Identifying Business Components on the basis of an Enterprise Ontology

Antonia Albani[1], Jan L.G. Dietz[2]

[1]University of Augsburg, Business School
Chair of Business Informatics and Systems Engineering,
86159 Augsburg, Germany
antonia.albani@wiwi.uni-augsburg.de

[2] Delft University of Technology
Chair of Information Systems
PO Box 5031, 2600 GA Delft, The Netherlands
j.l.g.dietz@ewi.tudelft.nl

Abstract. Companies are more and more focusing on their core competencies, outsourcing business tasks to their business partners. In order to support collaboration between business partners, adequate information systems need to be built automating inter-organizational business processes. The bases for such information systems are *business components* combining software artefacts from different vendors to applications which are individual to each customer. The crucial factors in identifying and building reusable, marketable and self-contained business components are the appropriateness and the quality of the underlying business domain models. This paper therefore introduces a process for the identification of business components based on an *enterprise ontology*, being a business domain model satisfying well defined quality criteria.

1. Introduction

The software components of an information system that support directly the activities in an enterprise are usually called *business components* [1, 2]. All other software components are considered either to deliver services to these business components or to offer some general functionality. The identification of business components thus is the first step in the development of an information system according to current standards (component-based, object-oriented etc.). Needless to say that this step is a very crucial one and that it consequently should be performed at the highest possible level of quality. The starting point is the set of requirements that have been elicited from the business domain. Requirements engineering is still a weak link, although considerable progress has been made since it is being based on a business domain model. These models offer a more objective starting point for extracting requirements and a more objective criterion for evaluating them than the traditional 'waiter strategy' [3]. In a very true sense however, this new approach to engineering requirements only shifts the problem to an earlier stage instead of solving

it. The crucial factor now is the appropriateness and the quality of the business domain model. In [4] some quality criteria are proposed regarding a business domain model, which we adopt for our current research:

- It should make a clear and well-founded distinction between the *essential* business actions and informational actions. For example, requesting a supplier to deliver articles is essential, but computing the amount of articles is informational (it is no new fact, only the outcome of a computation).
- It should have the right granularity or level of detail. "Right" means in this respect: finding the actions that are *atomic* from the business point of view. They may be composite only in their implementations. For example, the request to a supplier is atomic from the business point of view, but to perform a request by postal mail, a number of non-essential actions have to be taken like mailing the order form, transporting it and delivering it to the supplier.
- It should be *complete*, i.e. it should contain everything that is necessary and it should not contain anything that is irrelevant. As will be shown in the sequel, this requirement is probably the most hard to satisfy since it is common practice in most organizations to perform several kinds of coordination acts tacitly, according to the rule "no news is good news".

We will call a business domain model that satisfies these requirements an *enterprise ontology*. The goal of the research that is reported in this paper is to identify business components on the basis of an enterprise ontology. It builds on previous work regarding enterprise ontology [5] and regarding business components [6, 7]. The outline of the paper is as follows. In section 2, the method is presented that we apply to arrive at a right ontological model of an enterprise and to derive from this model the business components. In section 3, the method is applied to the example of strategic supply network development (SSND), as reported in [8, 9]. On the basis of the ontological model that is the outcome of section 3, we derive the corresponding business components in section 4. Discussions of the findings as well as the conclusions that can be drawn are provided in section 5.

2. The method and the example case

A precondition to component based development of application systems by using business components is a stable component model. In order to obtain stable business component models, a well defined identification process is necessary. The basis for the identification of reusable, marketable and self-contained business components is an appropriate and high quality business domain model. Such a model not only serves to satisfy the requirements for a single application system but rather for a family of systems – and therefore for a certain domain. In order to achieve that, we adapted the Business Component Modeling (BCM) [6] process by modeling the business domain using an enterprise ontology as introduced in section 1. An overview of the adapted *Component Based Domain Analysis* phase of the BCM process is given in Fig. 1. In this paper we concentrate on the *Domain Scope* and the *Business Components Identification* sub phases and will not describe the *Specification* sub phase.

