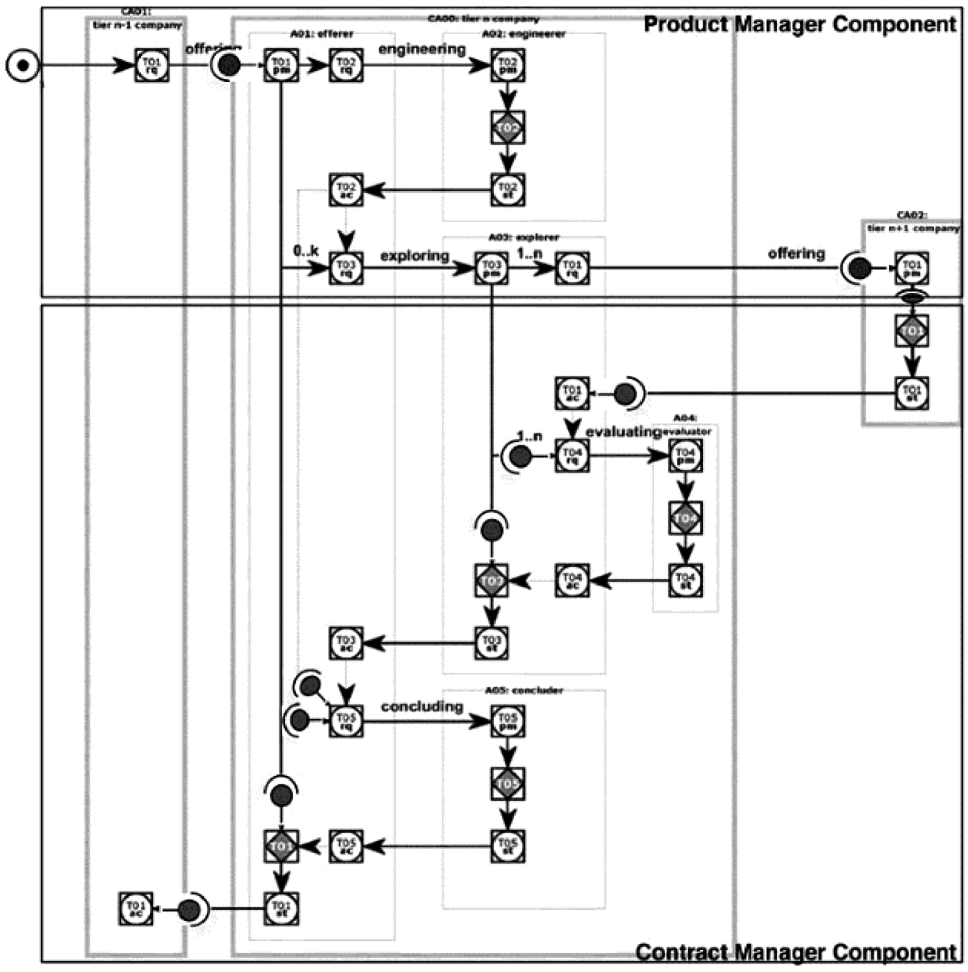
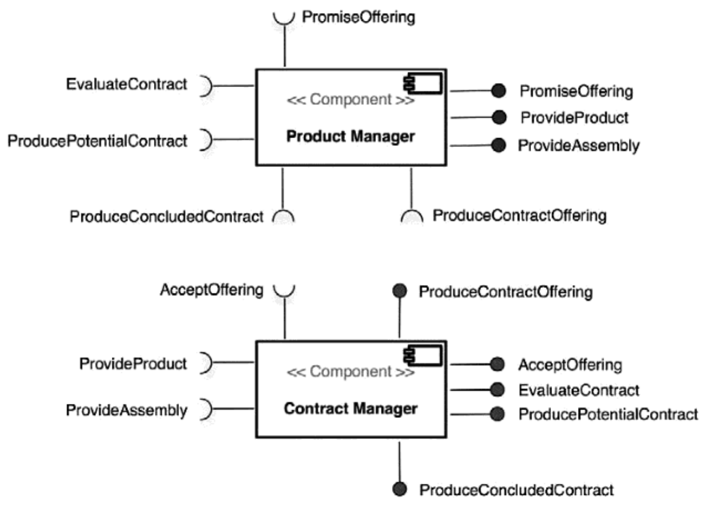


