From Reference Model to Component Model

Antonia Albani, Johannes Maria Zaha

Chair of Business Information Systems and Systems Engineering
Business Faculty
University of Augsburg
Universitaetsstrasse 16, 86135 Augsburg, Germany
{antonia.albaniljohannes.maria.zaha}@wiwi.uni-augsburg.de

Abstract. Stable component models are an essential prerequisite for developing customer-individual business applications. Thereby the information for the identification and specification of their components is gained from domain models. Reference models constitute a potential source for building enterprise-specific domain models. Based on the analysis of existing reference models, this article shows how information available through reference models can be used for the development of stable component models. The derivation of information required for the identification and specification of reusable business components is discussed using example reference modeling techniques. Additionally, potential extensions of existing reference modeling techniques are shown.

Keywords: reference model, component model, specification, business component, software component

1. Introduction

The idea of developing application systems from prefabricated software components [1] has been traced at least since the publication of McIllroy in 1968 [2]. Building component-based customer-individual application systems requires software markets [3] where software components of different producers can be exchanged and composed in order to meet individual requirements. The reuse of components is primary enabled by standards, which consider domain-specific artefacts as well as technical aspects.

Reference models are standardized descriptions of a specific business domain and are generated from concrete implementations in enterprises as well as from the evaluation of best practices. Therewith they have a recommendation character for the development of domain-specific application systems. In case of component-based development, information contained in a reference model could be used to derive reusable component models, covering the relationships between single components and a complete specification of each software-component. A specification describes the external view of a software-artefact and considers business-related as well as technical aspects (cf. [4]). The aim of this paper is to define which information

reference models provide for the identification and specification of business components.

Based on the Business Component Modeling (BCM) Process (cp. [5],[6]) for deriving component models, the usage of reference models for all process steps in the component-based domain analysis phase is examined. The three activities in this process are the domain scope, the Business Component Identification (BCI) and the standardized specification of software components. In the domain scope phase a complete description of the domain model is developed, which is completely covered by reference models. For the remaining steps, the information that is gained from such reference models is identified in this paper. Moreover the information which is missing in reference models, but necessary for the identification and specification of components, is named.

For this purpose in chapter two the BCM process is introduced giving a short explanation of the single steps and a definition of business components and their specification. In chapter three the commonalities of reference models are examined and identified in order to evaluate their usage for deriving component models. Having all reference models actually describing the same views on a business domain, one specific technique for describing reference models is selected and used in chapter four. By means of an example domain in the field of asset accounting, available and missing information for the derivation of component models is discussed. Chapter five summarizes the outcome of this survey. Conclusions are drawn and an outlook on future work is given in chapter six.

2. Business Component Modeling (BCM) Process

According to the definition of the working group Wi-kobAS of the German informatics society (GI) [7] a component is defined as follows:

A *component* consists of different (software) artefacts. It is reusable, self-contained and marketable, provides services through well-defined interfaces, hides its implementation and can be deployed in configurations unknown at the time of development. A *business component* is a component that offers a certain set of services of a given *business* domain.

To integrate business components to customer-individual application systems the establishment of content-related, functional and methodical standards as well as the standardization of domain-specific functions and interfaces is needed. Therefore the working group has introduced a specification framework (see Fig. 1), defining notations which have to be regarded for the specification of business components in order to simplify their reusability between companies and software developers.

In this framework standardized techniques for the specification of business components of the different levels of abstraction have been chosen, like the Interface Definition Language (IDL) [8] on Interface Level, the Object Constraint Language (OCL) [9] on Behavioural Level or the Restructured Business Language [10] on Task Level. With the specification framework a methodical standard was set, which defines the techniques to completely specify the external view of business components. This

framework constitutes a methodical standard, which considers business-related as well as technical aspects of business components on seven layers.

