A Foundation for a Model to use in Analysis of Documentation - a Systematic Framework to find the meeting between Software Component Specifications and Business Processes

Marie-Therese Christiansson, Benneth Christiansson
Institutionen för informationsteknologi
Informatik
651 88 Karlstad

Marie-Therese.Christiansson@kau.se
Benneth.Christiansson@kau.se

Abstract. Can software requirements for component-based systems be mapped to business processes? And in that case how can this mapping be viewed? The answer must be found in models and documentation approaches used by business- and systems developers. These models should take into account business knowledge, communication and logic to support and satisfy customer, partners and co-workers in systems which are developed. How is this done in integrated models that combine process modelling and requirements specifications or in pure process documents versus pure specifications? To find out, we need a model to use in analysis of documents. This meta-model should take notice of both theories in process- and component-based systems and approaches in practice to be useful. In this paper we present a foundation for such a model - a systematic framework to find the meeting between software components and business processes.

Introduction

Today, the need for well functioning and economically justifiable information systems is crystal clear. The quality of a company's information systems has become recognized as a strategic corporate advantage. Information systems and information technology can be viewed as the backbone of the modern enterprise and as such crucial to its supporting and providing new possibilities of running business operations. Communications patterns need to be clarified, assessed and developed, to achieve efficient operations, internally and in co-operation with others. Activities (tasks) in operations and their resource requirements in the form of co-workers, knowledge, machinery; flows of information, materials and payments; must be distributed and coordinated. Part of this work is to establish the roles and responsibilities needed to perform the task, and the necessary resources. Another part is to determine if the tasks should be supported or run by a software system, to efficiently provide

people with information that is to be used in support and control of operational actions [1][2][3].

The software industry is challenged by the new claims made by its customers. Sutherland [4] describes the situation today as: *"The global market has become an intensely competitive environment moving at an accelerating rate of change. To gain the strategic advantages of speed and flexibility, corporations must remodel their business processes, then rapidly translate that model into software implementations."* This means that the need to reduce time to market for software products is extremely crucial in today's software industry.

Schach [5] declares that *"Software is more complex than any other construct made by human beings. Even hardware is almost trivial compared to software*" This quotation illustrates a fact that we think most of the actors in the software industry agree upon, constructing software is a very complex task in itself. At the same speed as hardware capacity is increasing, the complexity of the software solutions is increasing. This means that it is only getting more difficult to create the software solutions that today's customer require, because of the increased complexity in the required solutions.

In the increasingly competitive software industry the need for new innovative techniques to deliver satisfying solutions is greater than ever. This may in some sense explain the great belief and adaptation of so-called 'Silver Bullets' in the software industry. A 'Silver Bullet' is a new technique or tool that is made out to be the great solution to the different problems that the software industry is facing [6]. The newest trend is considered something to adapt and use, new trends and tools are presented at an accelerating speed. Maybe the requirement for reduced time to market is not the only reason to this eagerness to adapt 'Silver Bullets' Peisl [7] identifies a list of reasons *"To be successful in today 's rapidly changing market, your business requires a solution that allows you to integrate your business processes across different vertical markets and various operating environments. Effective business integration enables disparate business resources—both inside and outside an enterprise—to work together to support a company 's business strategy. A business process management (BPM) solution is the collection of instruments with which you can model your enterprise business processes, perform activity-based costing, simulate the processes and deploy them."*

The eagerness to adapt 'Silver Bullets' has created a golden opportunity for people with good ideas to make big money coming up with new innovative techniques for software organizations to increase their competitiveness. We argue, in general, for a more well thought out and stable foundation for developing and using component based information systems. In this paper we focus in particular on the hypothesis that well functioning component-based information systems should be founded in business processes. In this paper we have chosen to focus on a foundation for a model to use in analysis of documents - a systematic framework to find the meeting between software components and business processes. To be able to justify and argue for this need we will define and discuss important concepts and terms that we use in association with component- and process-based systems development.

1 Component-based information systems

One of the more recent techniques in the software development industry is component-based software development. This means that information systems are created through assembling more or less standardized software components into a unique software solution. *"The concept of component software represent a middle path that could solve this problem. Although each bought component is a standardized product, with all the advantages that brings, the process of component assembly allows the opportunity for significant customization."* [8] We regard this statement as an ideal-situation or a theoretical standpoint, in reality a software development project should use whatever way to get the job done. This may mean an application package with a few added components or such.