Figure 10 Required and provided services of the business components identified



Additionally to inter-component services, we identify intra-component services. Following the reuse principle of component-based development of information systems, different enterprises within the network can use the same business functionality provided by one of the identified business components. This means that the same component is deployed on different network nodes. If one component needs to call a process step located in the own component, but deployed on a different enterprise, a service needs to be provided in order to be able to request that functionality. As shown in Figure 9, if e.g., tier-(*n*–1) company requests a contract offer from tier-n company, the process steps T01/rq and T01/pm are executed. While T01/rq is executed by company in tier-(*n*–1) T01/pm is executed by company in tier-n. Since both process steps are logically clustered within the Product Manager component, a service needs to be provided in order to request the contract offering. In this example the service provided by the Product Manager is called Promise- Offering and is shown in Figure 10. The intra-component services can be identified while analysing Figure 9. All intra-component services are additionally added to Figure 9 in order to visualise the required and provided services of each component.

With the BCI-3D method business components and their corresponding services can be identified based on functional design principles fulfilling defined business goals. The partitioning of the weighted graph allows the clustering of process steps and information objects to business components and provides the basis for the definition of required and provided services. Not only the services between components within an enterprise, but especially also all services, which need to be provided or which are required from components deployed on other enterprises are identified. The business component model with their services provides the functional definition of the supporting information system on a high level of abstraction, without defining their internal view. The internal view and the coordination between components are defined in the white-box ontological model of the object system. The details explaining that step is not within the scope of this paper.

5 Evaluation

The ontology-based design of inter-enterprise systems has been illustrated by means of a non-trivial example case, namely the SSND case. The domain of strategic supply network development refocuses the object of reference in the field of strategic sourcing by analysing and selecting supplier networks instead of single suppliers. The recursive structure of such networks can easily and elegantly be dealt with by DEMO and the required and provided services can be identified and visualised by applying the BCI-3D method. Even if for this small example only two business components were found, the derivation of the services would not have been possible without a detailed analysis and modelling of the business domain. This information is essential and provides the basis for the implementation of the business components. Place limitations prevented us to derive components from a larger example domain. A prototype implementation of the supporting information system for the SSDN network, consisting of the identified business components, has been built and deployed on different network nodes.

The BCI-3D method has been introduced as a formal method for designing the function of the supporting information systems by means of business components. The method clusters process steps and information objects into business components. With BCI-3D we provide an optimisation method that identifies satisfactory solutions, obeying defined functional design principles. In order to automate the component identification process, a prototype of the BCI-3D method has been implemented. In the past years, the BCI-3D method has been applied in several industrial and academic projects (e.g., (Peters, 2009; Eberhardt et al., 2006; Selk et al., 2005)), emphasising the advantages of using a formal approach for the derivation of information systems models from domain models. For these projects, different domain modelling methodologies/notations had been used, as e.g., ARIS (Scheer, 1999), UML (OMG, 2005) or DEMO (Dietz, 2006b), resulting in business component models of different quality, (e.g., providing services for implementation specific tasks). Since the quality of the resulting ontological models of the information system(s) is strongly dependent on the underlying business domain models, the ‘best’ business domain modelling methodology need to be applied. In order to take maximal advantage we combined DEMO and BCI-3D in a new methodology for developing information systems.

Consolidated findings about the new presented methodology could be gained by a case study conducted at an international logistic service provider (Peters, 2009). The core activities of the company are the transportation and storage of liquids for chemical companies, the transportation of gas, and the cleaning of tank-containers and road-barrels. By applying DEMO and BCI-3D for setting the base for a service oriented architecture, the company could gain important insights concerning the business and the information systems.

6 Conclusions

Most of the current approaches to modelling enterprises do not focus on the essential features of an enterprise, resulting in unnecessarily complex, unstable, and unwieldy models, which are unsuitable for business process re-design and reengineering. Also the, still running, European SUPER project (SUPER, 2009) has failed to recognise the inherent, social action based, deep structure of business processes, instead frenetically trying to accommodate unscientific and unsuited approaches, like ARIS and BPEL. The same holds for the models of the supporting information systems. In order to provide valuable information to business people who decide about requirements, use the solutions and decide about future strategies, both the business domain models and the supporting information system models need to be appropriate, i.e., be in accordance with the system category, and ontological, i.e., completely implementation independent. As has been shown, this can only be achieved by applying methodologies that are founded in appropriate theories. As also shown, this goal can be reached in a very effective and efficient way by applying the DEMO methodology and the BCI-3D method, covering only the essential features of both enterprises (or enterprise networks) and their supporting information systems. The combination of both approaches to a new methodology for the development of information systems is the main contribution of this paper. It has been explained in the framework of the generic system development process.

