# An Integrated Component-Oriented Framework for Effective and Flexible Enterprise Distributed Systems Development

**Zoran Stojanovic**
Systems Engineering Group
Faculty of Technology, Policy and Management
Delft University of Technology
Jaffalaan 5, 2628BX, Delft, The Netherlands
Z.Stojanovic@tbm.tudelft.nl

## Abstract

Although component-based platforms and technologies such as CORBA, COM+/.NET and Enterprise Java Beans (EJB) are now widely used for implementation and deployment of complex systems, the component way of thinking is still immature. Current CBD best practices, concepts, approaches and methods do not provide a full and consistent support for various component concepts, and therefore are not able to provide a full benefit of the CBD paradigm. This paper defines a new approach to components through an Integrated Component-Oriented Framework that provides a comprehensive component-oriented support for enterprise systems development. The framework enables that the same component way of thinking and the same consistent set of technology-independent component concepts can be applied in different aspects of enterprise systems development, from business services to distributed components.

## 1. Introduction

Intensive use of advanced information and communication technology (ICT) is nowadays considered as an integral part of modern enterprises' business strategy, significantly impacting the way they perform business services and compete on the market. The main challenge enterprises face today is how to manage the complexity inherent in the systems they are deploying, while at the same time to be able to rapidly adapt to changes in technology and business environments. The solution by many lies in growing interest in the research community and industry over Component-Based Development (CBD) [Szyperski, 1998; Brown and Wallnau, 1998; Gartner Group, 1997; Butler Group, 1998]. CBD provides organizations with a method for building complex enterprise-scale solutions that are flexible and able to accommodate the ever-changing demands in the environment, in a cost-effective, timely manner. By using the CBD approach, a system development becomes the selection, reconfiguration, adaptation, assembling and deployment of encapsulated, replaceable and reusable, functional elements called components, rather than building the whole system from scratch. Although CBD has been introduced by many as a new silver bullet for complex, enterprise-scale system development in the Internet age, it is rather evolutionary than revolutionary approach. CBD inherits many concepts and ideas from the earlier encapsulation and modularization, "divide-and-conquer" initiatives in the computer science, such as module programming and object-orientation.

CBD provides higher productivity in system development and reduced cost and time-to-market by providing component reusability, more effective system maintenance, higher product quality and possibility for parallel work. Moreover, it provides a higher level of system flexibility and adaptability through component replaceability, localization of changes, better solution scalability, and possibility for easy integration of legacy assets.

To date, the CBD paradigm has impacted mainly the implementation and technology level, providing pieces of system's functionality to be deployed across the network nodes. Several distributed infrastructure technologies based on Microsoft's Component Object Model (COM) and .NET [Microsoft], Object Management Group's (OMG) Common Object Request Broker Architecture (CORBA) [Siegel, 2000] or Java-based tools [Sun Microsystems] are now actually standards in the development of complex Internet-enabled systems. While the technology is a necessary element of any solution, it is not sufficient by its own. Equally important for the effective use of the CBD paradigm and for gaining real benefits of it are the component-oriented way of thinking, component modeling and notation, component-oriented development process, strategy and tools, before one actually starts using technology. Even if Commercial-Off-The-Shelf (COTS) components are used in building a system, still a proper methodological guidance is required in order to achieve a successful solution. Academia and industry have just started recognizing the importance of new CBD methods and techniques, by introducing their approaches, best practices and experiences in following the CBD paradigm [Brown and Wallnau, 1998]. Approaches and methods proposed so far do not provide a complete and consistent support for various component concepts. The first-generation component-based approaches, such as Rational Unified Process [Jacobson et al., 1999], Catalysis [D'Souza and Wills, 1999] and Select Perspective [Allen and Frost, 1998], are characterized by a common use of the standard Unified Modeling Language (UML) [Booch et al., 1999] and object-oriented (OO) constructs and concepts in building component-based systems. As suggested by the UML, components are treated as implementation concepts - packages of binary or source code that can be deployed over the network nodes, and used at the implementation and deployment phases of the development lifecycle. On the other hand classical object-oriented way of thinking, modeling, analysis and design are followed. Regarding that, the main difference between objects and components is in a physical granularity [Henderson-Sellers et al., 1999], instead in completely different philosophy and levels of abstraction [Stojanovic and Dahanayake, 2002]. The questions of how to identify, model and specify components, how to follow a component-based development process in a systematic and consistent manner, and how to assembly formally specified components into the component-based system architecture are not properly addressed yet.

