Journal of Computer Science 7 (8): 1167-1171, 2011
ISSN 1549-3636
© 2011 Science Publications

Integrating Components in Software Product Line to Build High Quality Products

Lena Khaled
Department of Software Engineering, Faculty of Sciences and
Information Technology, Zarqa Private University, Amman, Jordon

Abstract: Problem statement: The main part of building any system is achieving high level of quality and developing qualities it is achieve. Many organizations do not take into account the highest level of quality as a main necessary part through building its systems; they think mainly on budget and reducing time to market. **Approach:** One of the important approached to achieved quality was used components through building products and then selecting the most appropriate component to put them into the product line according to system requirements. **Results:** The main result of adopting component-based approach to software product line was premise of high quality in addition to reused and reduced time to the market. **Conclusion:** The ultimate goal of using components through software product line is to increase quality of software as flexibility, reliability in addition to the characteristic of making software reusable in another types of business especially in electronic commerce application.

Keywords: Software Product Line (SPL), Component Based Software Development (CBSD), software quality, Configuration Management (CM)

INTRODUCTION

Reuse-based is a software engineering strategy where the life cycle process changed to reusing existing software. Moving to this strategy has been in response to need for lower cost, fast delivered system and high quality of software. Software units may be in different sizes. For example: object and function reuse where a single function in the component may be reused, component reuse is another example which represents that components can interoperate within a framework and finally application system reuse where the whole application may be reused by integrated it into other systems. Because software product became complex and large and all customers need dependable software that has a high level of qualities to develop more quickly, component based development has become an important development approach. Achieving high quality is the main important goal to any company for building its system, in addition to that, system's goal affected mainly with the quality attributes (Khaled, 2010). So adopting reuse strategy is the main concept that must focus on. Therefore, we must adjust and extend components to use in products and then make product more standard and hence more reusable.

This study is concentrate on integrating components in Software Product Line (SPL), it shows how SPL can consists of selected components which have particular qualities to build the complete framework of the system to specify the customer's requirements and then achieve goal.

Related works: Fazal_e_Amin *et al*. (2011) discussed the state of art of the reusability assessment of software components, this work presented that reuse is facilitated by employing a systematic methodology such as in software product line. A validation technique is used to validate the result application such techniques are human evaluation, statistical analysis, experiment, test data, case studies are also used to validate the result application.

(Omar *et al*., 2011) worked on changing in software requirements. It presented that this changeability can be solved through agile methodology. It presented the initial finding on the impact of agile approach among software engineering teams in one of the computer center in Malaysia. It exposed that data in this research collected from four software engineering teams through interview, short-term observations and questionnaire.

(Fazal_e_Amin *et al*, 2010) this study enhanced modularity in software by using aspect orientation. Thus, modularity directly related to the reusability and this main core to build any software product. This relation between aspect oriented technique and software product line caused the development of aspect oriented software product line components.

Ab-Rahman *et al*. (2009) built a specific architectural design to achieve one type of quality that is reliability. Reliability is a failure free operation; this study solved a failure problem by building a proposed network design so it improves the reliable operation to the system.

This study is concentrate on integrating selected components in Software Product Line (SPL) to build high quality systems.

Software Product Line (SPL): It consists of a group of software systems (products) with common features to identify the needs of market or achieve a specific goal; it contains a set of various reusable components shared between these groups of systems. Features of components in the product line must suitable to the properties of the product. This facility enhances the activity of quality assurance to the product.

The design of SPL can use several tools and techniques to achieve reusability such techniques are object oriented frameworks, design pattern, aspect oriented and component based and the last one is the main core of this study.

Product line framework: A framework of product line consists of three steps following one another in a series. These steps are requirement specification, configuration management and the development of the product line. In the following, we discuss each step in detail and in the result part; we show how this framework communicates with component to build a high quality product.

Requirement specification: The concept of requirement means what the users want from the systems, but it says nothing about how the system should built. Requirement specification should avoid implementation issues because user could not expect to understand technical issues of the implementation and therefore he could not agree elements of the specification. A good specification should have the following characteristics (Omar *et al*., 2011):

- Un ambiguous: A requirement must express the whole need and state all constraints under which it applies
- Completeness: It is the property that the requirements are sufficient to distinguish the desired behavior of the component from any other undesired one
- Finally, all specified requirements must define clearly and this is the main characteristic

For a specification, two types of documents appeared:

- A requirements specification which written specifically to the users describing the user view of the system and is written in natural language
- A technical specification written primarily by developers, described in some formal notations

Configuration Management (CM): can be defined as using tools, methods and standards for managing and extending any software system. In software product line, requirement specifications always change during use and these specifications must always incorporate with new versions of the software therefore we must use effective CM to organize these changes. Working with the framework of the product provides various advantages like managing different components but also has disadvantages like additional level of complexity.