The notion of software components is not new. Originally it was presented as early as 1968 on the NATO Conference on Software Engineering in a paper called 'Mass-Produced Software Components' by Douglas Mcllroy [9]. This paper is today famous and often referred to in component-based software development literature. He proposed a software industry that supplied off-the-shelf standard software components. He envisioned that programmers would combine these software components, instead of creating software solutions from scratch they would assemble software solutions from existing parts. We suspect that one of the reasons to why Douglas Mcllroy's vision isn't realized yet, is that each technique presented often only focus on the formal- or informal part of an information system. The predominance is on techniques that focus on the formal part. Some techniques focus on the informal part, they are very rare though and no techniques at all focus on both parts. Sims [10] illustrates this in what he describes as 'Perpetuating the Great Mistake'.

The term software component isn't easy to define, it does not have a clear-cut definition in the software development community, but the meaning fluctuates. This paper does not focus on the issue of defining the term software component even though this is something needed and hopefully soon to be done. Instead we support the definition that Christiansson [11] makes. This definition is based on a survey and symbiosis of several more or less well-established definitions. We start our discussion concerning the meaning of the term with two quotations. *"A component is a unit of software of precise purpose sold to the application developer community…with the primary benefit of eliminating a majority of the programming the byer most perform to develop one or more function points…"* [12]
"A component is a reusable piece of software in binary form that can be plugged into other components from other vendors with relatively little effort" [13]
These definitions, we believe, illustrates several of the more basic criteria such as a software components binary shape and its ability to connect without reconstruction to other software components. However these definitions does not include issues, such as identifying, maintaining and refining software components. Therefore we suggest a more mature definition of the concept software component. In this paper we will use a definition made by Christiansson [11] *" A software component:*

- *is independent and reusable;*
- *provides a defined functionality using a specific interface;*
- *can affect/be affected by other software components;*
- *should have a specification (in which the software component is described on a high level of abstraction);*
- *can have multiple implementations, meaning that the same component can be implemented in several programming languages; and*

- *can have several executable (binary) forms, i.e. the same component can be executed in different operating systems".*

The fact that a component is independent and reusable shows that a component can be used without other components present, the services provided by the component should be accessible without any external help except from the software glue and necessary run-time environment. A component can affect and be affected by other software components. This means that two components can 'work together' and 'as a whole' creates a greater service than used separately. In figure 1 we illustrate a software component with a context.

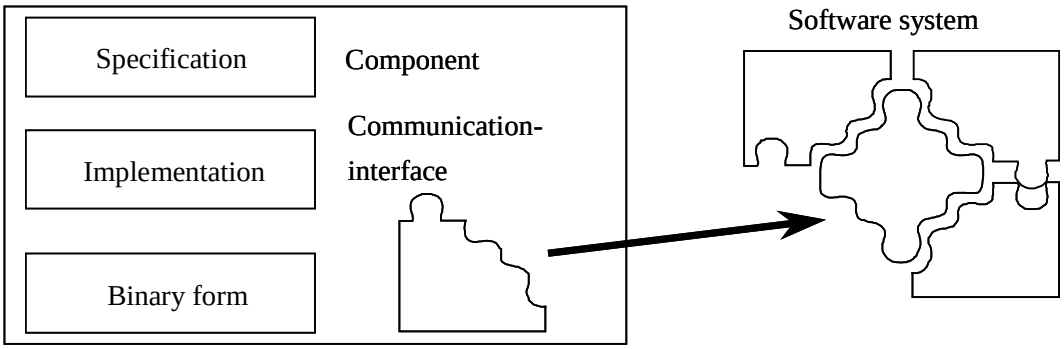

Figure 1. A software component with a context.

The need for a documented specification for a software component is obvious if one consider the process of acquiring a component. How can one find a software component if one doesn't have something to look for. This is maybe the one single factor that can decrease the gap between the formal- and informal-part when developing a information system. If there are documented specifications, these can be described in such ways that they are useful when dealing with the development of the informal part of a information system and then the component as such is directly applicable in the formal part [11]. This notion of documented specifications can be elaborated to incorporate the need for standardization concerning specifying software components. Vaughn [9] implies that *"A standard approach to building and using components must be set and universally practiced if the software engineering community is to reap the benefits…of reusable software components…"* . With this quotation we want to stress the fact that standardization of software component specifications will be one of the next issues for solving the fundamental problem with software development [11].

