ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 2, Issue 4, pp: (279-284), Month: October - December 2014, Available at: www.researchpublish.com

# A Comparative Study On Component-Based Software Engineering And Model-Driven Development On ProCom

R. Shobana[1], R. Vidhya lakshmi[2]

[12]Department of computer science and application, D.K.M College for women, vellore, India.

***Abstract*: Component-based software engineering (CBSE) and model-driven development (MDD) are two approaches for handling software development complexity. In essence, while CBSE focuses on the existing software modules called components; MDD promotes the usage of system models result with an implementation of the desired system. Even though they are different, MDD and CBSE are not mutually exclusive. The main goal of this thesis is to summarize the theoretical background of MDD and CBSE, and to propose and apply a systematic method for their comparison. The comparisons results are summarized, analyzed about future work on ProCom are made.**

***Keywords*: Component-based software engineering, Model-driven development, ProCom, Comparison.**

## I. INTRODUCTION

Two systematic approaches to coping with the development complexity are component-based software engineering (CBSE) and model-driven development (MDD). CBSE focuses on the construction of systems from existing software modules called components, and makes a clear distinction between developing a component and developing a system. From the perspective of CBSE the development of a component should result with a reusable software module implementing a cohesive set of functionalities called a component. System development is the selection and "binding together" of existing components. A key concept for this paradigm is reusability, since the developed components are meant to be relatively self-sustained and thus may be used in various systems.

## II. PURPOSE

This thesis has two main goals. The first one is to systematically compare CBSE and MDD. The thesis also aims to analyze the comparison results and to suggest how these two paradigms can be combined.

The second main goal is to enrich the results from the general comparison with a case study comparing CBSE and MDD with respect to ProCom. This comparison aims illustrate the key differences and similarities between CBSE and MDD with the concrete technological aspects of ProCom. Also, the thesis aims to analyze further the outcome of the practical case study and to suggest future improvements of ProCom so as to better accommodate features of the two paradigms.

## III. COMPONENT-BASED SOFTWARE ENGINEERING

The philosophy of CBSE is to build software systems from pre-existing components rather than to develop them from scratch. Ideally this should lead to reduced development time and efforts because of the component reuse. The paramount concept in CBSE is the ***component***.

"*A software component is a unit of composition with contractually specified interfaces and explicit context dependencies only. A software component can be deployed independently and is subject to composition by third party*". A major topic in

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 2, Issue 4, pp: (279-284), Month: October - December 2014, Available at: www.researchpublish.com

CBSE is the specification of components' functionality and behavior. Since components are "visible" only through interfaces their specification contains only the specifications of their interfaces. There are three kinds of interface specifications - ***syntax***, ***semantic*** and ***extra-functional***.

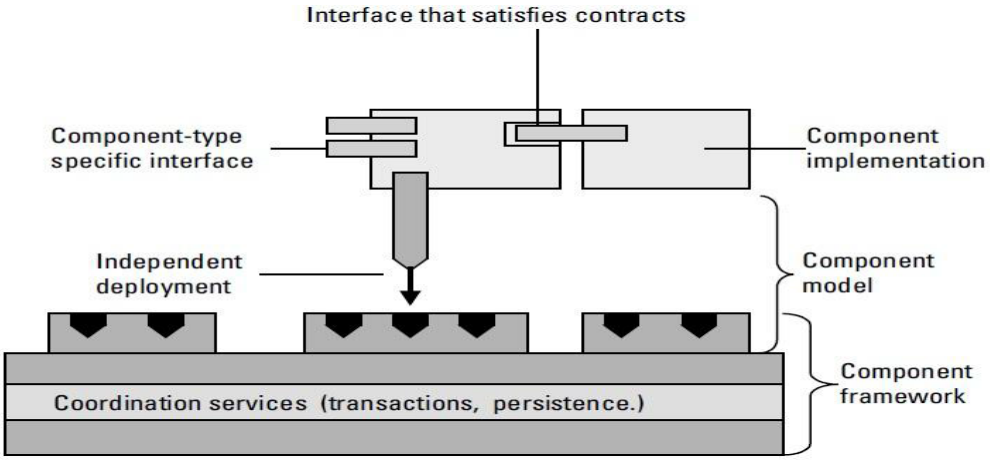


**Figure 1. Design of a typical component based system**

The component model defines the requirements for a component which is to be deployed in the framework. Each component defines its incoming and outgoing interfaces. All incoming interfaces of a component must be resolved before it is successfully composed into a framework.