In order to get full benefits from the component way of thinking and CBD, a completely new approach to components and a comprehensive CBD strategy are required. The approach should provide that the component-based paradigm and the main component concepts are applied consistently and systematically from business to technology. In that way the full benefit of the component way of thinking in business-driven system development can be achieved. Furthermore, the service-based component concept can represent a point of convergence between business and technical concerns.

## 2. Research Goal and Questions

The main goal of the research is to define an Integrated Component-Oriented Framework (ICOF) that provides a model-driven development process support for building Internet-enabled enterprise systems, from business processes to implementation. The ICOF should provide that the same component way of thinking and the same consistent set of component concepts can be effectively applied in different aspects and phases of enterprise systems development, from autonomous business services to distributed software components. The main idea behind is that a component-based solution for a business problem can be provided through the set of interrelated models. Each model represents the system from a particular viewpoint: enterprise, information, system architecture, and distribution. The overall component-oriented solution for a given problem is obtained through the integration of fully specified models organized around the unifying component concepts. After the complete specification of a system is provided, the choice is to build components, buy COTS components, wrap existing legacy assets, or invoke web services, and then assembly them into the complete solution. Based on the main research goal the following research questions can be formulated:

1. How to define an integrated component-oriented framework providing comprehensive, integrated and systematic support for various aspects of enterprise systems development?
   1.1 How to define consistent, technology-independent component concepts?
   1.2 How to define different, but mutually related levels of component granularity?
   1.3 How to define traceability from business processes to implementation artifacts using components?
   1.4 How to define proper component modeling and specification notations?
   1.5 How to define a component-oriented development process?
   1.6 How to define a useful separation of concerns to manage complexity in system development?
2. How to provide flexibility and tailorability of the proposed component-oriented framework?
   2.1 How to use a model-driven approach to provide technology-independency and interoperability of the framework?
   2.2 How to use the framework in different application domains where different entities, services, types of information and requirements are defined?
   2.3 How to specify the framework to allow component providing through component building in-house, buying COTS components, wrapping existing assets by component interface or invoking web services over the Internet?

3. How to validate and evaluate the framework using the real application cases?

## 3. Research Approach and Planning

A range of research approaches has been recommended for the use in the general field of information systems, among them the most important are literature survey, case study, field experiment, laboratory experiment, surveys and simulation [Galliers, 1994]. The selection of an appropriate methodology requires the evaluation of many factors and the determination of how well they work together in supporting the research objective. The research approach for this project is based mainly on the literature survey and case studies. Literature survey is considered to be an important (and the basic) part of every research in order to find out the state-of-the-art of the research area and for future comparison of research contributions with current research achievements in that area. Case study should help in capturing the knowledge of practitioners and developing theory of it. It gives the possibility to generate theories from practice and allows understanding the nature and complexity of the processes in the problem area [Yin, 1994].