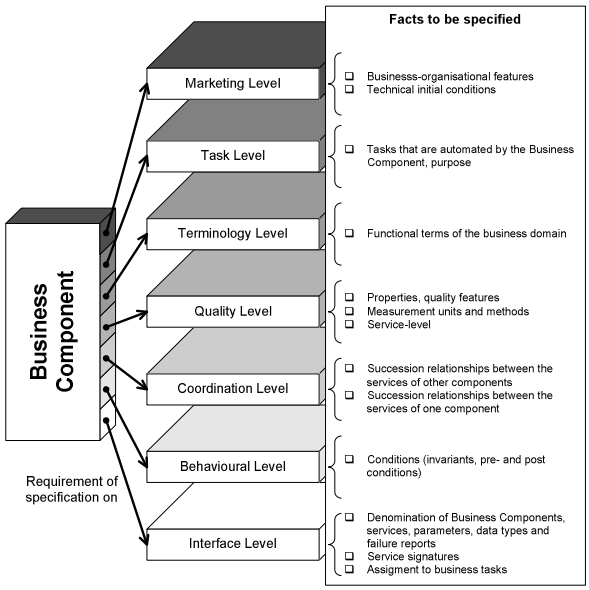


Fig. 1. Multi-layer specification framework (cf. [4])

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 derivation process is necessary. Based on the fact that business components do not only satisfy the requirements for a single application system but rather for a family of systems – and therefore for a certain domain – the derivation process requires throughout all development phases the consideration of the specific business domain. Several process models for component-based software engineering exist (cf. [5, 11-14]).The only one considering the identification and specification of business components in detail is the BCM process [5] which is used in this paper and is shortly introduced next.

As depicted in table 1, the BCM process is divided in the two phases Component Based Domain Analysis and Component Based Domain Design, whereby during the whole process the underlying domain is considered. This is vital for a stable component model, because business components shall not cover the demands of only one application but the demands of several applications within the domain.

During the sub phase domain scope in the phase *Component Based Domain Analysis* the domain of interest is identified, characterised and business processes with their functional tasks are defined. In addition, data is collected to analyse the information objects and their relationships. Possible sources of domain information include reference models, existing systems in the domain, domain experts, handbooks, requirements on future systems, market studies, and so on. This information is prerequisite for the building of business components with the Business

Component Identification (BCI) method, an extension of Business System Planning (BSP) [15] for the field of component-based software engineering.

BCM Phase	BCM sub phases	Performed Tasks
Component Based Domain Analysis	Domain scope	Identification and characterisation of the domain
		Definition of the business processes and functional tasks of the domain
		Data collection and definition of information objects
		Identification of relationships between information objects
	Business Component Identification (BCI)	Grouping of functional business tasks and informational objects for the identification of business components
	Standard specification of business components	Specification of all business component levels (marketing, task, terminology, quality, coordination, behaviour, interface)
Component Based Domain Design	Business components collaboration design	Definition of component instances
		Definition of dependencies between component instances
		Identification of service calls between component instances

Tab. 1: summary of the BCM process

BCI [5] takes as input the business tasks of a specific domain and the information objects and arranges them in a matrix, so that the relationships between them are depicted. The arrangement of the matrix is after that modified by the exchange of lines and columns to find candidates for components (cluster), which are optimized with respect to their communication relationships.

In the next sub phase all components are refined and specified on all layers of the specification framework introduced above.

In the phase *Component Based Domain Design* cooperation of the components and the allocation of the component-instances on different systems is constituted. This phase is not considered in this paper, since no information for this phase can be derived from reference models.

The derivation of component models according to the BCM process [5] requires – as introduced above – multifaceted information of the underlying domain. How far reference models can be used to provide this information will be shown in chapter four, after a generalizing appreciation of reference models for the purpose of this paper in chapter three.