## IV. MODEL-DRIVEN DEVELOPMENT

Models in MDD are used to reason about a problem domain and design a solution in the terms of that domain. The philosophy of MDD is to start with models which are simple and include only the most essential parts of the designed system. "*a model is a set of statements about some system under study*". The sake of simplicity in this classification it is not considered that source code itself is a system model, though it can be viewed this way.

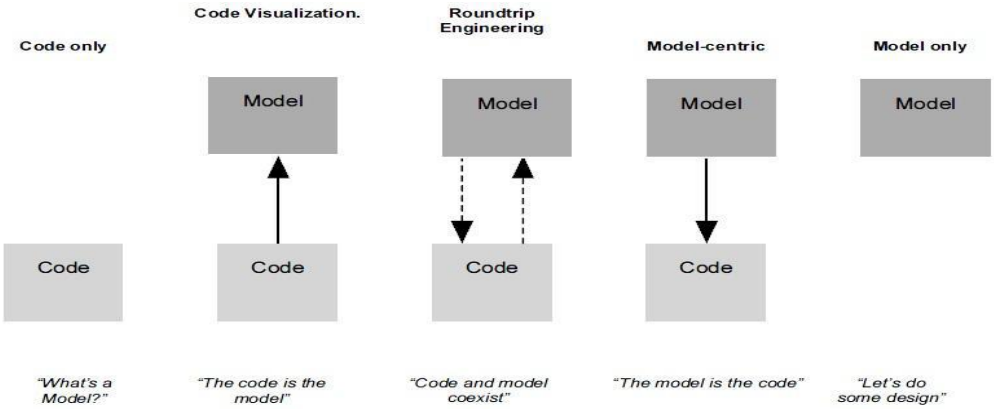


**Figure 2. The modeling spectrum**

***Code only*** - this approach considers the source code as the only representation of the system.

***Code Visualization*** - this approach uses the source code itself as an initial representation over which transformations are applied.

***Roundtrip Engineering*** - takes the idea of Code Visualization one step further.

***Model-centric*** - This approach requires that the models of the system are detailed enough so that the automatic generation of the full source code is possible.

***Model only*** - In this approach models are used for design discussions and analysis, better understanding of business requirements, communication etc.

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 2, Issue 4, pp: (279-284), Month: October - December 2014, Available at: www.researchpublish.com

## V. METHOD OF COMPARISON

This method is applied by discussing and comparing the influences of both CBSE and MDD. The comparison method proposed in this thesis is, two-level hierarchy of what is called ***comparison aspects***.The aspects are requirements, design, development, organizational specifics and business specifics.

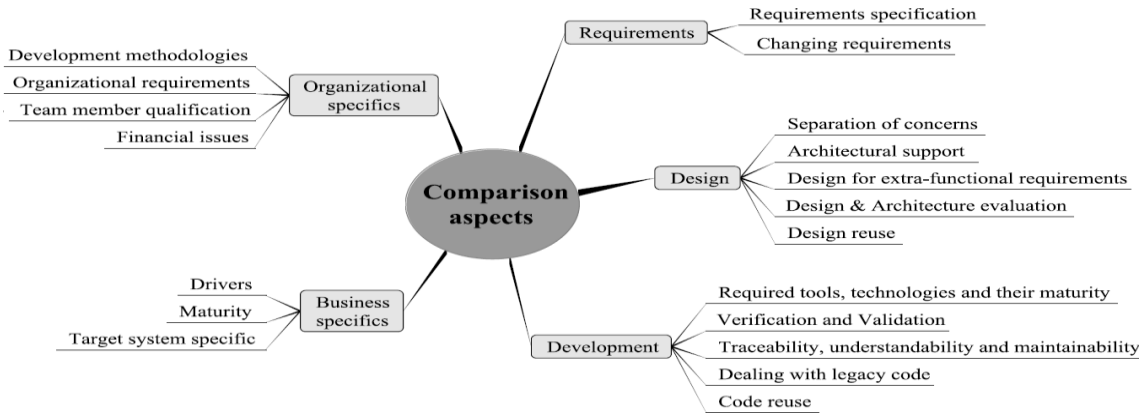


**Figure 3. Hierarchy of comparison aspects**

The method proposed herein is a general one and can be used to compare development approaches other than MDD and CBSE as well. This allows for systematic analysis of the impacts the objects under consideration have on the different phases of software development.

## VI. COMPARISON WITH RESPECT TO PROCOM

The goal of this section is to pave the way for the comparison of CBSE and MDD with respect ProCom, which is the second main goal of this thesis. ProCom is a component model designed for the development of component based embedded systems in the vehicular-, automation- and telecommunication domains. The goal is to provide theories, methods and tools to increase the quality and reduce the costs of embedded system development. A subsystem is specified by typed input and output ***message ports***, which define what type of messages the subsystem can receive and send. Besides message ports, the external view of a subsystem also contains a set of attributes and models related to functionality, reliability, timing and resource usage. They are used for analysis and verification throughout the development process. That is from CBSE perspective the interface of a subsystem consists of message ports, attributes and models.