Considering defined research objective and questions, the following research plan can be defined. The research consists of the pre-phase related to a research proposal, three main phases related to corresponding three research questions and the final phase dedicated to the thesis writing. In the pre-phase, literature survey was used for defining the state-of-the-art as well as problems in the chosen research field. Research aims and questions were specified and the research proposal was written. The first phase should provide an answer to the first research question and sub-questions. After this phase, a component-oriented framework and its main elements should be defined. A consistent set of technology-independent component concepts and different component granularity levels should be defined. The framework and proposed concepts should provide a component-oriented development process from business services to implementation assets using a separation of concerns to manage process and system complexity. The second phase should answer the second research question, by providing flexibility and tailorability of the framework. The framework and all its concepts must be model-driven and technology-independent to provide a higher-level support for system development that can end in any technology infrastructure. The framework should be applicable in different application domains and should allow building the system from components that are built, bought, made by wrapping legacy assets or invoked as web services across the Internet. The third phase should answer the third research question and prove the applicability of the framework in real application cases. This phase should end with a prototype of a system which development is supported by the framework from business requirements to realization. The final phase includes the practical work on formulating, collecting and structuring the results of the research phases and research project as a whole, in order to form a doctoral thesis, as a final result of the project work.

The whole period of the research should last four years. Each year is divided into halves, consisting of 6 months each. Table 1 gives an approximate time schedule of the project. After each phase and/or subphase of the research project, based on the current achievements and results, technical reports, and conference, workshop and journal papers have been, are being and will be prepared and published. It must be noted that the research is now on the half, which means that two years are completed by now. The pre-phase was successfully completed during the first year of the research, and the first phase should be completed till the end of the year. The second phase has been started, and now it is performed together with the first one. The preparations for the third phase are currently being made.

| Phases of the project | $1^{st}$ year | | $2^{nd}$ year | | $3^{rd}$ year | | $4^{th}$ year | |
|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 1 | 2 | 1 | 2 | 1 | 2 |
| Pre-phase | X | X | | | | | | |
| First phase | | X | X | X | X | | | |
| Second phase | | | | X | X | X | | |
| Third phase | | | | | X | X | X | |
| Final phase | | | | | | | X | X |

Table 1. Research planning

Research approach consists mainly of the literature survey and performing case studies. It represents a combination of inductive-exploratory and deductive research strategy. The inductive-exploratory research strategy has been chosen for the first phase of the research related to formulating the theory about the component-oriented framework. It consists of extensive literature study about the state-of-the-art of the considered research area to formulate a basic theory, and proper inductive case studies for refining the theory, adding new concepts and getting new ideas. Proper case studies should help first in formulating and establishing, and then proving the framework and the concepts and theory behind it. Combination of inductive and deductive case studies will be used in the second phase of the research. Variety and expressiveness of these case studies should help in providing tailorability and flexibility of proposed framework concepts and their applicability in different domains and environments. Finally an evaluation case study will be chosen to help in validating and evaluating the component-oriented framework in the real settings. That case study should result in a prototype of the system as the main proof of the proposed theory. The particular research approach for each of the phases of the research is shown in Table 2.

| **Research phase** | **Research approach** |
|---|---|
| Pre-phase | Literature survey |
| First phase | Literature survey + Inductive case studies |
| Second phase | Inductive + deductive case studies |
| Third phase | Evaluation case study |

Table 2. Research approach through the research phases.

Potential case studies for the research are in the fields of Geographic Information System (GIS), e-business, and e-government with possible extensions into the fields of location-based services and mobile business. One of the reasons for this choice is author's background in GIS. Furthermore, the study "GIS in business" conducted by Dutch-based Ravi Business Platform in collaboration with the Free University of Amsterdam and Britain's Manchester Metropolitan University shows that an estimated 90% of business-related information is geographical in nature, especially in business support systems [GeoEurope, 2000]. Therefore an integration of GIS, e-business and e-government currently represents the hot topic and the main direction of developments in ICT. Supporting the range and variety of concepts and infrastructures defined in these domains by the component-oriented framework represent a real challenge for this research.

## 4. Current State of the Research

The main research objective is to define an Integrated Component-Oriented Framework for supporting all phases and aspects of Internet-enabled systems development, from business to implementation. The main elements of the framework are presented in the Figure 1.