As described in previous section we identify a need to structure and preferably standardize the way we describe components to for instance enable acquisition. Data, functional, and behavioral models (represented in a variety of different notations) can be created to describe what a particular component or application must accomplish. Written specifications can then be used to describe these components or applications. A complete description of requirements is the result. Ideally, the requirements in the form of specifications are analyzed to determine those elements of the model that point to existing reusable components. The problem is extracting information from the requirements in a form that can lead to 'specification matching' i.e. The possibility to identify existing components from the documented requirements. Components should be defined and stored in three different states specifications, implementations and binary executables. Ideal these components are an engineered description of a product from previous applications. The specifications could be stored in the form of reuse-suggestions, which should contain directions for retrieving reusable components on the basis of their description and for composing and tailoring them

after retrieval. Bellinzona et. al [14] describes one very formal but possible way of doing this. Another way is what Tracz [15] has called the 3C Model-concept, content, and context. The concept of a software component is a description of what the component does [16]. The interface to the component is fully described and the semantics represented within the context of pre- and post-conditions-is identified. The concept should communicate the intent of the component. The content of a component describes how the concept is realized. In essence, the content is information that is hidden from casual users and need be known only to those who intend to modify the component. We summarize this discussion in a term-graph that illustrates the relations between the central aspects regarding component-based systems (figure 2), this illustration will be a part of the theoretical foundation for a systematic framework to find the meeting between software component specifications and business processes.

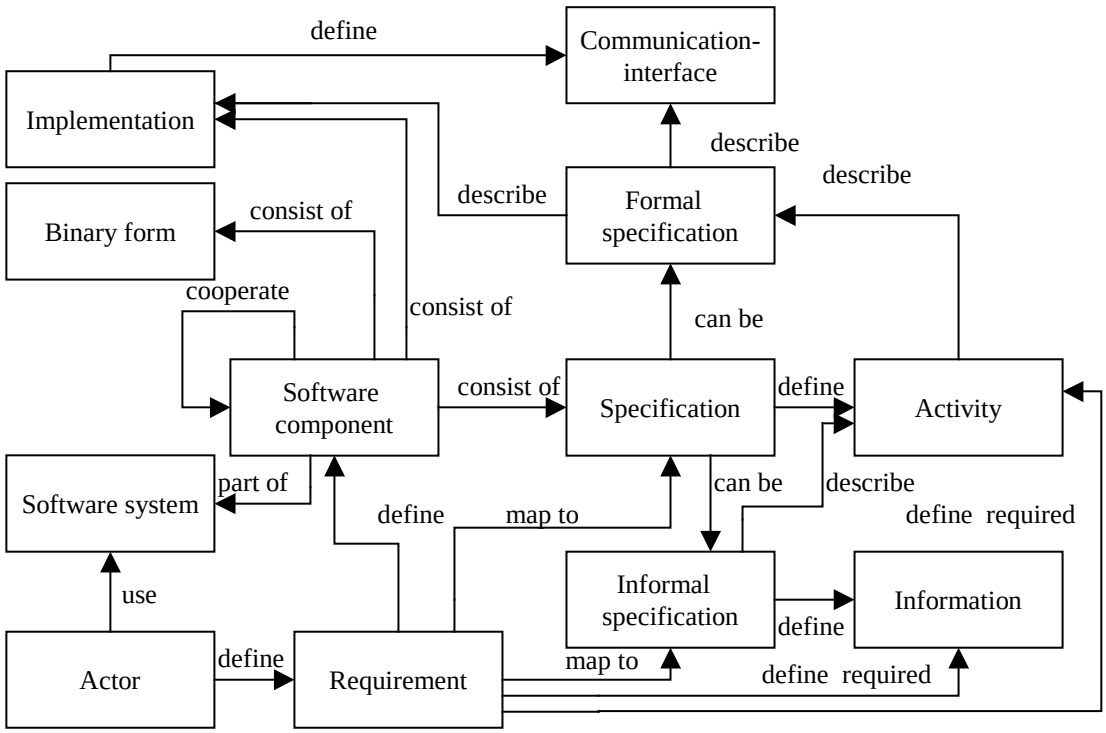

Figure 2. Component-based information systems.