Integrating components in product framework introduces two main problems in CM those are: (Kaur and Singh, 2010):

Controlling same component in more than one frameworks
- Building a new framework from existing components

Development SPL: To develop process of SPL the most appropriate choice is to use agile iterative method and incremental process, here SPL contains many components increment horizontally or vertically, component integration defined in the component section in detail.

Software product line uses a life cycle approach to create the final product. This life cycle are product analysis, product design, product implementation and product testing? Figure1 illustrates SPL in detail.

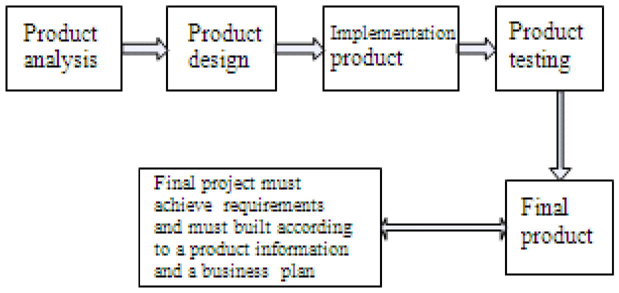
Product analysis
Product design
Implementation product
Product testing
Final product
Final project must achieve requirements and must built according to a product information and a business plan

Fig. 1: The Software Product Life Cycle (SPLC)

Figure1 represents how the final product must identify customer's requirements and must build according to a business plan.

A component: Software component is an independent piece of software and can be reachable through its interfaces, so in order to combine with other components it must be possible to obtain details of component interfaces. Component environment must be interoperable, wrapping existing applications and platform independence. A software component usually implemented for a specific component technology.

Component Based Software Development (CBSD): An approach focuses on the integration of pre-built software components to build products that increase flexible and portable attributes (Sharp *et al.*, 2010).

Therefore, component based is a new trend to software development and it has many advantages:

- Reducing cost
- Improving maintainability and flexibility of the product by replacing anew components instead of an old one
- Enhancing system quality done through developing components by those who are professional in the application area which base in the components on its build

The life cycle of software components handled by product line framework. The framework of the product line allows components that play different roles in a different framework s to be integrated (Kaur *et al*., 2010).

MATERIALS AND METHODS

The basic concept using in CBSD is component integrator, the success of the development depends on the ability of integrator to select the most suitable component that used to build the product. The component integrator can use a specific environment to show the constraints that imposed on the process; all this done through sequence of steps named selector processes. Selector process consists of three main stages:

- Component supplier :supplies the components that used to build the product
- Component integrator: select the most appropriate components that can be used
- The product: the final product that consists of selecting components

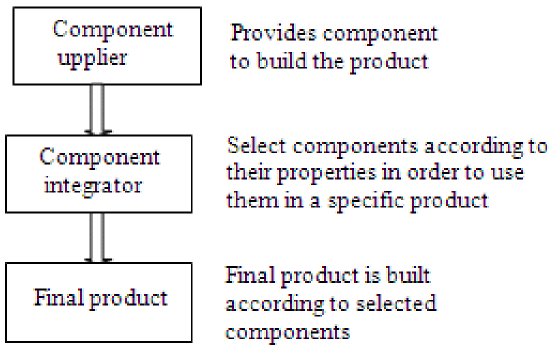
Component upplier
Provides component to build the product
Component integrator
Select components according to their properties in order to use them in a specific product
Final product
Final product is built according to selected components

Fig. 2: The selector process stages

Figure 2. represents the dependability of component integrator on component supplier to constantly describe the product that consists, the supplier effectively describe simple and complex properties of the components and the associated certifications ensures this descriptions. We describe the component certifications in detail in result.

RESULTS

Figure 3 represents the idea of this study; it works as follows. First, the product built on a set of requirements specified previously. The important thing is building the product upon these requirements and on a high level of quality.

To enhance achieving a high level of quality to the product, this only done by using a set of components the supplier provides and the selector chose the components based on the specified requirements within the framework. Selecting the component made through the certification that associated each component during the supplements.

Component certification is the information that available about component, to present component for reuse or sale, this type of information treats component as a black box and maintained in a component catalogue. One of the important items in component certification is that it should be possible to estimate the component's quality; within a company, an independent group of quality assurance could take the role of estimation. One idea for ensuring the trust in quality of components is a process of third parity certification,, many companies test the components and then give the component's certification to specify the fitness of the component to a specific purpose. The term component catalogue used before specify: the existing component, the importance of the component, the status of the component (if designed or under development) and finally, the interface of the component.