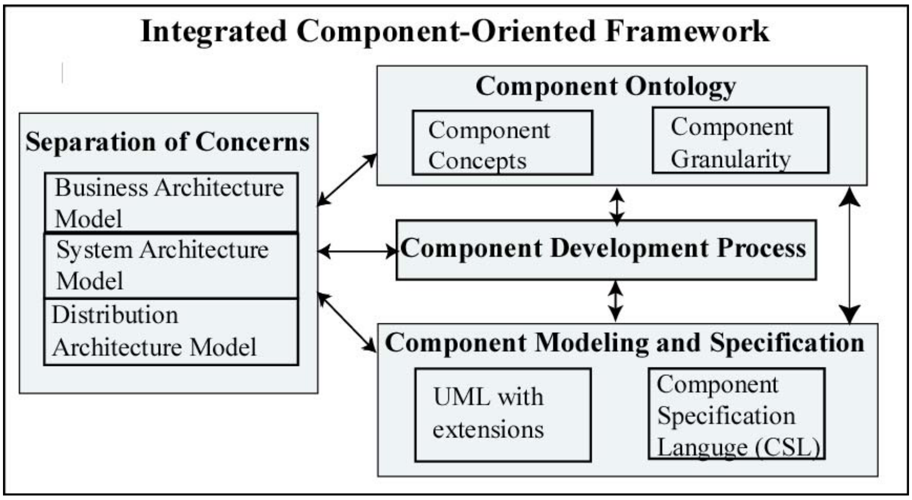


Figure 1. Integrated Component-Oriented Framework.

The framework actually specifies the way of thinking, the way of modeling and the way of working of a component oriented development methodology, as defined by Sol [Sol, 1988]. The framework should be actively supported by appropriate component-oriented tools, and controlled by appropriate management support, specifying the way of supporting and the way of controlling respectively.

As the first step of the research, a general, technology independent definition of a component has been proposed. By the definition, a component is an encapsulated, autonomous service provider that delivers useful services through well-specified interfaces to the wider context inside which it exists and collaborates with other components towards a common goal. By focusing on the concept of services, well understood by both business and technology, the component concept represents a point of integration of business and technical concerns [Stojanovic and Dahayanake, 2002]. Furthermore, a clear distinction between the concepts of component and object has been defined, defining a component at a higher-level of abstraction than an object. Regarding the size, role in the context and the nature of provided and required services, different levels of component granularity have been defined: business component, system component, and distributed component. Inside each of these levels, further detailed levels of granularity can be defined following so-called recursive composition. This means that each component can be represented as a collaboration of lower-level components, but also, in collaboration with other components, can form a higher-level component. Regardless of the granularity level, the essence of the component way of thinking is the same and represents the explicit separation between the outside and the inside of the concept (or service) represented by a component. This means that only the question ***what*** is considered (what useful services are provided by the component) not the ***how*** (how these services are actually implemented). Component is an encapsulated unit with completely hidden interior behind the interface. Interface defines all the necessary information about the component that its environment should know about and relay on, without opening that black box. The metamodel of the main component concepts is shown in the Figure 2.