3. Commonalities of reference models

In the area of business information systems for specific industries, branches of industries, industrial sectors or even smaller application domains a lot of work has been done in the field of reference models (cp. e.g. [16], [17], [18], [18], [19]). Basically an economic reference model is understood as an information model, which is developed or used to support the construction of application models. Reference models provide content-related support in construction processes. Even if different

modeling techniques are used in reference models they all describe the same views on a specific domain, namely functional, data and process view. This is illustrated in table two, which shows a selection of reference models [16, 17, 20-22], their views and modeling techniques.

reference model	*views*	*modeling technique*	*observation artifact*	*source*
EC	functional view	UML-diagramm type	electronic commerce	[19] [20]
	data view	object model		
	process view	UML-diagramm type		
SSCD	functional view	functional decomposition diagram	strategic purchasing	[6]
	data view	object model		
	process view	activity diagram		
Y-CIM	functional view	functional decomposition diagram	industrial firm	[15]
	data view	entity relationship-diagram		
	process view	event-driven process chain		
Handels-H	functional view	functional decomposition diagram	business concern	[16]
	data view	entity relationship-diagram		
	process view	event-driven process chain / workflow model		

Tab. 2: overview over reference models and depiction techniques

These views are according to [16] defined as follows: the *functional view* describes performed activities (functions), their decomposition in sub-functions as well as the existing relationships between them. The *data view* displays events (e.g. customer order has arrived) and states (e.g. state of customer, state of article) as information objects represented by data. The connection between functional and data view is established by the *process view*, where succession relationships between functions are defined and information objects are assigned to functions.

In order to derive component models from reference models, the examination of the considered views is not sufficient. Relevant therefore is the *information* modeled in a specific view of a reference model which is constituted by the used modeling technique. Hence, it is necessary to demonstrate the identity of the objects represented by the modeling artefacts provided by the notation of a specific reference model view. This identity is illustrated at the process view. According to ([17], p. 53), the following questions need to be answered on the process view:

- Which data is needed to perform a specific functions and which data is created by the functions?
- Which data is required by which organizational unit and which organizational unit is allowed to manipulate specific information?
- Which organizational unit performs which functions?

Modeling techniques on process view are e.g. directional graphs like Petri-nets, event-driven process chains or activity diagrams. The information displayed by these modeling techniques is identical and provides the necessary information to answer the questions stated above. The identity of the represented information is also true for the modeling techniques used on data view and functional view.

Having the fact that different reference models do not only describe the same views on a specific domain, but the same views of different reference models do also represent the same information using different modeling techniques, a general statement can be made for the derivation of component models from reference models by depicting one specific reference model. Therefore in the following chapter the

Handels-H-reference model (cf. table 2), describing the business domain in the field of asset accounting (cp. [17], p. 368), is used to elaborate the provided and missing information for the identification and specification of business components in general.

4. From reference model to component model

Considering the Handels-H-reference model as an example, the information available in the different views – functional view, data view and process view – that can be used to identify and completely specify business components will be examined. As an example business domain the area of asset accounting (cp. [17], p. 368) has been chosen. The modeling techniques in the example reference model are the functional decomposition diagram on functional view, the Entity-Relationship-diagram on data view and the event-driven process chain on process view.

4.1 Functional view

Functional decomposition diagrams [23] are a well known modeling technique for describing the functional view. This technique allows the decomposition of business functions in their corresponding sub-functions. An example functional decomposition diagram is shown in Fig. 2. It belongs to the reference model introduced above and is used in order to discuss information that can be used for the specification of business components from the functional view. Information in the functional decomposition diagram needs to be gained in order to be able to identify and specify the business components on each level of abstraction [7]. For better illustration, examples are given on some levels of abstraction using the notation proposed in the memorandum for standardized specification of business components [7], as introduced in Fig. 1. Information necessary for the identification and specification, but not gained from the functional view, is discussed as well.

Business Component Identification: information gained from functional view

For executing the BCI process step, business functions as well as information objects from the semantic model are required. The functional view of the example reference model provides the business functions. They are gained from the leaves of the functional decomposition diagram.

Business Component Specification: information gained from functional view

The purpose of the *marketing level* (see specification framework in Fig. 1) is to specify features of business components that are important from the business-organizational point of view. Apart from features that describe business related and semantic properties – e.g. name of the component, branch of economic activity, business domain – technical conditions are necessary as well. Examples of technical features are the scope of supply in order to determine which artifacts the component comprises, the specification of the component technology that has been used and the component version.