The starting point for designing supporting information systems is the ontological model of an enterprise (or a network of enterprises). Besides being very well suited for modelling the essential structure of business processes within an enterprise, the DEMO methodology contributes especially to the design of inter-enterprise cooperation, as has been shown in the SSND example, which is a non-trivial one due to its recursive structure. Applying DEMO for the modelling of a business domain results in a drastic reduction in complexity, while at the same time ending up with an ontological model that is coherent (i.e., the parts constitute an integral whole), consistent (i.e., there are no contradictions or irregularities), comprehensive (i.e., all relevant issues are dealt with), concise (i.e., the model does not contain superfluous matters), and essential (i.e., it shows only the essence, according to the system category, independent of the realisation and the implementation of the enterprise). Such ontological models have proven to be an ideal starting point for discussing all kinds of problems in enterprises, including business process management, organisational changes, information systems management, etc., and suggesting effective solutions. The DEMO methodology has been applied in hundreds of practical projects over the past 15 years.

By applying DEMO to derive the business domain models and BCI-3D for identifying components, the resulting information system models contain only their essential features, which appear to be very well understandable by business people. The accurate and deep understanding of the construction and operation of the supporting information systems provides a valuable basis for a well-controlled and optimal evolution of these systems. Based on the underlying essential enterprise models and on the component-based approach, a drastic reduction in complexity can be achieved also in the modelling of the supporting information systems. Additionally, the identified components and their services have a reference character. That means that they are stable since they are based on ontological models, which are completely implementation independent models. A business domain is not going to change often, but the implementation of that business domain may change easily.

Currently, we are extending the BCI-3D tool with functionality to automatically generate the ontological model of the supporting information systems, i.e., to automatically generate the internal views and the composition of the identified components. One of the planned future projects is to integrate the tool for modelling the business domain following the DEMO methodology and the tool implementing the BCI-3D method in order to realise automatic identification of business components after the business models are built, without the need of manually transforming the ontological model into the representation needed for applying the BCI-3D method.

References

Albani, A. and Dietz, J. (2008) 'Ch. benefits of enterprise ontology for the development of ICT-based value networks', *Software and Data Technologies, Second International Conference, ICSOFT/ENASE 2007*, Vol. 22, Springer-Verlag, Barcelona, Spain, 22–25 July, 2007, pp.3–22, Revised Selected Papers.

Albani, A. and Dietz, J.L. (2006) 'The benefit of enterprise ontology in identifying business components', *The Past and Future of Information Systems: 1976–2006 and Beyond, IFIP 19th World Computer Congress, TC-8, Information System Stream. Vol. 214/2006 of IFIP International Federation for Information Processing*, August, Springer Boston, Santiago de Chile, Chile, pp.243–254.

Albani, A., Dietz, J.L. and Zaha, J.M. (2005) 'Identifying business components on the basis of an enterprise ontology', in Konstantas, D., Bourrieres, J-P., Leonard, M. and Boudjlida, N. (Eds.): *Interoperability of Enterprise Software and Applications*, Springer-Verlag, Geneva, Switzerland, pp.335–347.

Albani, A., Keiblinger, A., Turowski, K. and Winnewisser, C. (2003) 'Dynamic modelling of strategic supply chains', in Bauknecht, K., Tjoa, A.M. and Quirchmayr, G. (Eds.): *4th International Conference on E-Commerce and Web Technologies (EC-Web), Vol. 2738 of LNCS*, Prague, Czech Republic, September, pp.403–413.

Albani, A., Müssigmann, N. and Zaha, J.M. (2007) *Reference Modeling for Business Systems Analysis*, Idea Group Inc., Ch. A Reference Model for Strategic Supply Network Development, pp.217–240.