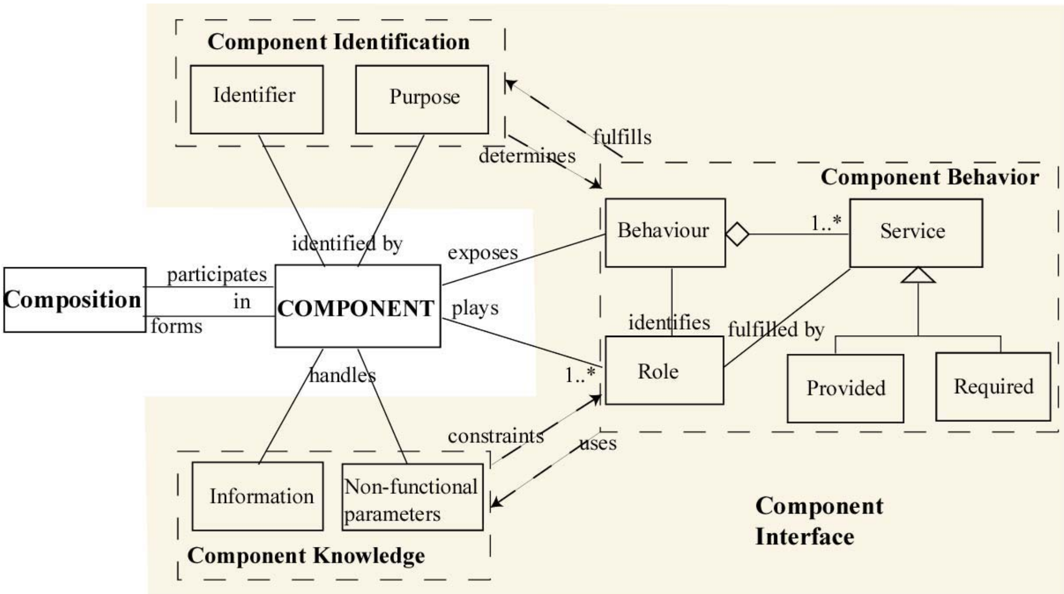


Figure 2. Component metamodel.

For the purpose of defining the state-of-the-art but also shortcomings of CBD methods and techniques proposed so far, an evaluation of CBD modeling methods has been performed. Methodology evaluation framework has been proposed for the purpose of evaluating the methods and defining the ways for improving development process support [Stojanovic et al., 2001a]. A component-oriented development process has been defined, focusing on the component concept from business requirements to implementation. The phases of requirements, analysis, design and implementation in a traditional development process has been substituted by service requirements, component identification, component specification, component assembly and deployment. After the components of the system are fully specified, a decision can be made to build components, wrap existing assets, buy COTS components or invoke web services over the Internet. The phases of an iterative and incremental component-oriented development process is shown in the Figure 3.

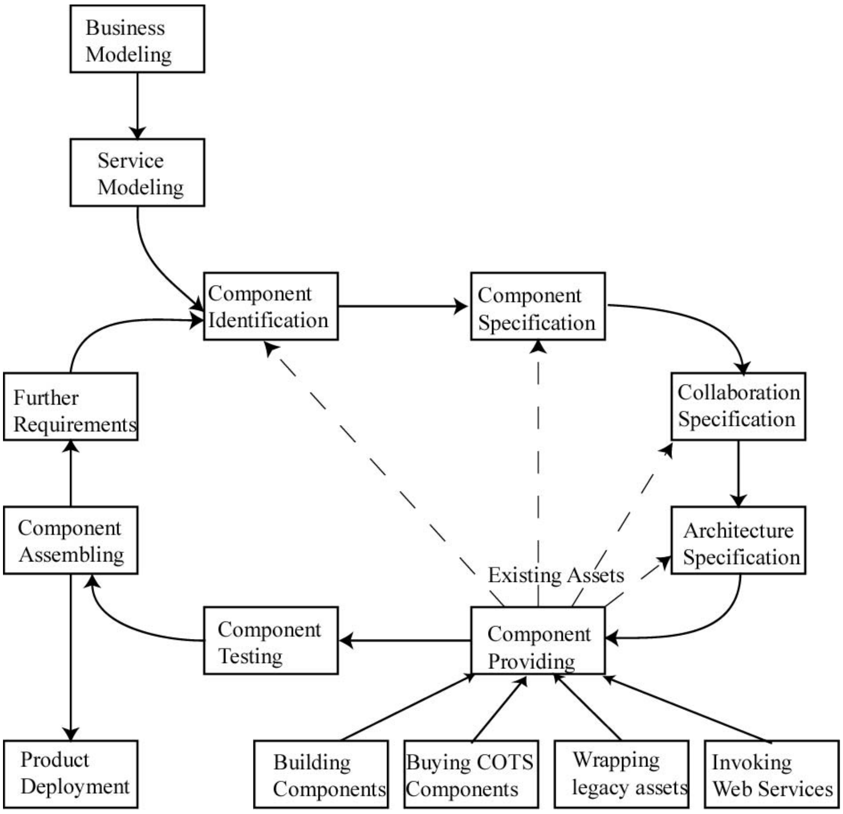


Figure 3. Component-oriented development process.