2 Process-based information systems

Business development may be initiated for a number of different motives. Examples of such motives in industry are deficient communication between companies; ineffective and unreliable flows of material and information; too much hassle in the daily work; and in aiming to increase profitability from the co-operation with other parties [17]. Other motives includes generating long-term planning and flexibility, creating a learning organization and improving the ways of working by a supporting information systems, to provide a valuable result to those for whom the business operations are intended [18]. A process-based business-/systems development means to focus on business activities from the viewpoint of those who will make use of their results, and on how activities will add value to the clients.

A boundary-crossing quality is one of several features inherent in processes. Other features are that processes should [19][20]:

- have a beginning and an end;
- consist of structured, coherent activities that are repetitive and consume resources of different kinds;
- contribute to fulfil the business goals; and
- help to create value for external customers/clients.

The above criteria are evident to a greater or less degree, i.e. relatively difficult to identify and determine, depending on type of business operation. One aim of process orientation is to elucidate the "*white space on the organisation chart*" [21]. This implies to elucidate connections, handovers and the responsibility for what happens between business departments. One way to gain this general view is to study the flow of activities in organisations horizontally, focusing on 'what, how and for whom' an activity is performed. In other words, to study operations as processes. The horizontal flow of value added activities, its execution and need of resources is studied from an overall perspective, from the point of view of the client/s of the process results. This leads to different processes being identified, as the operations are run in different ways depending on who the beneficiaries of the results are. The co-workers are in focus, as they know which activities are done, and how. Their knowledge is essential to be able to identify and describe how business operations are run, or could and should be run. Process orientation is therefore built on, and at the same time creates, commitment and participation in the business development. It is more difficult to attain an overall perspective by studying activities vertically in business operations, i.e. as functions focusing on 'what is done and where'. The attention is then centred on how each function can be improved separately, whereby the connection between them may be lost. There is a risk that some functions will be improved and developed 'for their own sake', and not with the clients as a point of departure. One perspective should not exclude the other, however. It may be necessary to study operations in a business development both vertically and horizontally.

Business modelling is the work of reconstructing and describing how operations have been run, are being run, may or should be run to achieve a basis for observing and comprehending operations [22][23][24]. Business models aim to communicate different phenomena in organisations, i.e. to identify and bring up matters for discussion. The purpose is not to describe the 'truth', as this is a highly subjective quantity, but rather to provide a basis for discussions, by which people can communicate differing viewpoints to arrive at an intersubjective view.

Willars [25] points out that it is impossible to have fully objective and comprehensive models. An obvious strategy would be to choose business models that correspond to the motives of the business development, reflecting the desired knowledge about organisations. Successful results of business development depend on grounded decision bases. This means that change requirements can be identified on the basis of how the business 'is doing at present', and that relevant improvements measures can be formulated [26]. An example of grounded decision bases is business models describing business processes, experienced problems, strengths and goals in these processes, and showing clear connections between each model, to achieve traceability across the descriptions. The motives for improvement measures should be reflected in the drawn up business models. The arguments should be found in the business processes, experienced problems, strengths and goals. If important information to learn about operations is missing, time has been spent in vain to develop models. This will become critical when following-up a business development. As we know, we need to determine a 'present situation' from which to follow-up how change measures have improved operations. Phenomena that has not been mapped cannot be followed-up.

Business models are worked out in development work founded on two basic needs, namely to *"… try to understand and change"* business operations in organizations [24]. Understanding and changing organizations signifies that the starting point should be what is actually carried out in these. Taylor [27] underlines this as follows: *"… communication (and therefore organization) is grounded in action, not in information transmission, nor even in the transfer of knowledge."* In other words, the focus should be on the way we carry out tasks and on the communication pursued as prerequisites for, and results of, these activities. Business models describing what is done by whom, how, in which order and for whom afford knowledge of how business operations work and how they should be designed as regards information and knowledge dissemination.