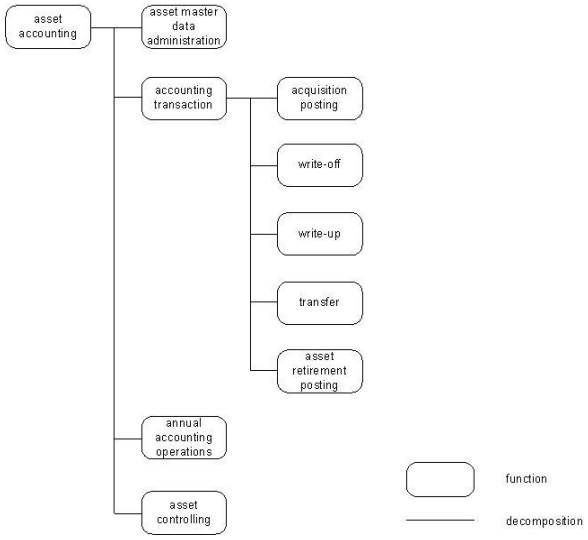


Fig. 2: Functional decomposition diagram *asset accounting* (see [17], p. 368)

From the functions modeled in functional decomposition diagrams it is possible to gain information for the specific business component regarding the business domain and the branch of economic activity. E.g. from the functions and sub-functions displayed in the example diagram (see Fig. 2) the business domain *accounting* can be identified. The economic sector a component could be applied to is not obtainable from the functional decomposition diagram, since asset accounting is not specific for a sector of industry. Information about the naming of the component which is needed for marketing reasons can be retrieved from the function *asset accounting*. Since the functional decomposition diagram describes only the functional requirements of a system, no information for describing the technical features of the component can be retrieved.

The documentation of tasks supported by the business component and their decomposition in sub-tasks is given on *task level*. From a functional decomposition diagram containing functions and sub-functions, information about tasks and sub-tasks can be obtained by mapping the functions and sub-functions to tasks and sub-tasks. Example business tasks, gained from the diagram in Fig. 2, are *accounting transaction, write-off, write-up, transfer* etc. Despite accompanying texts used to describe the functionality of the related reference models, structured and detailed information – as needed for the specification of business components on task level – is missing.

At all different levels the specification of business components uses technical terms having a domain specific functional meaning. Generally, these terms do not have an unequivocal meaning or definition and, hence, have to be specified on the terminology level to guarantee their unequivocal use. From the functions in the functional decomposition diagram, some terms, which are relevant for the specification of the business component on *terminology level*, can be obtained. Examples of such terms are *accounting transaction, acquisition posting, write-off, write-up,* etc. A definition of those terms does not emerge from the reference model

diagram. A distinct meaning of the single terms is therefore missing for the specification at terminology level.

Non-functional properties of a business component are specified on *quality level*. Examples are availability, performance properties or maintenance needs for the services offered. The specification on this level has to determine suitable quality criteria, the appropriate measures and methods for their actual measurement and, if appropriate, specification of service levels. The functional decomposition diagram describes business and therefore functional requirements of a system to be built. Therefore no information for the specification of business components on quality level can be gained from the functional view.

The specifications on *coordination level* describe succession relationships between services and synchronization requirements. Purpose of the coordination level is to provide relevant information of how the business component can be integrated in a component based software solution from a process point of view. From the decomposition of functions in sub-functions – information provided by the functional decomposition diagram – no succession relationship between functions, and therefore between services, can be defined. The functional decomposition diagram thus does not provide any information for the specification on coordination level.

The specifications on the *behavioral level* serve as detailed description of the business component behavior. That means that the behavior of a component is specified in general and in problem situations. Additionally, invariants, pre- and post-conditions of single services need to be specified. According to the coordination level, the functional decomposition diagram does not provide any information for specification on behavioral level.

On *interface level* the denomination of services that are offered publicly by a business component, furthermore public attributes, variables and constants, the definition of special data types, the definition of signatures of offered services, and the declaration of error messages and exceptions are specified. Indirectly the functional decomposition diagram provides information through mapping terms defined on the terminology level to data types or through denomination of services by mapping tasks to services. E.g. the term *asset retirement* can be mapped to the data type `asset_retirement` and the task *asset retirement posting* can be mapped to the service `void posting(asset_retirement a);` Detailed information, e.g. regarding the naming of available attributes, variables, constants, or about the definition of specific data types, is not deducible from terms and tasks gained from the functional decomposition diagram.