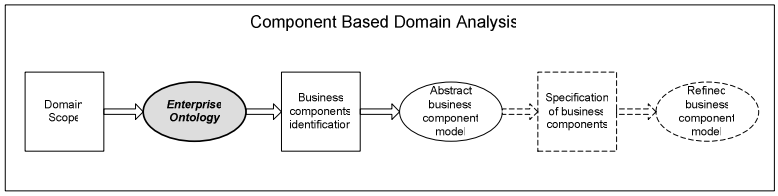

Fig. 1 Component Based Analysis Phase of the BCM process

In the *Domain Scope* sub phase, the method that is used for constructing the ontology of an enterprise is taken from DEMO (Design & Engineering Methodology for Organizations) [10-12]. As is explained in [10, 11] a distinction is made between production acts and facts and coordination acts and facts. The transaction axiom puts these acts/facts together in the standard pattern of the (business) transaction. Consequently, two worlds are distinguished: the production world (P-world) and the coordination world (C-world). The complete ontological model of an organization in DEMO consists of four aspect models (Fig. 2).

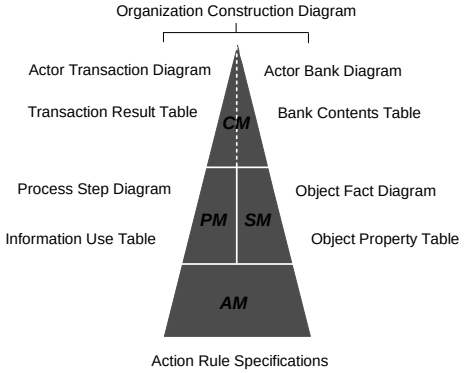

Fig. 2 The four aspect models

The Construction Model (CM) specifies the composition, the environment and the structure of the organization: the identified transaction types and the associated actor roles, as well as the information links between the actor roles and production banks or coordination banks. The Process Model (PM) contains for every transaction type in the CM the specific transaction pattern of the transaction type. Next to these patterns, it contains the causal and conditional relationships between transactions. The Action Model (AM) specifies the action rules that serve as guidelines for the actors in dealing with their agenda. It contains one or more action rules for every agendum type. These rules are grouped according to the actor roles that are distinguished. The State Model (SM) specifies the entity types and fact types in the P-world, but only those object classes, fact types and ontological coexistence rules that are contained in the AM.

In Fig. 2, the CM triangle is split by a dashed line in a left and a right part. This has got to do with the logical sequence of producing the aspect models. First, the left part of the CM can be made straight away after having applied the elicitation procedure as discussed in [13]. It contains the active influences among actor roles, through their

being initiator or executor of a transaction type. The CM is expressed in an Actor Transaction Diagram (ATD) and a Transaction Result Table (TRT). Next, the Process Step Diagram (PSD), which represents a part of the PM, is produced, and after that the AM, which is expressed in Action Rule Specifications (ARS). The action rules are expressed in a pseudo-algorithmic language, by which an optimal balance is achieved between readability and preciseness. Then the SM is produced, expressed in an Object Fact Diagram (OFD) and an Object Property Table (OPT). Next, the right part of the CM is produced. It consists of an Actor Bank Diagram (ABD) and a Bank Contents Table (BCT). Usually the Actor Bank Diagram is drawn as an extension of the Actor Transaction Diagram. Together they constitute the Organization Construction Diagram (OCD). After that we are able to complete the PM with the Information Use Table (IUT).

Having defined the enterprise ontology, the complete information related to the business domain is available in order to identify business components as denoted in the *Business Components Identification* sub phase of the BCM process in Fig. 1. In order to optimize the process of identifying high quality, reusable and marketable business components the Business Components Identification (BCI) method is used. BCI is based upon the Business System Planning (BSP) [14] method and has been modified for the field of business components identification. BCI takes as input the object classes and fact types from the SM and the process steps from the PM, obtained from the domain scope phase and summarized in the Create/Use Table (CUT), an extension of the IUT. Using a genetic algorithm a number of possible solutions (component models) are generated in order to select the most suitable solution fitting best to the specified quality factors. One of the most important quality factors concerning component models is the minimal communication between components. The result of the BCI is an abstract business component model with defined dependencies between components.