ISO Reference Model of Open Distributed Processing (RM-ODP) has been used as an underlying idea for an integrated component-oriented process support [Stojanovic et al, 2001b]. The reason for this is that RM-ODP represents an integrated specification of the complex enterprise system from different viewpoints – enterprise, information, computational, engineering and technology. RM-ODP provides a consistent, well-defined set of concepts inside the standardized separation of concerns of complex enterprise systems. Based on the RM-ODP concepts and viewpoints a useful separation of concerns through three architectural models – Business Architecture Model, System Architecture Model, and Distribution Architecture Model, has been defined. The relation between our architectural models and RM-ODP viewpoints, as well as the main relations between the models are shown in the Figure 4. While the RM-ODP concepts and viewpoints are defined using object-oriented paradigm, the three proposed architecture models are fully component-oriented, focused and organized around the component concepts. The focus of each architecture model is on the corresponding component granularity type.

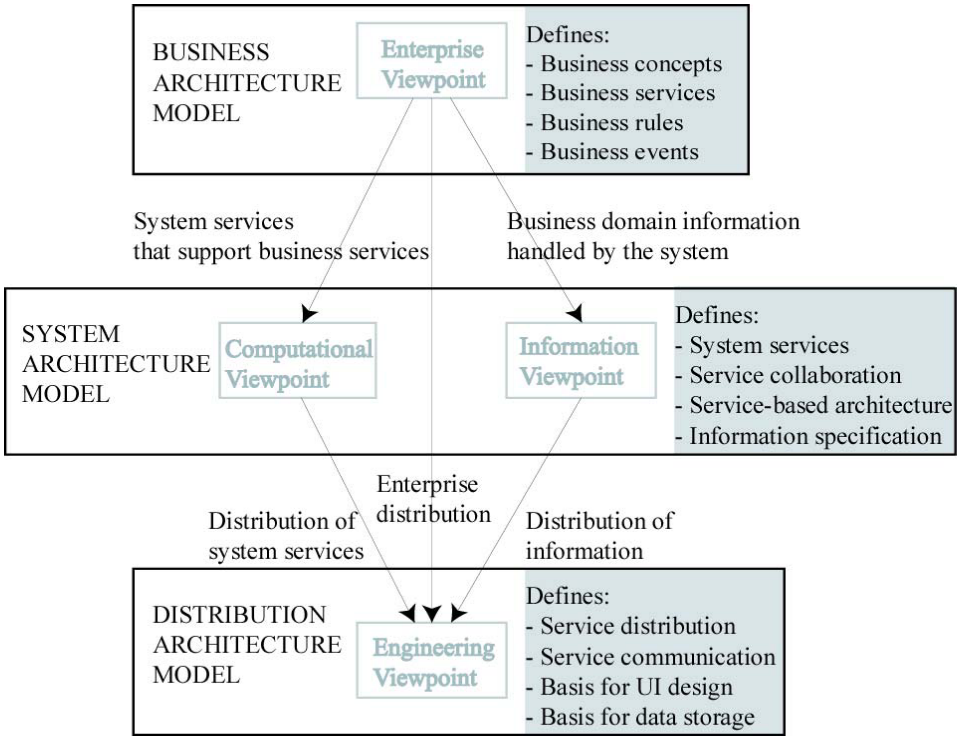


Figure 4. Relation of proposed models and RM-ODP viewpoints.