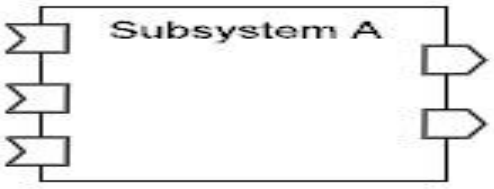


**Figure 4. External view of a subsystem with three input message ports and two output message ports.**

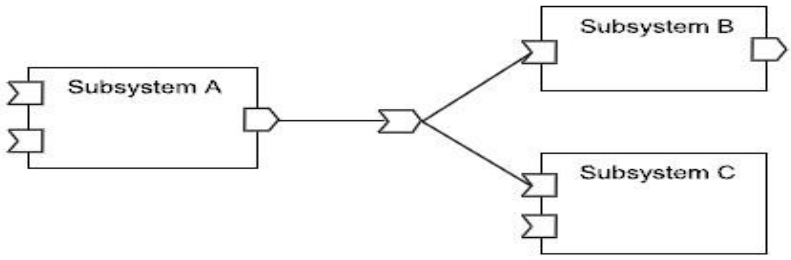


**Figure 5. Subsystems and a message channel**

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 2, Issue 4, pp: (279-284), Month: October - December 2014, Available at: www.researchpublish.com

A Subsystem within a given system communicates with each other through ***message channels***. A message channel connects one or more output ports to one or more input ports. The types of the input and output ports connected by a message channel must be compatible. Message passing is a non-blocking (asynchronous) action.

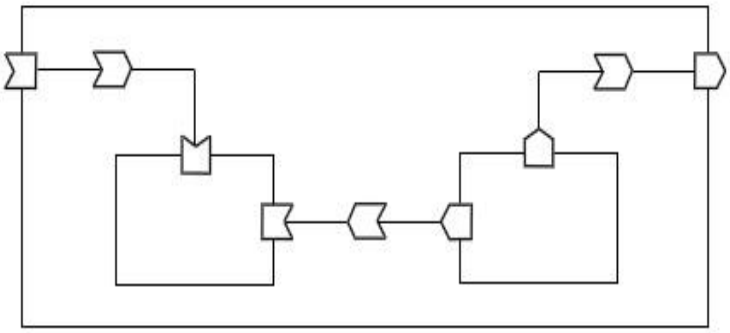

**Figure 6. An example of a composite subsystem**

ProSys is a hierarchical component model, meaning that subsystems can consist of other subsystems. Subsystems made up of other subsystems are called ***composite***, while those which are not are called ***primitive***. Primitive subsystems are usually internally modeled by ProSave components. ProSys composite subsystem internally consists of other subsystems thus forming a hierarchy. The subsystems comprising a composite one are related with message channels, which provide them with a mechanism for message exchange.

In ProSave, a subsystem is constructed by binding together components which encapsulate relatively small and rather low-level, non-distributed functionality. Unlike ProSys, ProSave components are passive.

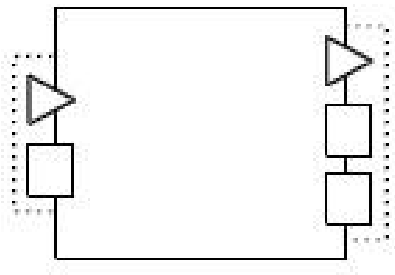

**Figure 7. A ProSave component with two input ports (on the left) and three output ports (on the right). Triangles and boxes denote trigger ports and data ports, respectively**

An ***input port group*** which consists of a single trigger port and a set of data ports. The data ports are used to store the parameters needed for the component execution.

A set of ***output port groups***. Each output group consists of a single trigger port which indicates when the resulting data is ready and a set of data ports used to store these results.

ProSave defines a model element called ***connection***, which is used to represent communication between components. A connection is a directed edge between an output port of a component and an input port of another one. These ports must be both either data ports or trigger ports. Connections between data ports denote data transfers. Connections between trigger ports define control flow transitions. The transfer of both data and triggering over a connection is loss-less and atomic.

Besides connections, ProSave defines another type of model elements that represent communication between components – *connectors*. They do not represent unconditional transfer of data or triggering like connections do.

Connectors can be used to manage the data- and control-flow between components. Usually connectors are represented by rounded rectangles with their names inside, but the most widely used connectors may have simplified notations.