If knowledge of the actual conditions is missing, we can only guess how the business operations are run in practice. Modeling business operations is a basis for identifying, analyzing, assessing, developing and following up organizations that are considerably better grounded. Business development may refer to strategic development, process development or systems development. Improvements in these various development areas should harmonies to make the overall organization vital and competitive [28]. A number of business models are set up in every development area. Each of these areas could, in one and the same business operation or in co-operative business, initiate different development projects that run parallel and, to a varying degree, in inter-play. Our point is that the intentions that exist in other development areas are materialized by the business processes. Therefore it is essential to initiate software component development in the existing business processes. It is also vital to map models describing the business processes to the models used in information systems development. Or at least gather relevant knowledge regarding the business processes and required components in an integrated model.

Depending on motive, the different phenomena in an organization will be of varying importance when considering a mapping of processes. There is a difference between studying processes focusing on learning and studying processes focusing on system development. Different motives may, of course, coincide. To map processes means to identify and describe them. Not until business operations are described in terms of processes do we see and become aware of processes. They are 'invisible' up to the moment when they are defined and discussed. To guide the attention to phenomena in business operations, different types of business process models may be worked out, such as graphical illustrations (diagrams), tables and lists. Irrespective of the descriptive model used, these should support communication, to attain knowledge about operations, so that we can analyze, assess, develop and follow up these. Obviously, it is not necessary to describe and understand all the processes in the organizations. Processes may be studied with different frame works, which implies that different filters help select which business phenomena to use, and how to focus these, or what should be excluded in an analysis [18]. As examples of frame works we can mention computer support, skills, care, learning and various priorities in the operations that are run. The framework should be chosen against the background of the motives that are involved in the business development, and the issues that are of importance to follow-up after an improvement.

Depending on type of organization and on who the beneficiaries are, processes may be considered as core or support processes [29]. Core processes should provide a flow-oriented total picture of the main assignments of the organization, to fulfill the business mission and to create a focus on how value be added for those who are to utilize the results of the process. Activities run in core processes are supposed to place strategic issues in focus and help bring the organization closer to these visions. Activities in support processes constitute

a necessary backing to perform successful work in the core processes. In industry several different types of customers can be identified for one and the same process, which makes different demands on its execution, i.e. different variant processes. Suppliers, owners and public administration are other beneficiaries of process results in the form of products, services and information. Processes can to a large extent be run with the same activities and resources, but the phenomena considered interesting for further study may vary depending on the beneficiary. In the health care sector this may be exemplified by the fact that an analysis by one and the same process can vary depending on whether it is patients, relatives or citizens who are the beneficiaries. When the beneficiary has been identified, the beginning and end of a process can be determined, based on the motives behind the business development.

Motives for using a process-based systems development, from the viewpoint of supporting and enabling information systems, can be summarized as follow:

- to be able to base decisions on actual conditions, i.e. from what is actually performed;
- to find motives for systems development;
- to gain a quality assurance by an increased orientation towards those who utilise the results;
- to create a basis for being able to follow-up the results of a systems development in organisations in a relevant way;
- to strengthen the exchange of experiences between co-workers and parties;
- to improve communications between co-workers and parties;
- to create commitment and participation among co-workers by forming a general picture of how organisations are fulfilling their assignments and how the work of the co-workers is contributing to this end; and
- to take advantage of the full potential of the co-workers and information systems.

Our theoretical foundation of central aspects and their relations regarding business process are illustrated in figure 3.

