IJCSMS International Journal of Computer Science and Management Studies, Vol. 12, Issue 02, April 2012
ISSN (Online): 2231-5268
www.ijcsms.com

Design of Model for Component Based System

Sh. Ashok[1], Dr. Vijay Deep Gaur[2]

[1]Department of Computer Science, Shri Venkateshwara University, Gajraula, India
ashokrtk@gmail.com

[2]Assistant Professor, Government College Krishan Nagar, M.Garh, Haryana, India
vijaydeepgaur83@gmail.com

Abstract

This paper is based on designing such kind of model that will enhance the reusability of software modules, while component based software development approach is used to develop software. It is widely growing engineering discipline that deals with the cycle of developing components and developing with the components. In software industries this approach is very popular. This model not only reduces the development time & cost, but also increases the productivity, reliability, portability and scalability.

Keywords: *Component based software development, component based software engineering, reusability.*

Introduction

As we know that most of software modules[1,2] are, heterogeneous system that are made of many unrelated parts including interfaces, software, hardware, operating systems, programming languages, and network protocols.

We can find lots of solutions to heterogeneity problems at all levels of computer technology. For instance, the internet is an example of success at network and transport protocol layers of heterogeneous communication solution, the key to this success is standardization. In one hand, Component-Oriented Middleware Platforms arise as a way to solve the operating systems; programming languages, platform, and networking heterogeneity of inter processes component communication. On the other hand, the boom of middleware systems has transferred the problem of heterogeneity to this level. In this moment, two different middleware platforms such as CORBA and Java RMI can not interoperate. In this case, none of the existing platforms have won the standardization's war.

The results of this study are:

- A taxonomy definition inspired on other heterogeneous communications technical solutions, which were already working properly in the realm of networking communications such as: adapters, bridges, tunnels, switches and buses.

- Identification of requirements needed to build heterogeneous interoperable platforms solutions.

In other research on abstract machines [3], we discovered how virtual machines and reflection can support interoperability since there is an isomorphic structure and behaviour between abstract machines and middleware platforms. The above mentioned works steered to a first Object Oriented Reflective Abstract Machine Architectural Model not based on components. This architectural model specifies the creation of software designs to implement systems that allow heterogeneous interoperability among applications with most of the requirements identified in[4] and described bellow.

With the idea of getting more easily extensible and modular systems here we expose a second version that is an Architectural Component-Based Model [5]. Besides, extending the system

to interoperate with a new middleware platform is as easy as creating and adding a new component by implementing two interfaces.

Literature Review

Component-based approaches for the development of information systems are a widely growing engineering discipline that deals with the cycle of developing components and developing with components. Software engineers try to borrow concepts of components composition from other engineering disciplines to component software (Khayati and Giraudin, 2002)[6]. CBSD focuses on construction of large systems using existing components. This approach is geared at constructing large flexible and reliable systems, at a reduced cost and in reason-able time (Harriot, 2003) [7] .

A notable characteristic of component-based development is it emphasis on integrating independently developed components produced by multiple developers. Thus, while component-based development can benefit from the capabilities of previous generations of environments, its special nature induces requirements for new capabilities not found in previous environments (Luer and Rosenblum, 2001) [8].

Before each stage of the research process, a specific study of relevant literature will be done to support and check the actions to be taken.