After the components at all granularity levels are identified, they should be modeled and specified. In order to model and fully specify components inside three architectural models of the system, the standard UML with proper extensions and stereotypes can be used. For more formal specification of components and their properties, a dedicated Component Specification Language (CSL) can be defined. The CSL should provide a more formal way for specifying various component concepts and properties, including self-adaptable components that are aware of user profiles, QoS, different levels of security, temporal information, geographical information, etc. It seems logical to start first with the business architecture model, than system architecture model and finally distribution architecture model, but in practice the models should be completed in incremental and iterative manner. The final result is the full system specification architecture defined and fully specified based on different types of correlated components in particular business, system and distribution settings. This represents so-called Platform Independent Model (PIM) as defined by [OMG]. This PIM can then be mapped to a Platform Specific Model (PSM) to target platforms like the CORBA Component Model (CCM), Enterprise JavaBeans (EJB) or Microsoft COM+. PSM can then be actually implemented on that particular platform. Based on the precise specification of components organized in distributed system architecture, the necessary components can also be bought as COTS components, made as wrapped legacy assets, invoked in the form of web services, and then properly plugged and assembled in the system solution.

By this moment, the work on the first inductive case studies (more experimental than real ones) has been started. The main case study in this phase is GeoPolice -

a system for supporting the work of police agents using mobile handheld devices and GIS capabilities. This case study should provide establishing and proving of the framework basic concepts, ideas and definitions. The idea in using this case study is to architect the whole GeoPolice system solution, from business requirements to complete system architecture ready for implementation, using the ICOF component-oriented concepts and guidelines. First, based on the user needs, business components of the GeoPolice can be identified and specified. They should structure the whole system development in a component manner. Business components are realized using lower-grained system components and further deployed on the distribution tiers using distributed components. Collaboration between different kinds of components as well as complete distributed system architecture must be specified. Final result should be a PIM of the GeoPolice system that can be implemented using different component platforms and components from different sources (build, buy, wrap or invoke). This case study should prove the capabilities of the framework in guiding a component-oriented system development, from business to implementation, using uniform component thinking. Components identified at the early phase of system development structure the whole development process in a component manner through smooth traceability between different levels of component granularity. Such consistent and uniform component approach supported by the ICOF reduces the complexity of enterprise systems development and provides flexibility of the solution through all the phases of the lifecycle.

Together with the GeoPolice case study, some classical examples and theoretical cases from the literature are also being used at this stage of the research, such as the video store example and the hotel reservation system. The purpose of these inductive cases performed during the first research phase is to properly define the main concepts and elements of an integrated component-oriented framework. More complex case studies are planned for the following period covering partly the first and the second research phase. Possible case studies include architecting of an Internet-enabled system for more efficient functioning of city governments as well as the system that provides urban planning and cadastre activities. Finally one case study will be chosen for evaluation purposes and will result in a prototype or partial implementation of the system. Hopefully, the research will result in the thesis defense in 2004.

## 5. Conclusion

The main goal of the research is to define a new approach to components that will provide full benefits from component-way of thinking in system development, from business services and requirements, to system analysis and design, and finally to system implementation and deployment. Achievements made so far in academia and industry treat components mainly as implementation concepts and use them for component-based system implementation and deployment over network nodes using EJB, CORBA or COM+/.NET technology infrastructures. At the same time, the way of mapping business needs into proper component-based implementation is not defined in a consistent, systematic and component-oriented manner.

The main result of the research is an Integrated Component-Oriented Framework. The framework provides a comprehensive support to a model-driven development of Internet-enabled enterprise systems, from business to implementation. The framework provides that the same component way of thinking and the same consistent set of technology-independent component concepts can be effectively applied in different aspects and phases of enterprise systems development, from autonomous business services to distributed software components. The framework should be flexible enough to be useful and applicable in different application domains, using different implementation technologies, and managing components from different sources.