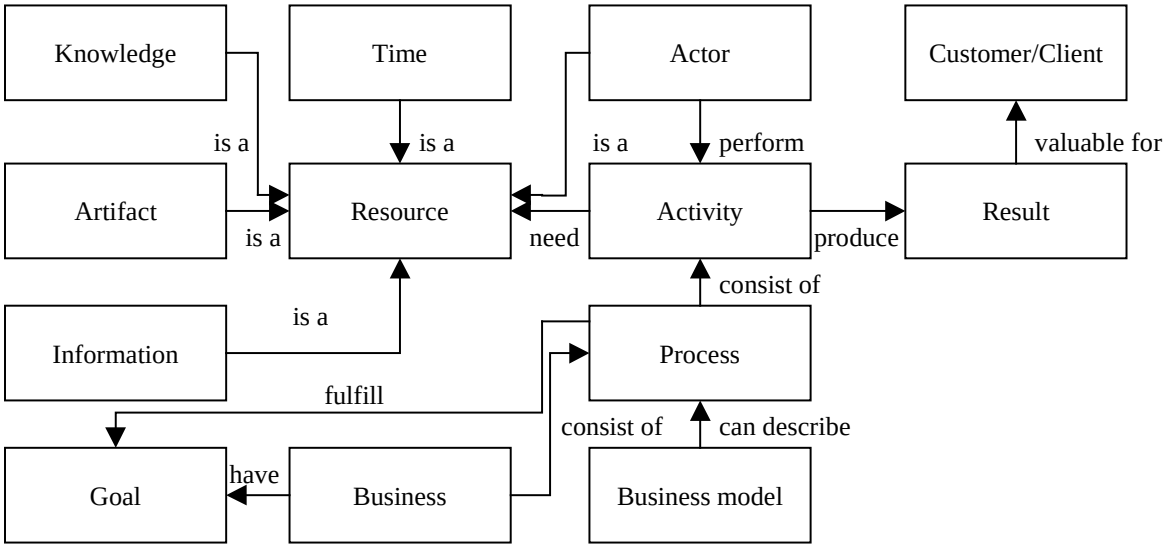

Figure 3. Business process

3 A foundation for a model to use in analysis of documentation

We identify three terms that exists in both term graphs (figure 2 and figure 3) connections between the areas of component- and process-based business systems. The terms are actor, activity and information. These connections indicate areas of overlapping in the different perspectives regarding component- based business systems on one hand and process-based business systems on the other. To explore these possible overlapping in perspectives further we define these terms from each of the perspectives see table 1.

Table 1. Similar terms and their definitions.

Term	Component perspective	Process perspective
Actor	software component user (artefact, human)	person or artefact who perform activities
Activity	component behaviour and software functionality	tasks in business operations
Information	the interpretation of data processed by software components, interpreted by actor	signals (messages) to be used for support and control of business operations

Regarding the term actor there is a mutual meaning in the different perspective. They both identify actor as human or artefact and the importance is action. In the process perspective the term is used in a wider sense than in component perspective, activity per say is not bound to software usage or software-component usage. The definitions are overlapping but the scope is wider with process perspective.

Activity from a component perspective focuses on component behaviour or functionality, activity is what components does. Again this is a more narrow definition than in process perspectives where activity can be any tasks in a business environment. Another issue is granularity or abstraction level on activity. In component perspective activity can be a single operation for a single component which indicates a low-level of abstraction, this is necessary to be able to handle specific software components and their functionality sufficiently. Activity in a process perspective can be very high-level of abstraction where one activity in a process-model can mean thousands of activities in a component-model.

Information is the interpreted data processed by software components in a component-perspective, the definition is near the implemented data in software systems again we identify the difference in granularity. In a process-perspective information is more generally defined. Information can of course originate from software but does not have to, information are signals that are used to control business operations.

4 Conclusions

In this paper we have chosen to focus on a foundation for a model to use in analysis of documentation - a systematic framework to find the meeting between software component specifications and business processes. We have described the basic theory regarding software component specifications and business processes. Each of these areas have been

described in a term-graph. These term-graphs have then functioned as a point of origin for identifying a theoretical foundation for a systematic framework to find the meeting between software component specifications and business processes. This foundation is illustrated in table 1. In our future research we will use this foundation as a starting point to generate our framework with focus on actor, activity and information.

References

[1] Langefors, B. (1973) Theoretical Analysis of Information Systems, 4:e upplagan, Studentlitteratur, Lund.

[2] Avison, D. E. & Fitzgerald, G. (1988) Information Systems Development, Methodologies, Techniques and Tools, Blackwell Scientific Publications, Oxford.

[3] Goldkuhl, G. (1992) Från HUMOR till VITS - Humaninfologi anno 1992, Institutionen för datavetenskap, Linköpings universitet.

[4] Sutherland J. (1996) *The Object Technology architecture: Business Objects for Corporate Information Systems.* OOPSLA'95 Workshop on Business Object Design and Implementation, Springer-Verlag, Berlin.