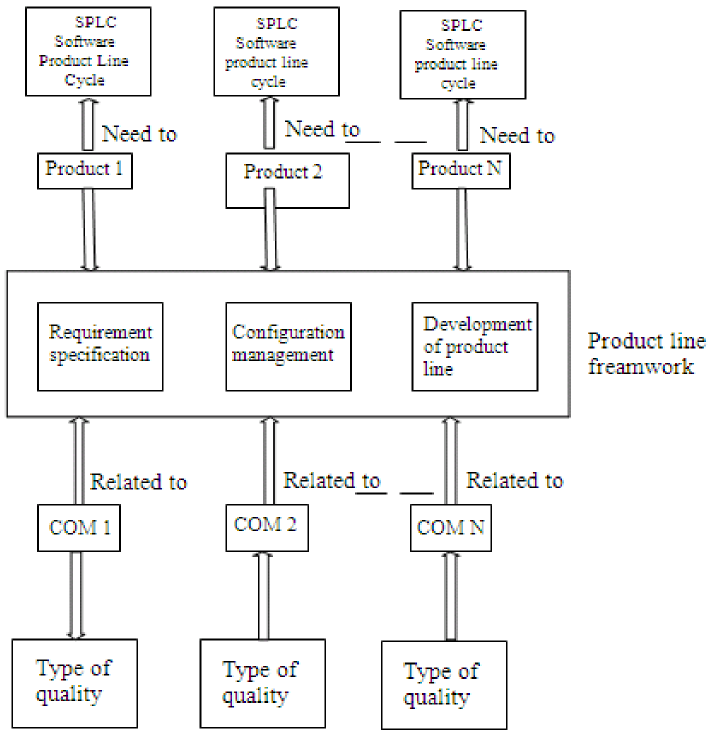

Fig. 3: Integrating components in SPL

DISCUSSION

From Fig. 3, we can remark the following points on using component based software development approach:
CBSD is the foundation for reusing

- It improves product quality
- It reduces development time
- It reduce time to market

So organizations whose want to build their products from selecting and integrating components will be able to meet their requirements only if they found sufficient number of components that satisfy specific quality requirements, because without this quality level the usage of components is not useful.

CONCLUSION

A quality is a main goal to every business, this study is considerate on achieving high level of quality by integrating components in products; this become the more effective way to achieve quality goal level by choosing the most appropriate components to specific products. The perspective of this type of the architecture is reducing costs and reducing time to market by the property of reusing that components have and the most important thing is achieving high level of quality.

REFRENCES

Ab-Rahman, M.S., S.A.C. Aziz and K. Jumari, 2009. Protection for an immediate split structure of tree-based epon architecture-ideal condition analysis. Am. J. Eng. Applied Sci., 2: 372-380. DOI: 10.3844/ajeassp.2009.372.380

Amin, F., A.K. Mahmood and A. Oxley, 2011. Review of software component Reusability Assesment Approaches. Res. J. Inform.. Technol., 3: 1-11. DOI:10.3923/rjit.2011.1.11; http://docsdrive.com/pdfs/academicjournals/rjit/2011/1-11.pdf

Amin, F., Mahmood and A. Oxley, 2010. A review on aspect oriented implementation of software product lines components. Inform. Technol. J., 9: 1262-1269. **DOI:** 10.3923/itj.2010.1262.1269; http://webcache.googleusercontent.com/search?q=cache:53qqHiWXlpwJ:scialert.net/abstract/index.php%3Fdoi%3Ditj.2010.1262.1269+A+review+on+aspect+oriented+implementation+of+software+product+lines+components&cd=2&hl=en&ct=clnk&gl=pk&source=www.google.com.pk

Kaur, P. and H. Singh, 2010. Metrics Suite for Component Versioning Control Mechanism in Component based Systems. Journal of software engineering, 4: 231-243. http://webcache.googleusercontent.com/search?q=cache:RDgJ7_zvR24J:ajinc.net/abstract/%3Fdoi%3Djse.2010.231.243+Metrics+Suite+for+Component+Versioning+Control+Mechanism+in+Component+based+Systems&cd=2&hl=en&ct=clnk&gl=pk&source=www.google.com.pk

Khaled, L., 2010. Driving architectural design through business goals. Int. J. Comput. Sci. Inform. Sec.. 8: 68-71. http://www.doaj.org/doaj?func=abstract&id=563920

Omar, M., S. Lailee and A. Yasin, 2011.The The impact of agile approach on software engineering teams. Am. J. Econ. Bus. Admin., 3: 12-17. DOI: 10.3844/ajebasp.2011.12.17

Sharp, J. and S. Ryan, 2010. A theoretical framework of component-based software development phases. ACM SIGMIS Database, 41: 56-75. DOI: 10.1145/1719051.1719055