International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

Architectural Solution of Time Management System in Test Driven Development Approach

Olena Kryvoruchko[1], Mykhailo Kostiuk[2], Mykola Tsiutsiura[3]

[1]Kyiv National University of Trade and Economics, 19, Kyoto St., Kyiv 02156, Ukraine

[2]Programmer, Kyiv National University of Trade and Economics, 19, Kyoto St., Kyiv 02156, Ukraine

[3]Kyiv National University of Construction and Architecture, 31, Povitroflotskiy Ave, Kyiv 03680, Ukraine

Abstract*: The purpose of SOA is to allow users to combine together fairly large chunks of functionality to form ad hoc applications built almost entirely from existing software services. The larger the chunks, the fewer the interfaces required to implement any given set of functionality; however, very large chunks of functionality may not prove sufficiently granular for easy reuse. Each interface brings with it some amount of processing overhead, so there is a performance consideration in choosing the granularity of services*.

Keywords: develop Time Management System, Service Oriented Architecture (SOA) and N-tired Architecture, used Test Driven Development (TDD)

1. Introduction

Information technology rapidly developed and implemented in various fields of human activity to solve different problems. In this regard, there are needs for new programming specialists.

In modern life has become a profession programmer high demand and this contributes to the popularity and scope of information technology.

Today, for many organizations that are engaged in software development, there is a need in correct distribution of tasks among developers, time tracking of their realization, and monitoring of the implementation of these tasks. In order to solve such problems organization, need to develop Time Management System. However, choosing of the proper application architecture, as well as the proper development methodology plays a very important role in the creation of information systems of such kind. In order to improve the quality of the created software, various technique and development technologies are used, but the choice of a particular technique and technology is directly dependent on the application.

2. The Main Research

In order to develop time management system must be taken into account several factors: who guided system, the importance of data security in the system, how need to be used the system, on which machines the system will operate etc. The choice of the architecture of such a system is directly dependent on these factors.

For systems of such level typically used several architectural patterns such as Service Oriented Architecture (SOA) and N-tired Architecture. A technique for developing such a system typically used Test Driven Development (TDD).

The use of architectural patterns Service Oriented Architecture (SOA) and N-tired Architecture will split the application into multiple parts, that is a good thing in the security of such informational systems, and to minimize the load of the server on which the system is deployed. The database system is on the one server, a web service that handles business logic is on the other, advanced web client – on the third server, thus preserving safety of a particular part of the system or data as long as possible at any level of attacks. Also, with today's computer hardware, a good practice is to implement all data manipulation on the web client directly from the user part (implementation of logic in javascript). It also allows to unload the server on which the system is deployed.

N-tier architecture is a client-server architecture in which presentation, application processing, and data management functions are physically separated. The most widespread use of multi-tier architecture is the three-tier architecture [1].

N-tier application architecture provides a model by which developers can create flexible and reusable applications. By segregating an application into tiers, developers acquire the option of modifying or adding a specific layer, instead of reworking the entire application. A three-tier architecture is typically composed of a presentation tier, a domain logic tier, and a data storage tier [1].

A service-oriented architecture (SOA) is an architectural pattern in which application components provide services to other components via a communications protocol, typically over a network. The principles of service-orientation are independent of any vendor, product or technology.

The purpose of SOA is to allow users to combine together fairly large chunks of functionality to form ad hoc applications built almost entirely from existing software services. The larger the chunks, the fewer the interfaces required to implement any given set of functionality; however, very large chunks of functionality may not prove

sufficiently granular for easy reuse. Each interface brings with it some amount of processing overhead, so there is a performance consideration in choosing the granularity of services.

SOA also provides connection for enterprise level systems, which contain one or many data sources and a big number of user-oriented applications.

SOA as an architecture relies on service-orientation as its fundamental design principle. If a service presents a simple interface that abstracts away its underlying complexity, then users can access independent services without knowledge of the service's platform implementation [2].

In order to provide data communications and processing of business logic for enterprise systems like TMS used Representational State Transfer (REST) services.

Representational State Transfer (REST) is a software architecture style consisting of guidelines and best practices for creating scalable web services. REST is a coordinated set of constraints applied to the design of components in a distributed hypermedia system that can lead to a more performant and maintainable architecture. RESTful systems typically, but not always, communicate over the Hypertext Transfer Protocol with the same HTTP verbs (GET, POST, PUT, DELETE, etc.) used by web browsers to retrieve web pages and send data to remote servers.

The main goal in the development of TMS is to create the high quality system that will help to resolve the problem of time management and task distribution between employees in software development organizations.