4.2 Data view

Entity-Relationship-Diagrams (ERD), based on the definition of Chen [24], are often used to model data structures. A specific type of ERD has also been used in the example reference model.

Business Component Identification: information gained from data view

As already mentioned in section 4.1, apart from business functions it is also necessary to define the information objects for executing the BCI process step. The data model provides this information in form of its Entity-types (see Fig.3).

Business Component Specification: information gained from data view

For the specification of business components on *marketing level*, the data model does not provide any information. The reason is that the data types are not assigned to a specific business domain and that they are independent of the branch of economic activity.

The same holds also for specification on *task level*, since no assignment of functionality to the objects in the model is available.

From the data model it is possible to infer terms – either from entity-types or from relationship-types – and their relationships, needed for the specification on *terminology level*. E.g. terms like *asset, asset group* and *cost center* or relations like *an asset is assigned to an asset account* can be gained from the example data model in Fig. 3. According to the functional view, detailed definitions of the terms are missing despite the fact that reference models provide additional information through accompanying texts. Therefore a complete specification of the business component on terminology level is not possible.

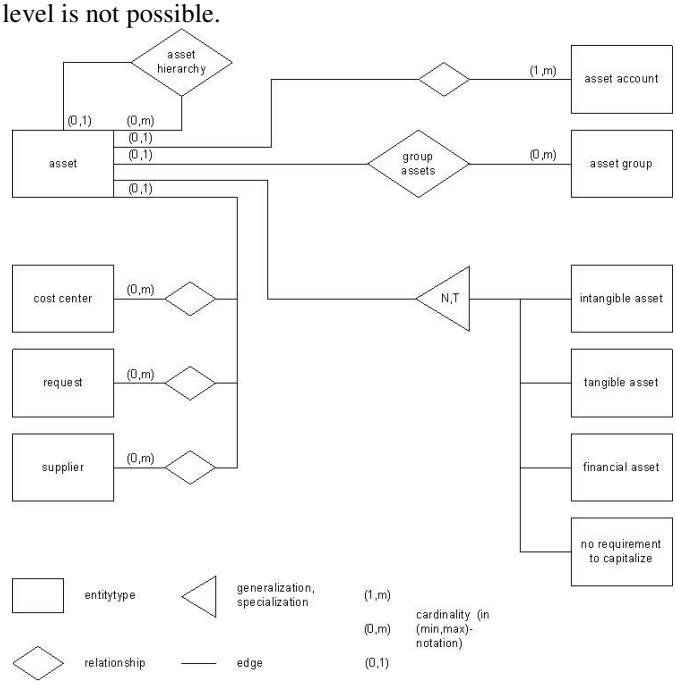


Fig. 3: Data model *asset accounting* (see [17], p. 372)

In modeling objects and their relationships no information about quality features of the business component can be obtained. That means that no information for specifying business components on *quality level* is available through ER-diagrams.

On *coordination level* too, entity relationship models do not provide information for specifying the components, since no functions and no succession relationships between those functions are described.

Through cardinality constraints, characterizing the connection between objects, information about specific invariants, needed for the specification on *behavioral level*, can be gained. Beside invariants no additional information for describing pre- and post-conditions of services is available.

For the specification of components on *interface level* data-types can be mapped either from entity-types or relationship-types provided through the data model. Whereas no information for the naming of available services, attribute, variables, constants, parameters, return values and error messages can be gained from ER-diagrams.

4.3 Process view

The process view serves as documentation of the process-oriented organization. In order to model the process view, event driven process chains [25] are used in the example reference model for commercial enterprises [17]. The example process model for *asset acquisition posting* (see [17], p. 375) is shown in Fig. 4.

Business Component Identification: information gained from data view

Information still missing for the execution of the BCI process step is the assignment of information objects to business functions. This information could in principle be gained from the process view, but is not provided by the example reference model. The reason is that no extended version of event driven process chains is used. Therefore no information of the process view can be gained for the identification of business components.