Albani, A., Overhage, S. and Birkmeier, D. (2008) 'Towards a systematic method for identifying business components', in Chaudron, M., Szyperski, C. and Reussner, R. (Eds.): *Components-Based Software Engineering, 11th International Symposium, CBSE 2008, Vol. 5282 of LNCS*, Springer-Verlag, pp. 262–277.

Albani, A., Winnewisser, C. and Turowski, K. (2004) 'Dynamic modelling of demand driven value networks', in Meersmann, R. and Tari, Z. (Eds.): *On The Move to Meaningful Internet Systems and Ubiquitous Computing: CoopIS, DOA and ODBASE, Vol. 3290 of LNCS*, Springer-Verlag, Larnaca, Cyprus, pp.408–421.

Alexander, C. (1960) *Notes on the Synthesis of Form*, Ablex Publishing Company, New Jersey, USA.

Apte, U., Sankar, C.S., Thakur, M. and Turner, J.E. (1990) 'Reusability-based strategy for development of information systems: Implementation experience of a bank', *MIS Quarterly*, Vol. 14, No. 4, pp.421–433.

Barbier, F. and Atkinson, C. (2003) *Business Component-Based Software Engineering*, Kluwer Academic Publishers Group, Ch. Business Components, pp.1–26.

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

Churchman, C.W. (1971) *The Design of Inquiring Systems: Basic Concepts of Systems and Organization*, Basic Books, New York.

Denning, P. and Medina-Mora, R. (1995) 'Completing the loops', *ORSA/TIMS Interfaces*, Vol. 25, No. 3, pp.42–57.

Dietz, J. (2008) Architecture – building strategy into design. Academic Services imprint of Sdu Uitgevers bv, The Hague, The Netherlands.

Dietz, J.L. (2003a) 'The atoms, molecules and fibers of organizations', *Data and Knowledge Engineering*, Vol. 47, pp.301–325.

Dietz, J.L. (2003b) 'Generic recurrent patterns in business processes', in van der Aalst, W.A.H. and Weske, M. (Eds.): *Business Process Management, Vol. 2678 of LNCS*, Springer Heidelberg, pp.200–215.

Dietz, J.L. (2005) 'System ontology and its role in system development', in Castro, J. and Teniente, E. (Eds.): *Advanced Information Systems Engineering Workshops (CAiSE 2005)*, Vol. 2 of FEUP Edicoes, Porto, Portugal, pp.273–284.

Dietz, J.L. (2006a) 'The deep structure of business processes', *Communications of the ACM*, May, Vol. 49, No. 5, pp.58–64.

Dietz, J.L. (2006b) *Enterprise Ontology – Theory and Methodology*, Springer-Verlag, Berlin Heidelberg.

Dietz, J.L. and Albani, A. (2005) 'Basic notions regarding business processes and supporting information systems', *Requirements Engineering Journal*, Vol. 10, No. 3, pp.175–183.

Dietz, J.L. and Habing, N. (2004) 'A meta ontology for organizations', in Meersmann, R. and Tari, Z. (Ed.): *On the Move to Meaningful Internet Systems 2004: OTM 2004 Workshops*, Vol. 3292 of LNCS, October, Springer, pp.533–543.

Eberhardt, A., Gausmann, O. and Albani, A. (2006) 'Case study automating direct banking customer service processes with service oriented architecture', in Meersman, R., Tari, Z. and Herrero, P. (Eds.): *On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops, Vol. 4217 of LNCS*, Springer, pp.763–779.

Fellner, K. and Turowski, K. (2000) 'Classification framework for business components', in Sprague, R.H. (Ed.): *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, Vol. 8, IEEE, Maui, Hawaii, p.8047.

Goldkuhl, G. and Lyytinen, K. (1982) 'Language action view of information systems', in Ginzberg, M. and Ross, C. (Eds.): *Proceedings of the 3rd International Conference on Information Systems (ICIS'82)*, TIMS/SIMS/ACM, Ann Arbor, MI, USA, pp.13–29.

Halpin, T. (2001) *Information Modeling and Relational Databases*, Morgan Kaufmann, San Francisco.

Hoogervorst, J. (2004) 'Enterprise architecture: enabling integration, agility, and change', *Journal of Cooperative Information Systems*, Vol. 13, No. 3, pp.213–233.