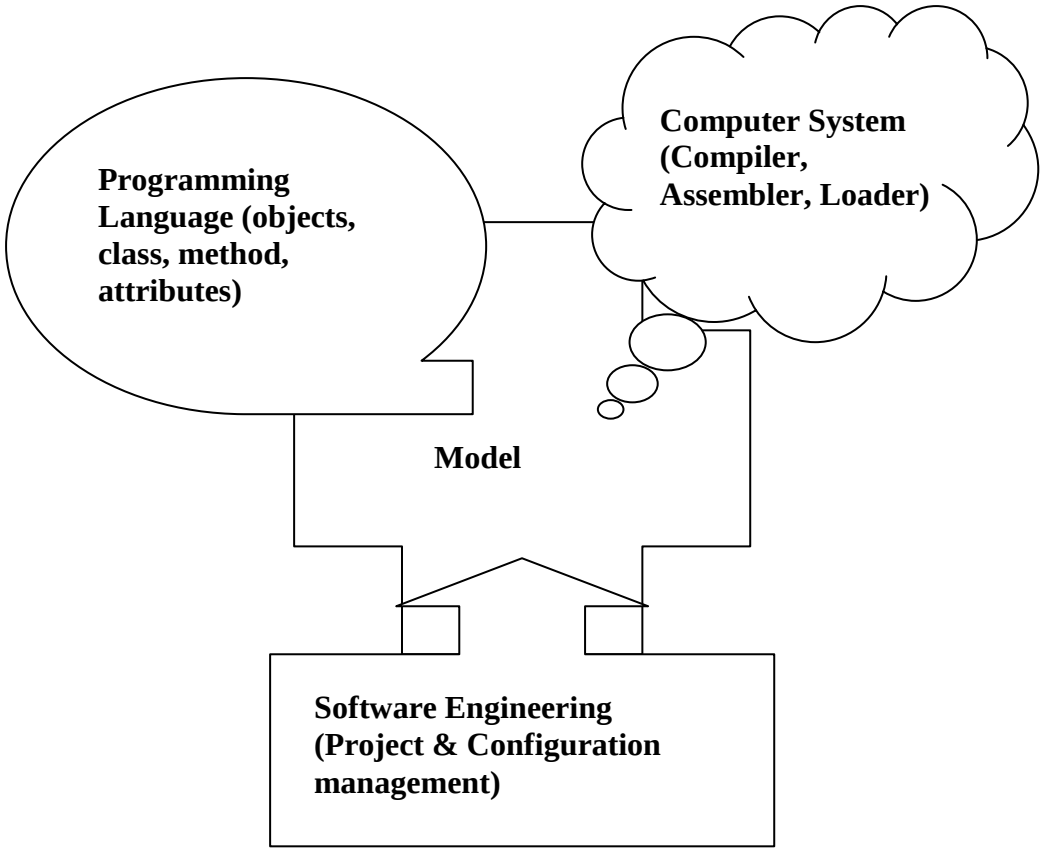
A decision will be made whether to modify an existing OO system to develop one from scratch, after reviewing existing system.
Software development will be based on a hybrid model since different parts of the system will be developed using different approaches. Software development will be based on the evolutionary model of the software process. Software development will be divided into phases. Thus each phase of evolutionary will represent a phase with the innermost loop representing the first phase and so on. In each phase, the following will be done.

- Defining the objectives for the phase. Constraints on the process and risks will be identified. Depending on the risks alternative strategies may be planned.
- A development model for the system will then be chosen and the development for the phase done.

On producing the product for the phase, the phase will be reviewed and a decision whether to continue with another phase made. If a new phase is to begin, plans for the next phase will be drawn.

Proposed Works

The classic idea behind to design this model to enhance reusability of model not even design phase of software development, but also enhance reusability model further stage of software development. It also helps to increases the performance of web based software modules in terms of portability, scalability etc…



'**Proposed Model**'

The benefit of this model is direct or indirect during software development. The direct benefits include the reuse of the components and knowledge congealed in them.

And indirect benefits include cost saving during qualification, development, testing and to measure the quality and reliability.