Business Component Identification: information gained from data view

From the functions used in the event driven process chains no information e.g. regarding business domain, branch of economic activity are given in order to be able to specify the business component on *marketing level*. Whereas information about tasks can be gained from the business process functions and be mapped to component services. Like the other two diagrams presented, terms for the specification on *terminology level* can be obtained from the diagram, but a detailed definition of the terms is missing as well.

Equal to the other two diagrams presented, quality features needed for the specification of components on *quality level* can not be attained from the process diagram.



Fig. 4: Process model *asset acquisitions posting* (see [17], p. 375)

Succession relationships between functions instead are apparent and make it possible to define succession relationship between business components services; provided that functions are mapped to tasks and tasks to component services. E.g. the service *creating an order with asset account assignment* can only be processed after having executed the service *create master record* (see Fig. 4). Thus a partial description of components on *coordination level* is possible. Whereas a complete specification is not possible, given that only example process variants are modeled and not all possible process flows.

Pre- and post-conditions can be gained from the business process models on *behavioral level*. The derivation of invariants from the process models instead is not possible.

According to the functional view, information about service description on *interface level* can be gained from the functions modeled in the process diagram. Whereas detailed information about data types, variables, constants etc. is not available through the functions and their processes in the process model.

5. Summary

Precondition for component-based development of business applications is a stable component model. In order to develop such a component model the use of a clearly defined process is essential. In chapter 2 the BCM process with its phases, sub-phases and corresponding tasks was introduced (see table 1).

Through modeling the functional-, data- and process-view, reference models provide information to be gathered otherwise through performing the tasks in the sub-phase *domain scope*. For the following sub-phase, the *business component identification phase (BCI),* reference models only partially contribute to the identification of business components. Reason is that common reference models [16, 17] do not use extended versions of event driven process chains for modeling their business processes. Important information gets lost in not allocating information objects to business functions. The allocation of information objects to business functions is not only necessary for the identification, but also for the specification of business components. *Specification of business objects* takes place in the sub-phase following the business components identification sub-phase (see table 1). Some reference models (e.g. [17]) use information flow diagrams, allocating information objects to business functions. The problem hereby is that the allocation is not as detailed as required for the identification and specification.

The impact of reference models for the specification of business components has been discussed in detail in chapter 3. The results are summarized in Tab. 3 and will be illustrated in the following.

Specification levels for business components	***Information to be used for the business components specification gained from the reference model***	***Information gained from the different diagrams***	***Property accounting example***	***Missing information needed for the specification of business components***
Marketing level	Information regarding functionality of a specific domain	Functional decomposition diagram	Domain: accounting, sector managerial accounting	Component version, scope of supply and component technology
	Information regarding the branch of economic activity	Functional decomposition diagram	Independent from the economic activity	
	Information regarding the name of the component	Functional decomposition diagram	Property accounting component	
Task level	Information about tasks and their decomposition in sub-tasks gained from functions and sub-functions out of the functional decomposition diagram. Additional tasks gained from the functions defined in the business process.	Functional decomposition diagram, Event driven process chains	Example task *accounting transaction* with related sub-tasks *acquisition posting, book depreciation;* Task *asset master record* gained from business process function.	Detailed task description
Terminology level	Technical terms and their relations optained for the specific domain.	Functional decomposition diagram, Entity-Relationship-Diagram, Event driven process chains	Terms like *accounting transaction, acquisition posting, book depreciation* etc.	Definition of the terms
Quality level	None			Quality properties classified into quality
Coordination level	Succession relationships visible between business process tasks. When mapping business tasks to component services, succession relationship between services is apparent.	Functional decomosition diagram	Example in natural language notation: *An order with asset account assignement can only be posted after having stored the asset master record.*	All variants of business processes required for component orchestration
Behavioral level	Information about potential invariants is received from the cardinality constraints of the data model. Pre- and postconditions for the business component services are gained from the business process.	Entity-Relationship-Diagram, Event driven process chains	Example in natural language notation: *At least one asset needs to be assigned to an assets account.*	Invariants related to one specific information object