To illustrate the domain scope and component identification sub phases with their resulting diagrams and models, the BCM process is applied to the domain of strategic supply network development in the next sections. The main tasks in the domain of strategic supply network development derive from the tasks of strategic sourcing. The most evident changes regard the functions with cross-enterprise focus. Purchasing has become a core function in enterprises in the 90ies. Current empiric research shows a significant correlation between the establishment of a strategic purchasing function and the financial success of an enterprise, independent from the industry surveyed [15]. One of the most important factors in this connection is the buyer-supplier-relationship. At many of the surveyed companies, a close cooperation between buyer and supplier in areas such as long-term planning, product development and coordination of production processes led to process improvements and resulting cost reductions that were shared between buyer and suppliers [15]. In practice, supplier development is widely limited to suppliers in tier-1. With respect to the superior importance of supplier development we postulated the extension of the traditional frame of reference in strategic sourcing from a supplier-centric to a supply-network-scope [8] i.e., the further development of the strategic supplier development to a strategic supply network development. This refocuses the object of reference in the field of strategic sourcing by analysing supplier networks instead of single suppliers.

3. Constructing the ontology of the SSND case

This section contains the result of applying the method for constructing the ontology of an enterprise, as presented above, to the SSND case. Space limitations prohibit us to provide a more extensive account of how the models in the figures below are arrived at. Also, we will not present and discuss the action rules. Fig. 3 exhibits the Organization Construction Diagram (OCD). The Transaction Result Table (TRT) that belongs to it is shown in Table 1.

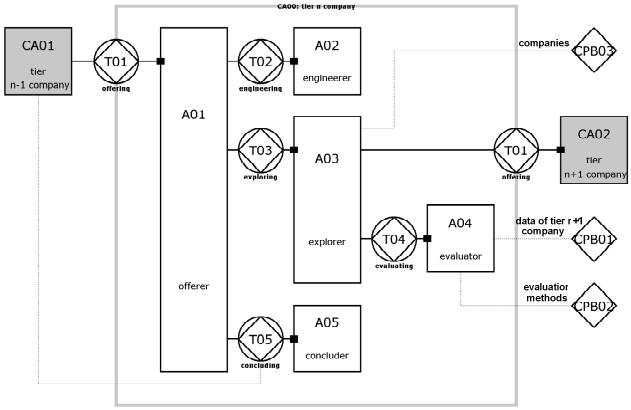

Fig. 3. Organization Construction Diagram of the SSND case

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

Table 1. Transaction Result Table of the SSND case

The top or starting transaction type is T01. Instances of T01 are initiated by the environmental actor role CA01, which is a company in tier n-1 and executed by actor role A01. This company asks for an offer regarding the supply of a particular product P. In order to make such an offer, A01 first initiates a T02, in order to get the bill of material of P. This is a list of (first-level) components of P, produced by A02. Next, A01 asks A03 for every such component to get offers from companies that are able to supply the component. So, a number of transactions T03 may be carried through within one T01, namely as many as there are components of P. In order to execute each of these transactions, A03 has to ask companies for an offer regarding the supply of a component of P. Since this is identical to the starting transaction T01, we model this also as initiating a T01. Now however, the executor of the T01 is a company in tier n+1. Consequently, the model that is shown in Fig. 3 must be understood as to be repeated 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 P.