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 2, Issue 4, pp: (279-284), Month: October - December 2014, Available at: www.researchpublish.com

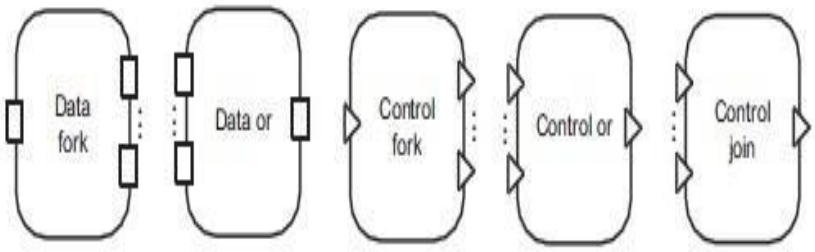


**Figure 8. Examples of connectors**

A major difference between components and connectors is that connectors are not limited in terms of a single input and a single output trigger port. A connector may have multiple input and output trigger ports or may not have any trigger ports at all.

ProSave and ProSys are used to model the functional architecture of an embedded system.

ProCom introduces the notion of ***virtual nodes***. Virtual nodes can be described as an intermediate level in the allocation of functional units to physical nodes. A virtual node represents an encapsulated abstraction of behavior with respect to timing and resource usage. Hence, a virtual node is also a reusable design unit like the other design-time components in ProCom. Each virtual node is associated with one or many ***resource budgets*** defining a minimum level of resource availability provided to a subsystem deployed in it.

## VII. CONCLUSION

In this thesis two systematic comparisons of CBSE and MDD were carried out. Following a new comparison method was introduced, which is based on a two-level hierarchy of comparison aspects. The discussions of the comparison aspects identified that ProCom augments the core concepts of CBSE with several model driven approaches, as proposed in the general comparison. Consequently, the shortcomings of CBSE concerning traceability, maintainability, analyzability and specifying architecture and design are mitigated in ProCom. It was identified that these could be further improved if requirements modeling capabilities were incorporated into ProCom. Such capabilities could also make ProCom systems more agile to requirement changes. A major problem for ProCom is the need for specific development methodologies, which is an inherent problem to CBSE.

## REFERENCES

[1] Jones, C.: Software Engineering Best Practices: Lessons from Successful Projects in the Top Companies 1st edn. McGraw-Hill Osborne Media (2009)

[2] McConnell, S.: Code Complete: A Practical Handbook of Software Construction 2nd edn. Microsoft Press, Redmond, Washington, USA (2004)

[3] Childs, A., Greenwald, J., Jung, G., Hoosier, M., Hatcliff, J.: CALM and Cadena: metamodeling for component-based product-line development.

[4] Carlson, J., Feljan, J., Mäki-Turja, J., Sjödin, M.: Deployment Modelling and Synthesis in a Component Model for Distributed Embedded Systems.

[5] Herzum, P., Sims, O.: Business Component Factory : A Comprehensive Overview of Component-Based Development for the Enterprise 1st edn. Wiley (2000)

[6] Szyperski, C.: Component Software Beyond Object-Oriented Programming 2nd edn. Addison-Wesley (2002)

[7] Crnkovic, I., Larsson, M.: Building Reliable Component-Based Software Systems. Artech House Publishers (2002)

[8] Eiffel Software: The Power of Design by Contract. (Accessed March 10, 2011) Available at: http://www.eiffel.com/developers/design_by_contract.html

[9] Selic, B.: The Pragmatics of Model-Driven Development. IEEE Software 20(5), 19-25 (September 2003 )

[10] Beydeda, S., Book, M., Gruhn, V.: Model-Driven Software Development 1st edn. Springer (2005)

[11] Bachman, F., Bass, L., Buhman, C., Comella-Dorda, S., Long, F., Robert, J., Seacord, R., Wallnau, K.: Volume II: Technical Concepts of Component-Based Software Engineering. Technical report, Carnegie Mellon Software Engineering Institute (SEI) (2000)

[12] Selic, B.: The Pragmatics of Model-Driven Development. IEEE Software 20(5), 19-25 (September 2003 )

[13] Feljan, J., Lednicki, L., Maras, J., Petricic, A., Crnkovic, I.: DICES technical report Classification and survey of component models. Technical report (2009)

**AUTHOR'S PROFILE:**

**Prof. Shobana R** is an Assistant Professor in the Department of Computer Science at DKM College for Women, vellore. Her special research interests are in software engineering, data mining, and programming in C, VB and Digital electronics.

**Prof. Vidhya Lakshmi R** is an Assistant Professor in the Department of Computer Science at DKM College for Women. Her special research interests are in software engineering, data mining, VB, Information technology and Networking.