Interface level	Relevant data types are gained from entity- and relationship-types through mapping technical terms to data types. Information about naming of business services is gained from mapping business tasks to component services.	Functional decomposition diagram, Entity-Relationship-Diagram, Event driven process chains	Example mapping of terms to data-types and business tasks to services:
			`interface property_accounting{`
			`  struct asset{...};`
			`  struct assets_account{...};`
			`  struct depreciation(...);`
			`  void book(depreciation a);`
			`  ....`
			`};`
				Detailed information needed for either identifying business component services, defining the relevant attributes, variables, constants, parameters and return values, or for the declaration of error messages

Tab. 3: Information needed for the specification of business components gained from the reference model

Regarding the specification framework, reference models contribute to the specification of business components on almost all levels of abstraction.

Information for describing the business domain and the branch of economic activity on marketing level is provided through functional decomposition diagrams. Technical data can not be gained from such diagrams. For specification on task level information is obtained from functional decomposition diagrams and event driven process chains. A detailed description of the tasks however is missing for a complete specification on task level. For the specification on terminology level, reference models provide limited information. Terms and relationships between terms are gained from all of the diagrams. Having the definition of those terms as important impact for the specification on terminology level, reference models – including the accompanying texts – do not provide information on that level of detail. The creation of a dictionary for a specific domain is not only necessary for the specification of business components, but is also of great importance for a better understanding of the reference models themselves. Having no information about the infrastructure given by reference models, little information is available for the description of business components on quality level. From the description of the process view succession relationships between tasks are gained in a limited form, having business processes describing only example processes and not the whole range of possible process variants. The decision about which process variant to use in the composition of business applications is the task of the system integrator and should not already be defined in the description of the business processes. From the succession relationship between tasks succession relationships between services can be mapped. The information about those relationships is needed for the specification of business components on coordination level. For the description on behavioral level, some invariants can be identified from relationships between entity-types. Invariants applying single information object can not be identified from the data model instead. Pre- and post-conditions related to services are obtained from the business processes and are used together with the invariants to specify the components on behavioral level. For the specification of components on interface level data-types can be mapped from terms gained through either entity-types or relationship-types. Information for the denomination of available services is gained through mapping business tasks to component services. Detailed information about attributes, variables, constants, parameters, return values and error messages can not be gained from reference models. Reason therefore is the missing assignment of information objects to business functions.

It can be summarized that reference models support the process of developing component based business applications and promote the reuse of business components in providing functionality of a specific domain for a wide range of users. The diagrams used by the reference models do not provide enough information needed for the identification and complete specification of business components. Additional information for the identification and complete specification of business components on interface level could be gained in using extended event driven process chains for modeling the business process level. Further important information, which has not been obtained from the reference model discussed, is needed for the definition of the available terms of a specific domain. This information is not only important for understanding the functionality of the business components, but also for better understanding of the reference models.

6 Conclusion

Regarding the similarities of existing reference models in the area of business information systems this article discusses which information can be gained, or is missing, from those reference models in order to support the modeling and specification of component based business applications. The goal was to support the process of developing business applications through information gained from the latest research in the area of reference modeling. Regarding the high degree in complexity of such business application systems the use of such reputable results is essential.

Looking at the similarities in the different reference models and at the typical modeling techniques used by those models, an example reference model has been chosen in order to illustrate the impact of reference models to the development of business component models. The example reference model uses the most common modeling techniques for the description of the different views – functional, data and process view. A functional decomposition diagram, an entity-relationship-diagram and an event driven process chain from the asset accounting domain [17] are used to illustrate the mapping of available information to artifacts of the specification framework for business components [7].