Every offer from the companies in tier n+1 is evaluated in a T04. So, there is a T04 for every 'output' T01. The result of a T04 is a graded offer for some component of P. So, what A03 delivers back to A01 is a set of graded offers for every component of P. Next, A01 asks A05, for every component of P, to select the best offer. The result is a set of concluded offers, one for every component of P. This set is delivered to A01. Lastly, A01 delivers a contract offer for supplying P, together with the set of concluded offers for delivering the components of P. Because of the recursive character of the whole model, this offer includes the complete bill of material of P, regardless its depth. The OCD in Fig. 3 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 T04, one of these methods is applied. CPB03 contains identifiers of all companies that may be addressed for an offer. Lastly, in the transaction result table (Table 1), the supply of product by a (supplying) company to a (customer) company is called a contract.

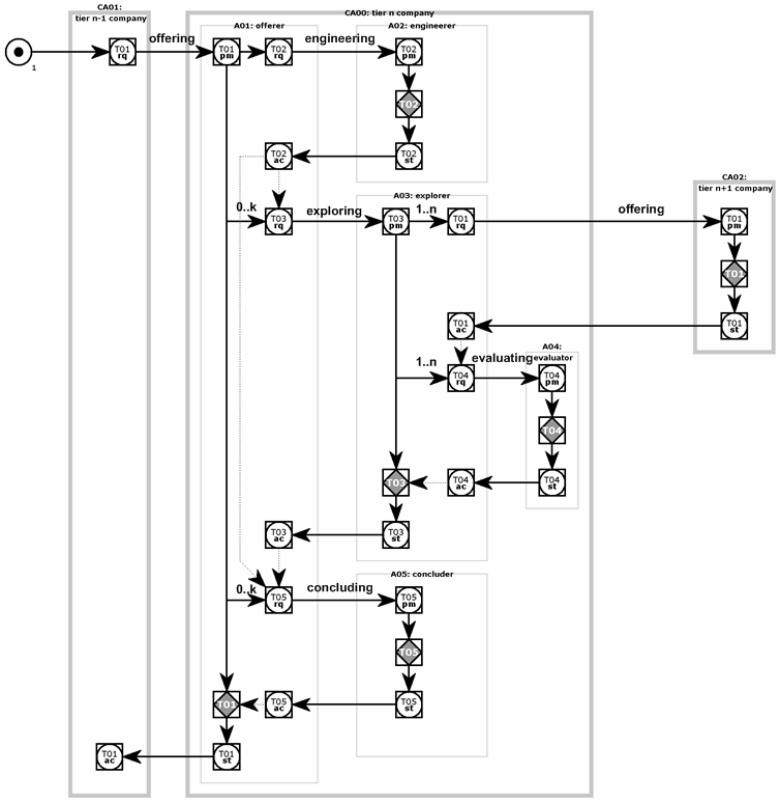

Fig. 4 Process Step Diagram of the SSND case

Fig. 4 exhibits the process step diagram of the SSDN case. It shows how the distinct transaction types are related. From the state T01/pm (promised) a number of transactions T03 (possibly none) and a number of 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 T03/pm, a number of transactions T01 and a number of 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 a T03/rq has to wait for the being performed of the T02/ac. Fig. 5 exhibits the object fact diagram and Table 2 the object property table. Together they constitute the State Model of the example case.

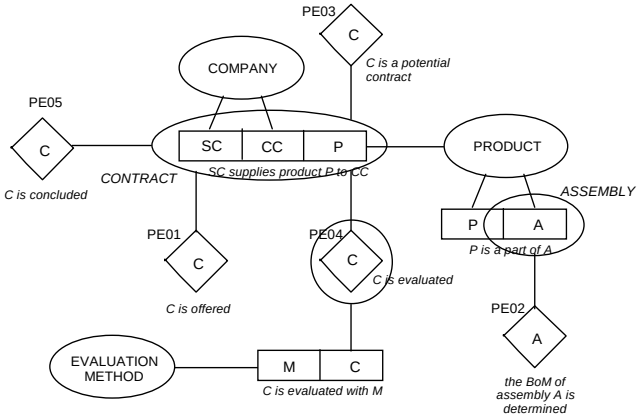

Fig. 5 Object Fact Diagram of the SSND case

property type	**object class**	**scale**
< company information >	COMPANY	< aggregated data >
< contract terms >	*CONTRACT*	< aggregated data >
sub_products(*)	PRODUCT	set of PRODUCT
#sub_products(*)	PRODUCT	NUMBER
companies(*)	PRODUCT	set of COMPANY
sub_contracts(*)	*CONTRACT*	set of *CONTRACT*
evaluation_mark	*CONTRACT*	NUMBER
evaluation_marks(*)	*CONTRACT*	set of NUMBER