Jain, H., Chalimeda, N., Ivaturi, N. and Reddy, B. (2001) 'Business component identification – a formal approach', *Proceedings of the Fifth International Enterprise Distributed Object Computing Conference (EDOC'01)*, IEEE Computer Society, Washington DC, USA, p.183.

Jang, Y-J., Kim, E-Y. and Lee, K-W. (2003) 'Object-oriented component identification method using the affinity analysis technique', in Konstantas, D., Leonard, M., Pigneur, Y. and Patel, S. (Eds.): *Proceedings of the 9th International Conference on Object-Oriented Information Systems (OOIS), Vol. 2817 of LNCS*, Springer-Verlag, Geneva, Switzerland, pp.317–321.

Jensen, K. (1997) 'Coloured Petri nets', *Basic Concepts, Analysis Methods and Practical Use*, Basic Concepts of Monographs in Theoretical Computer Science, Vol. 1, Springer.

Kim, S.D. and Chang, S.H. (2004) 'A systematic method to identify software components', *Proceedings of the 11th Asia-Pacific Software Engineering Conference (APSEC)*, Washington DC, USA, pp.538–545.

Levi, K. and Arsanjani, A. (2002) 'A goal-driven approach to enterprise component identification and specification', *Communications of the ACM*, Vol. 45, No. 10, October, pp.45–52.

March, S.T. and Smith, G.F. (1995) 'Design and natural science research on information technology', *Decision Support Systems*, Vol. 15, pp.251–266.

OMG (2005) *OMG Unified Modelling Language*, Version 2.2, February, URL http://www.omg.org/technology/documents/formal/uml.htm

Peters, H. (2009) *An Architecture Based on Services for Den Hartogh – Created with Demo and BCI-3D*, Master's Thesis, Delft University of Technology, Delft, The Netherlands.

Réquilé-Romanczuk, A., Cechich, A., Dourgnon-Hanoune, A. and Mielnik, J-C. (2005) 'Towards a knowledge-based framework for cots component identification', *Proceedings of the Second International Workshop on Models and Processes for the Evaluation of off-the-shelf Components (MPEC'05)*, ACM Press, New York, NY, USA, pp.1–4.

Scheer, A-W. (1999) *ARIS – Business Process Modeling*, 2nd ed., Springer Verlag, New York, USA.

Selk, B., Bazijanec, B. and Albani, A. (2005) 'Experience report: Appropriateness of the bci-method for identifying business components in large-scale information systems', in Turowski, K. and Zaha, J.M. (Eds.): *Component-Oriented Enterprise Applications (COEA)*, Vol. P-70 of LNI, pp.87–92.

Simon, H.A. (1996) *The Sciences of the Artificial*, The MIT Press, Massachusetts Institute of Technology, USA.

Skroch, O. and Turowski, K. (2008) 'Validation of architectural targets in business components identification', in Thoben, K-D., Pawar, K.S. and Goncalves, R. (Eds.): *Proceedings of the 14th International Conference on Concurrent Enterprising: ICE 2008*, Lisbon, Portugal, pp.283–290.

Stojanovic, Z. and Dahanayake, A. (2005) *Service-Oriented Software System Engineering: Challenges and Practices*, Idea Group Inc., Hershey, PA, USA.

SUPER (2009) *Semantics Utilized for Process Management within and between Enterprises (Super)*, Web, URL http://cordis.europa.eu/ist/kct/super synopsis.htm

van der Aalst, W. and van Hee, K. (2001) *Workflow Management: Models, Methods and Tools*, MIT Press, MA.

van Reijswoud, V., Mulder, J. and Dietz, J.L. (1999) 'Speech act based business process and information modeling with demo', *Information Systems Journal*, Vol. 9, No. 2, pp.117–138.

Vitharana, P., Zahedi, F. and Jain, H. (2003) 'Design, retrieval, and assembly in component-based software development', *Communications of the ACM*, Vol. 46, No. 11, pp.97–102.

Wang, Z., Xu, X. and Zhan, D. (2005) 'A survey of business component identification methods and related techniques', *International Journal of Information Technology*, Vol. 2, No. 4, pp.229–238.

Winograd, T. and Flores, F. (1986) *Understanding Computers and Cognition: A New Foundation for Design*, Ablex, Norwood, NJ.