[5] Schach S. R. (1997) *Software Engineering with Java*. McGraw-Hill Companies Inc. New York USA.

[6] Goldberg A. & Rubin K S. (1995*) Succeeding with Objects, Decision Frameworks for Project Management.* Addison-Wesley Publishing Company, California, Menlo Park.

[7] Roland Peisl (2002) *Delivering business agility and service level management with business process integration*. WebSphere Business Integration Marketing IBM Software Group, p 2.

[8] Szyperski C. (1997). *Component Software Beyond Object-Oriented Programming*. Addison Wesley Longman, Inc., California, Menlo Park.

[9] Vaughn T. (1990) *Issues in Standards for Reusable Software Components*. (november, 1996) http://ruff.cs.umbc.edu.

[10] Sims O. (1996) *Perpetuating the Great Mistake*. http://www.sigs.com.

[11] Christiansson, B. (2001) *"Component-Based Systems development*". In: Nilsson A.G. & Pettersson, J.S. (eds.).*On Methods for Systems Development in Professional Organisations*, Studenlitteratur, Lund

[12] Steel J. (1996). *Component Technology Part I An IDC White Paper.* International Data Corporation, UK, London.

[13] Microsoft. (1996) *The Microsoft Object Technology Strategy: Component Software.* www.microsoft.com.

[14] Bellinzona R., Gugini M. G. & Pernici B. (1995) *Reusing Specifications in OO Applications.* IEEE Software, March pp. 65-75.

[15] Tracz W. (1990). *Where does reuse start?* Proc Realities of Reuse Workshop, Syracuse University CASE Center, Jan.

[16] Whittle B. (1995). Models and Languages for Component Description and Reuse. ACM Software Engineering Notes vol. 20, no. 2 April, pp. 21-22.

[17] Christiansson, M.-T. (1998) Inter-organisatorisk verksamhetsutveckling – metoder som stöd vid utveckling av partnerskap och informationssystem, Licentiatavhandling, Institutionen för datavetenskap, Linköpings universitet.

[18] Christiansson, M-T. (2001) Process orientation in inter-organisational co-operation - by which strategy? in Nilsson, A. G. & Pettersson, J. S. (eds.) On Methods for Systems Development in Professional Organisations - The Karlstad University Approach to Information Systems and its Role in Society, Studentlitteratur, Lund.

[19] Davenport, T. H. (1993) Process Innovation – Reengineering Work through Information Technology, Harvard Business School Press, Boston.

[20] Goldkuhl, G. (1995) Processtänkande vid verksamhetsutveckling, Institutionen för datavetenskap, Linköpings universitet.

[21] Rummler G. A. & Brache, A. P. (1995) Improving Performance – How to Manage the White Space on the Organization Chart, Jossey-Bass Publishers, San Francisco.

[22] Goldkuhl, G. (1991) Stöd och Struktur i Systemutvecklingsprocessen, Institutionen för datavetenskap, Linköpings universitet.

[23] Jayaratna, N. (1994) Understanding and Evaluating Methodologies – NIMSAD: A Systematic Framework, McGraw-Hill, London.

[24] Tolis, C. & Nilsson, A. G. (1996) Using Business Models in Process Orientation, in Lundeberg, M. Sundgren B. (eds.). Advancing Your Business – People and Information Systems in Concert, Studentlitteratur, Lund, Chapter VIII, www.hhs.se/im/efi/ayb.htm.

[25] Willars, H. (1993) Modelleringsledarens bashandledning – En inledning till hur man kan förstå, leda och dra nytta av verksamhetsanalys med modellering, Triad-rapport, No. 10:2, SISU, Stockholm.

[26] Goldkuhl, G. & Röstlinger, A. (1988) Förändringsanalys – Arbetsmetodik och förhållningssätt för goda förändringsbeslut, Studentlitteratur, Lund.

[27] Taylor, J. R. (1993) Rethinking the Theory of Organizational Communication: How to Read an Organization, Ablex Publishing, Norwood, New Jersey, p. 112.

[28] Österle, H. (1995) Business in the Information Age – Heading for New Processes, Springer-Verlag, Berlin.

[29] Rentzhog, O. (1998) Processorientering – En grund för morgondagens organisationer, Studentlitteratur, Lund.