Table 2. Object Property Table of the SSND case

Properties are nothing more or less than binary fact types that happen to be a pure mathematical function of which the range is set of, usually ordered, values, called a scale. The OFD is a variant of the ORM model [16]. Diamonds represent unary fact types that are the result of transactions, also called a production event type. They correspond with the transaction results in Table 1. An ellipse 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 ellipse around the ternary fact type “SC supplies product P to CC” defines the concept of contract. The ellipse around the production event type “C is evaluated” defines the concept of evaluated contract. Lastly, the ellipse 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. The property types marked by “(*)” in the OPT are derived fact types. The derivation rules are as follows:

sub_products (P) = {X | X is a product **and** X is a part of P};
#sub_products(P) = card(sub_products (P));
companies(P) = {X | X is a company **and** X supplies P to the ‘this company’};

sub_contracts(C) = {X | X is a contract **and** the product of X is Z **and** the product of C is Y **and** Z is a part of Y};
evaluation_marks (C) = {X | X is an evaluation mark of C};

object class or fact type	**process steps**
PRODUCT	T01/rq T01/pm T02/rq T02/pm *T02/st* T02/ac T03/rq T03/pm
product P is a part of product A	*T02/st* T03/pm
the BoM of assembly A is determined	*T02/ac*
COMPANY	T01/rq T03/pm T04/pm
< company information >	T04/pm T04/st
CONTRACT	T01/rq T01/pm T02/ac T03/rq *T03/pm* T01/st T01/ac T04/rq T04/pm T04/st T04/ac T03/st T03/ac T05/rq T05/pm T05/st T05/ac
< contract terms >	*T05/st*
supply contract C is offered	*T01/ac*
supply contract C is a potential contract	*T03/ac*
supply contract C is evaluated with method M	*T04/ac*
supply contract C is concluded	*T05/ac*
EVALUATION METHOD	T04/pm T04/st
sub_products(*)	*T02/st* T03/rq T03/pm
#sub_products(*)	*T02/st* T03/rq
companies(*)	T03/pm
sub_contracts(*)	*T03/pm* T03/st T01/st T03/ac T04/rq T04/ac
evaluation_mark(*)	*T04/st* T04/ac T03/st T04/pm T03/ac
evaluation_marks(*)	*T03/st*

Table 3. Create/Use Table of the SSND case

Table 3 exhibits the Create/Use Table of the SSND case. It consists of an Information Use Table, extended with the process steps in which an instance of an object type or fact type is created. These steps are printed in italics, as are the fact types that are production event types.

4. Identifying Business Components for the SSND case

Based on the enterprise ontology introduced in the previous section and providing a complete and formal description of the business domain, business components for the domain of strategic supply network development are identified in this section and the resulting component framework is introduced.

The underlying idea of business components combines components from different vendors to an application which is individual to each customer. This principle of modular black-box design demands reusable, marketable, self-contained, reliable and manageable business components. Therefore the business components need to provide services at the right level of granularity and a formal and complete specification of its external view. The description of the specification step of the BCM process is not within the scope of this paper. Instead, for the identification of business components, an enhanced version of the Business Component Identification (BCI) [6] method is applied to the SSND domain and is described next.