The main scientific contribution of the research should consist of a new definition of component concepts, component granularity levels, a component-oriented development process, and other concepts and elements of the framework necessary for effective support of component-oriented business-driven system development. The framework should provide flexibility and reduce complexity in a development process and a developed system itself. The practical contribution of the research should be applicability of the framework in real application cases where it should provide efficient and systematic component-oriented system development, starting from business requirements and ending in working application.

## Acknowledgements

The author would like to thank his promoter Prof. Dr Henk Sol and supervisor Dr Ajantha Dahanayake for their support and valuable suggestions in carrying out this research. The author gratefully acknowledges the support of the BETADE research project (http://www.betade.tudelft.nl) and its members, as this research represents a part of the BETADE.

## References

Allen, P. and Frost, S. (1998), "Component-Based Development for Enterprise Systems: Applying the Select Perspective", Cambridge University Press.

Booch, G., Rumbaugh, J., Jacobson, I. (1999), "The unified modeling language user guide", Add.-Wesley.

Brown, A.W., Wallnau, K.C. (1998), "The Current State of Component-Based Software Engineering", IEEE Software, September/October 1998.

Butler Group (1998), " Component-Based Development", Management Guide, Researched by D. Sprott and L. Wilkes, available at http://www.butlergroup.com

D'Souza, D.F., Wills, A.C. (1999), "Objects, Components, and Frameworks with UML: the Catalysis Approach", Addison-Wesley.

Galliers R. (Eds.) (1994), "Information Systems Research: Issues, Methods and Practical Guidelines"

Gartner Group (1997): "Componentware: Categorization and Cataloging," Applications Development and Management Strategies Research Note, by K. Loureiro and M. Blechar, December 5, 1997, http://www.gartnergroup.com.

GeoEurope (2000), "GIS in business", April 2000, available at http://www.geoplace.com.

Henderson-Sellers, B., Szyperski, C., Taivalsaari, A., Wills, A., (1999)"Are Components Objects?" OOPSLA'99 - Panel Discussion.

Jacobson, I., Booch, G., Rumbaugh, J. (1999), "The unified software development process", Reading MA: Addison-Wesley.

Microsoft, COM Technologies and .NET Web Service Platform, information at http://www.microsoft.com/com/ and http://www.microsoft.com/net/

OMG Object Management Group, MDA- Model Driven Architecture, information available at http://www.omg.org/mda/

Siegel, J. (2000), "CORBA 3: Fundamentals and Programming" OMG Press, John Wiley & Sons.

Sol, H.G. (1988), "Information System Development: A Problem Solving Approach", in Proceedings of 1988 INTEC Symposium Systems Analysis and Design: A Research Strategy, Atlanta, Georgia.

Stojanovic, Z., Dahanayake, A.N.W., Sol, H.G., (2001a) "A Methodology Framework for Component-Based System Development Support", the sixth EMMSAD'01, Interlaken, Switzerland, June 4-5 2001, pp. XIX-1 – XIX-14

Stojanovic, Z., Dahanayake A.N.W., Sol H.G., (2001b), "Integration of Component-Based Development Concepts and RM-ODP Viewpoints", the 1st Workshop on Open Distributed Processing WOODPECKER 2001, Setubal, Portugal, July 6-7, 2001, pp. 98-109.

Stojanovic Z., Dahanayake A.N.W., (2002), "A New Approach to Components", accepted for the 2002 Information Resources Management Association (IRMA) International Conference, to be held in Seattle Washington, USA, May 19-22, 2002.

Sun Microsystems, Enterprise JavaBeans, The source for Java™ technology at http://java.sun.com

Szyperski, C. (1998), "Component Software: Beyond Object-Oriented Programming", ACM Press, Addison-Wesley.

Yin R.K. (1994), "Case Study Research: Design and Methods", Second Edition, SAGE Publications.