This model is help-fulfil for following purpose:

I. A facility for component composition; this facility should enable the composition of deployable components in assembling a new application or maintaining an existing one.
II. Components with mandatory self-description; the environment should have a mechanism for ensuring that each component used is properly self-described.
III. A facility for component adaptation; this facility should allow a pre-packaged component to be adapted so as to suit a specific need required by the application under consideration.
IV. Support from a broad scale repository of reusable components; the environment should be able to use the services of a software repository stocked with reusable software components that span all software development life cycle activities.
V. Support for all life cycle activities; the environment should have the necessary tools to support all activities of the software development process.
VI. Retrieval of components from the repository based on behaviour specification.
VII. Platform independent target environment; Application development should allow deployment on various hardware and operating systems platforms.
VIII. Collaboration with third party repositories; the environment should allow the use of components from other

repositories as long as the components meet specified standards.

IX. Cooperation with other development environments; the environment should facilitate application configuration across other development environments.

X. Scalability motivated by changing user requirements.

XI. Support for access to remote repository; the environment should allow access to support repositories located remotely.

XII. A formal specification to define mission critical applications; There should be a tool in the environment that allows for application specification.

XIII. A code generation facility; once the application has been modelled, the environment should provide a tool to transform this model to an implementation model by generating the necessary source code to realize the application.

XIV. Graphical user interfacing

The major goals of CBSE are the provision of support for the development of systems as assemblies of components, the development of components as reusable entities and the maintenance and upgrading of systems by customizing and replacing their components (Heineman and Councill, 2001)[9].

Objective of Proposed Works

The main objective of proposed works is that it must be cost-effective to:

I. Create Reusable Components: The design of reusable components is difficult,
   a. first create such kind of components that becomes is more critical since error are replicated whenever a component is reused
   b. Second, components must be understandable, i.e. well-written and well documented
   c. Third, component must be easily adaptable for different uses, either in original or in modified form

II. Find existing components: Components must organize 'properly' so that they can be rapidly found when needed by the programmer to help in their reuse.

III. Share components among different users: To truly facilitate reuse, it must be possible for several programmers to co-operating policies need to be established by the co-operating users to derive the most from such as a shared environment. Sharing is aggravated when user are working in a distributed programming environment.

IV. The main objective of proposed model is to develop such kind of an environment that supports CBSE in its effort to re-use pre-existing components in the software development process.

Conclusion

The component based development model leads to software re-used and re-usability provides a number of tangible benefits. It leads to

I. Reduction in development cycle time.
II. Significant reduction in project cost.
III. Help in significant increases the
   a. Productivity
   b. Reliability
   c. Portability
   d. Scalability

References

[1] Grady BOOCH, Ivar JACOBSON, and James RUMBAUGH. RUP Software Engineering. 1997.

[2] Kruchten P., « Architectural BluePrints – The "4+1" view Model of Software Architecture », IEEE Software 12, pages 42-50,1995.

[3] Francisco Domnguez Mateos, Manuel Rubio. Investigacin en Sistemas Distribuidos
ysu Heterogeneidad, Servitec. ISBN: 84-689-3501-8, 2005.

[4] Francisco Domnguez. Dise˜no e Implantaci´on de la M´aquina Abstracta MA y la
extensi´on reflectiva de Carbayonia, Servitec. ISBN: 84-689-3500-X, 2005.

[5] Lau, K. K., & Wang, Z. (2005). A Taxonomy of Software Component Models.

[6] Proceeding of the 2005 31st EUROMICRO Conference on Software Engineering and
Advanced Applications

[7] Khyati, O. and Giraudin, J.P. (2002). Components retrieval systems. OOIS Workshop on Reuse in Object Oriented Information, France.

[8] Harriot, K.K.(2003). Component-Based Development Environments: Current Research Efforts. Msc.Thesis, University of the West Indies

[9] Luer, C. and Rosenblum, D.S. (2001). WREN- An Environment for Component-Based Development. In:Proceedings of the Joint 8th European Softwaring

[10] Engineering Conference and 9th ACM SIGSOFT Symposium on the Foundations of Software Engineering Sep. 2001, vienna, Austria, 207-217.

[11] Heineman, G. and Councill, W. (2001). Component-based Software Engineering: Putting the Pieces Together. Addison Wesley, California, USA.