The basis for BCI builds the Create/Use table (see Table 3) of the enterprise ontology. In a first step a matrix is built defining the relationships between the object class respectively fact types and the single process steps, gained from the Create/Use Table. The relationships are visualized inserting “C” and “U” in the matrix. “C”

denotes that the object class respectively fact types are *created* in a specific process step and "U" denotes the *usage* of informational data by a given process step. In changing the order of the rows and the columns and placing the "C" as far left and as up as possible in the matrix, groups of relationships can be recognized (see Fig. 6). These groups identify potential business components.

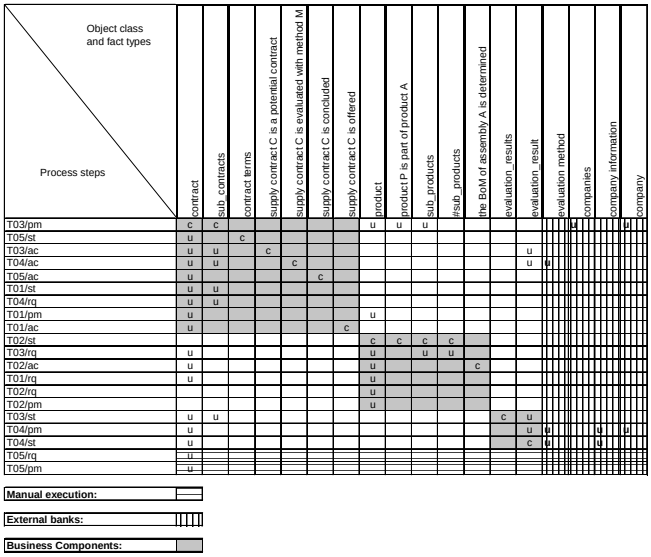

Fig. 6 Business Components Identification Matrix

In order to ensure optimal grouping regarding required metrics – such as minimal communication between components, maximum compactness of components – an optimization problem needs to be solved for which a genetic algorithm has been developed. The genetic algorithm starts with a predefined solution (specific assignment of process steps to components) and while iterating, generates better solutions using mutation and crossing-over of the best generated solutions available. For any iteration, the algorithm assigns each process step to a potential component (1, 2, 3, etc) and evaluates the potential solution with the following quality function:

$$q = \sum_{ps} \sum_{io} e(m(ps, io), p(ps), r(io)) \tag{4.1}$$

ps: process step in the matrix (row)
io: information object (object class or fact type) in the matrix (column)

$p(ps) \mapsto \mathbb{N}$: assignment of a *ps* to a component $\in \mathbb{N}$ (4.2)

$r(io) \mapsto \mathbb{N}$: assignment of an *io* to a component $\in \mathbb{N}$ (4.3)

$m(ps,io) \in \{'c','u','o'\}$: content of matrix entry (*c*reate; *u*se; *o*ther) (4.4)

$$e(m(ps,io),p(ps),r(io)) = \begin{cases} e_{in}(m(ps,io)) : p(ps) = r(io) \\ e_{out}(m(ps,io)) : p(ps) \neq r(io) \end{cases} \tag{4.5}$$

$$e_{in}(m(ps,io)) = \begin{cases} 0 : m(ps,io) = 'u' \vee 'c' \\ 0.1 : m(ps,io) = 'o' \end{cases} \tag{4.6}$$

$$e_{out}(m(ps,io)) = \begin{cases} 0 : m(ps,io) = 'o' \\ 1 : m(ps,io) = 'u' \\ +\infty : m(ps,io) = 'c' \end{cases} \tag{4.7}$$

The evaluation function (4.5) evaluates for each potential solution any entry in the matrix as follows. Entries which are a *use* or a *create* and located in a component are evaluated with the value 0, anything else is evaluated with a 0.1 (see formula 4.6). Entries outside a component which are a *use* are evaluated with 1 and a *create* with $+\infty$ (see formula 4.7). The quality function (4.1) calculates the entire sum over each evaluation for all entries in the matrix. After all iterations the *min(q)* provides the best component solutions for a given matrix, whereby different solutions with the same *min(q)* may exist. That means that all *creates* are located inside a component, few *uses* are outside of the components and few *other* (empty entries) are inside a component, fulfilling the required metrics of minimal communication and maximum compactness.