References

1. Szyperski, C., *Component software: beyond object-oriented programming*. 2 ed. 1998, Harlow: Addison-Wesley.
2. McIlroy, M.D. *Mass Produced Software Components*. In *Software Engineering: Report on a Conference by the NATO Science Committee*. 1968. Brussels: NATO Scientific Affairs Devision.
3. Hahn, H. and K. Turowski. *General Existence of Component Markets*. In *Sixteenth European meeting on Cybernetics and Systems Research (EMCSR)*. 2002. Vienna.
4. Turowski, K., ed. *Standardized Specification of Business Components: Memorandum of the working group 5.10.3 Component Oriented Business Application System*. 2002, University of Augsburg: Augsburg.
5. 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.
6. Albani, A., J.L.G. Dietz, and J.M. Zaha. *Identifying Business Components on the basis of an Enterprise Ontology*. In *Interop-Esa 2005 - First International Conference on Interoperability of Enterprise Software and Applications*. 2005. Geneva, Switzerland.
7. Turowski, K., ed. *Vereinheitlichte Spezifikation von Fachkomponenten*. 2002, Arbeitskreis 5.3.10 der Gesellschaft für Informatik: Augsburg.
8. OMG, ed. *The Common Object Request Broker: Architecture and Specification: Version 2.5, September 2001*. 2001, OMG: Framingham.
9. OMG, ed. *Unified Modeling Language Specification: Version 1.4, September 2001*. 2001, OMG: Needham.
10. Ortner, E., *Methodenneutraler Fachentwurf: Zu den Grundlagen einer anwendungsorientierten Informatik*. 1997, Stuttgart: Teubner.
11. Allen, P. and S. Frost, *Component-Based Development for Enterprise Systems: Applying The Select Perspective*. 1998, Cambridge: Cambridge University Press.

12. D'Souza, D.F. and A.C. Wills, *Objects, Components, and Frameworks with UML: The Catalysis Approach*. 1999, Reading: Addison-Wesley.
13. Sametinger, J., *Software engineering with reusable components*. 1997, Berlin ; New York: Springer. xvi, 272 p.
14. Ortner, E., *Ein Multipfad-Vorgehens-Modell für die Entwicklung von Informationssystemen - dargestellt am Beispiel von Workflow-Management-Anwendungen.* Wirtschaftsinformatik, 1998: p. 329 - 337.
15. IBM, *Business Systems Planning-Information Systems Planning Guide*, ed. I.D. (Ed.). 1984, Atlanta: International Business Machines.
16. Scheer, A.-W., *Wirtschaftsinformatik: Referenzmodelle für industrielle Geschäftsprozesse*. 7 ed. 1997, Berlin: Springer.
17. Becker, J. and R. Schütte, *Handelsinformationssysteme*. 1996, Landsberg.
18. Brocke, J.v., *Referenzmodellierung. Gestaltung und Verteilung von Konstruktionsprozessen*. Advances in Information Systems and Management Science. Vol. 4. 2003, Berlin: Logos.
19. Rosemann, M. and R. Schütte. *Grundsätze ordnungsmäßiger Referenzmodellierung*. In *Entwicklungsstand und Entwicklungsperspektiven der Referenzmodellierung*. 1997. Münster: Westfälische Wilhelms-Universität Münster.
20. Klein, S. and N. Szyperski, *Referenzmodell zum Electronic Commerce*. 1997.
21. Frank, U. *Entwurf eines Referenzmodells für Handelsplattformen im Internet*. In *Tagungsband der Fachtagung KnowTech*. 2000. Leipzig.
22. Albani, A., et al. *Komponentenmodell für die Strategische Lieferkettenentwicklung*. In *6. Internationale Tagung Wirtschaftsinformatik (WI-03) Medien - Märkte - Mobilität*. 2003. Dresden, Deutschland: Physica-Verlag.
23. Scheer, A.-W., *Architektur integrierter Informationssysteme*. 1991, Berlin: Springer.
24. Chen, P.P.-S., *The Entity-Relationship Model - Toward a Unified View of Data.* ACM Transactions on Database-Systems, 1976. **1**(1): p. S. 9-36.
25. Keller, G., M. Nüttgens, and A.-W. Scheer, *Semantische Prozeßmodellierung auf der Grundlage "Ereignisgesteuerter Prozeßketten (EPK)*. Veröffentlichungen des Instituts für Wirtschaftsinformatik, ed. A.-W. Scheer. Vol. 89. 1992, Saarbrücken.