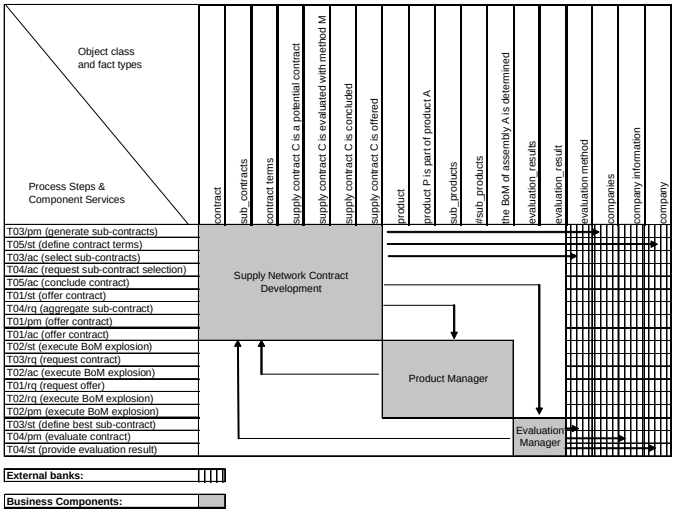

Fig. 7 Identified SSND Business Components

The result of the BCI is an abstract business component model with already defined dependencies between components. The dependencies are defined by the *uses* which are located outside the components and which are substitute by arrows as shown in Fig. 7. The first business component shown in Fig. 7 offers services for the development of the contract offers for the whole supply network and is therefore called *supply network contract development*. The second business component is responsible for the specification of products and for the execution of the bill of material explosion and is therefore called *product manager*. The last component identified is the one responsible for the *evaluation* of the offered contracts.

For the business components identified, the services they provide need to be defined. Single process steps need therefore to be assigned to component services. The mapping for the strategic supply network development domain is shown in Fig. 7 in the row *process steps and component services.* As can be seen, some process steps are mapped one to one to business component services, e.g. *T03/pm* is mapped to *generate sub-contracts*, or *T05/st* is mapped to *define contract terms*, whereas other process steps are combined to one business component service e.g. *T01/st, T01/pm, T01/ac* are all mapped to one and the same component service *offer contract.* Fig. 8 shows the refined business component model with the services defined for each component.

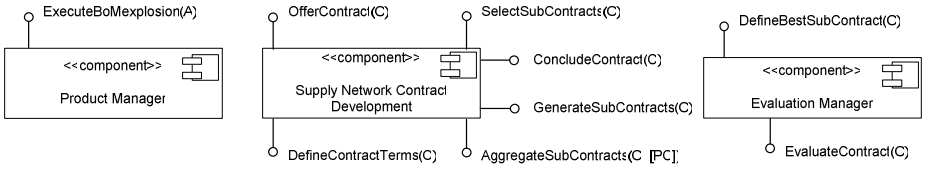

Fig. 8 SSND Business Components with provided Services

The information for the mapping of process steps to business component services is gained from the action rules of the enterprise ontology. A detailed description of the mapping would go beyond the scope of this paper.

5. Conclusion

In this paper, we have addressed the problem of identifying business components, defined as the highest level software components, i.e. the software components that directly support business activities. Although component-based software development may be superior to more traditional approaches, it leaves the problem of requirements engineering unsolved. Put differently, the component-based approach provides no criteria for determining that a set of identified components is complete (no component is missing) and minimal (there are no redundant components). Next to that, it provides no means for determining the point of departure, i.e. an appropriate model of the business domain.

The method presented and demonstrated in this paper does solve these problems. First, the enterprise ontology constructed by means of DEMO is an appropriate model of the business domain. It satisfies the quality criteria as proposed in the introduction. As a consequence, the identified business components do really and directly support the business activities in which original new facts are created. Moreover, since the process steps (cf. Fig. 4) are atomic from the business point of view, one can be sure to have found the most fine level of granularity that has to be taken into account. Also, one can be sure that this set of process steps is complete and minimal.

Second, the BCI method, based on the resulting models and tables of the enterprise ontology, provides an automated approach for the identification of business components satisfying defined metrics which are relevant for the development of reusable, marketable and self-contained business components. The metrics defined in this paper – being minimal communication between and maximum compactness of

business components – are the basic metrics for the component-based development of inter-organizational business applications focusing on the deployment of components which can be on different enterprise systems. Additional metrics can easily be added to the BCI method in extending the genetic algorithm.

6. References

1. Turowski, K. and J.M. Zaha, Methodological Standard for Service Specification. International Journal of Services and Standards, 2004.
2. Turowski, K., Fachkomponenten: Komponentenbasierte betriebliche Anwendungssysteme. 2003, Aachen: Shaker Verlag.
3. Dietz, J.L.G. and J.A. Barjis. Petri net expressions of DEMO process models as a rigid foundation for requirements engineering. In 2nd International Conference on Enterprise Information Systems. 2000. Escola Superior de Tecnologia do Instituto Politécnico, Setúbal: ISBN: 972-98050-1-6.
4. Dietz, J.L.G. Deriving Use Cases from Business Process Models. In Conceptual Modeling - ER 2003, LNCS 2813. 2003: Springer Verlag.
5. Dietz, J.L.G. and N. Habing. A meta Ontology for Organizations. In Workshop on Modeling Inter-Organizational Systems (MIOS), LNCS 3292. 2004. Larnaca, Cyprus: Springer Verlag.
6. Albani, A., et al. Domain Based Identification and Modelling of Business Component Applications. In 7th East-European Conference on Advances in Databases and Informations Systems (ADBIS-03), LNCS 2798. 2003. Dresden, Deutschland: Springer Verlag.
7. Albani, A., et al. Identification and Modelling of Web Services for Inter-enterprise Collaboration Exemplified for the Domain of Strategic Supply Chain Development. In On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE - OTM Confederated International Conferences CoopIS, DOA, and ODBASE 2003, LNCS 2888. 2003. Catania, Sicily, Italy: Springer Verlag.
8. Albani, A., et al. Dynamic Modelling of Strategic Supply Chains. In E-Commerce and Web Technologies: 4th International Conference, EC-Web 2003 Prague, Czech Republic, September 2003, LNCS 2738. 2003. Prague, Czech Republic: Springer-Verlag.
9. Albani, A., C. Winnewisser, and K. Turowski. Dynamic Modelling of Demand Driven Value Networks. In On The Move to Meaningful Internet Systems and Ubiquitous Computing 2004: CoopIS, DOA and ODBASE, LNCS. 2004. Larnaca, Cyprus: Springer Verlag.
10. Dietz, J.L.G., The Atoms, Molecules and Fibers of Organizations. Data and Knowledge Engineering, 2003. 47: p. 301-325.
11. Dietz, J.L.G. Generic recurrent patterns in business processes. In Business Process Management, LNCS 2687. 2003: Springer Verlag.
12. van Reijswoud, V.E., J.B.F. Mulder, and J.L.G. Dietz, Speech Act Based Business Process and Information Modeling with DEMO. Information Systems Journal, 1999.
13. Dietz, J.L.G. The notion of business process revisted. In OTM confederated International Conferences (CoopIS, DOA, ODBASE), LNCS 3290. 2004. Larnaca, Cyprus.
14. IBM, Business Systems Planning-Information Systems Planning Guide, ed. I.D. (Ed.). 1984, Atlanta: International Business Machines.
15. Carr, A.S. and J.N. Pearson, Strategically managed buyer - supplier relationships and performance outcomes. Journal of Operations Management, 1999. 17: p. 497 - 519.
16. Halpin, T.A., Information Modeling and Relational Databases. 2001, San Francisco: Morgan Kaufmann.