In addition, this software is only the part of system, which could be created for better time management for software development organizations.

Based on the data about the system in general and architectural patterns TMS will consist of the three separated servers. The design of the time management system architecture is shown on the Figure 1.

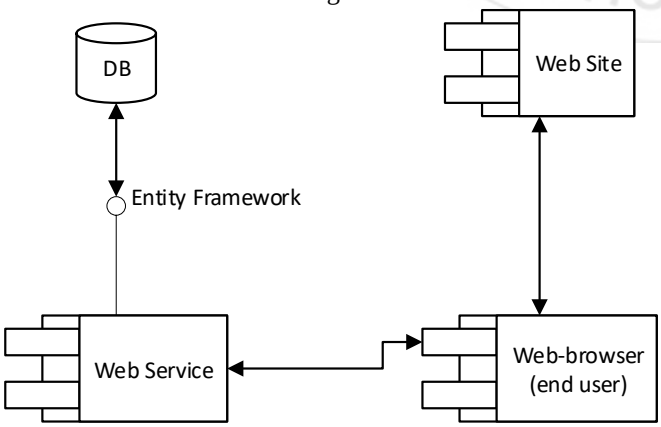

Figure 1: TMS architecture

Choosing of the right development technique is also important for TMS as enterprise level system. In order to ensure the quality of the developed enterprise level system the best way is to use Test Driven Development (TDD) technique.

This technique is a software development process that relies on the repetition of a very short development cycle: first, the developer writes an (initially failing) automated test case that defines a desired improvement or new function, then produces the minimum amount of code to pass that test, and finally refactors the new code to acceptable standards.

3. Conclusion

With regard to methodology development, TDD approach is the best choice in terms of quality assurance of time management system. At the heart of TDD lies development of test scenarios of methods (separate function), then write test covering the desired change, and then write code that will allow to pass the test, and finally refactor the new code to the appropriate standards. In such a way implements initial testing of the system, which in turn provides defining of bugs and defects on the earliest stages of the development process.

As a result of deep analysis and design were chosen two the most appropriate architectural patterns and TDD technique for the development of TMS as an enterprise level system. TDD technique makes it possible to increase the efficiency of development, thereby providing high quality of software with minimal black box testing. Using multiple architectural patterns, such as SOA and n-tier architecture, have made it possible to divide the system into three levels, allowing the system to distribute across multiple servers, thereby providing a distributed load on them

References

[1] [Multitier architecture - SI two, 2014. - 2 p. [Electronic Resource] – Access configuration: http://c2.com/cgi/wiki?MultiTierArchitecture
[2] Understanding Service-Oriented Architecture – Microsoft Development Network, 2015. - 20 p. [Electronic Resource] – Access configuration: https://msdn.microsoft.com/en-us/library/aa480021.aspx
[3] Innovative project and program management guidance, vol. 1, version 1.2 / Trans. from rus. Language under red. of S. D. Bushuev. - K .: Science. World, 2009. - 173 p.
[4] Ponomarenko L. A. Innovative computer management project technologies . - K .: Kyiv. Nat. Trade and Economy. University Press, 2001. - 423 p.
[5] Guidance for the Pantology of project management (guidance PMBOK®) Third edition, 2004 Project Management Institute, Four Campus Boulevard, Newtown Square, PA 19073-3299 USA / USA. - 401 p.
[6] S. Tsyutsyura Investment project management systems: Training. guidances. / S. V. Tsyutsyura, O. V. Kryvoruchko, M. I. Tsyutsyura // K .: KNUBA, 2013. - 152 p.

Author Profile

Prof. Olena Kryvoruchko received engineer-economist qualification in "economy and arrangement of groceries industry" from Kyiv Technological Institute of Food Industry in 1991. Since 1991 worked in the leading campaigns of Ukraine in the area of food industry as an economist. Since 2000 she works in Kyiv National University of Trade and Economics. She received Ph.D. degree in information technology in 2003 and she got a scientific rank of associate professor of economic cybernetics and information systems department in 2008. She is a doctor of engineering sciences in 2015. Presently she is Head of Department of the programmatic engineering and informative systems in the Kyiv National University of Trade and Economics.

Mr. Mykhailo Kostiuk Made off the National Aviation University in 2017 on speciality the "Programmatic engineering". He is a Master of Programmatic engineering. Since 2016 working Engineer by a programmer on the department of the programmatic engineering and informative systems systems in the Kyiv National University of Trade and Economics.

Mr. Mykola Tsiutsiura, Since 2012 is master in "Information control systems and technologies." Since 2013 is master in "Project Management". Since November 2012 is a graduate student of the Kiev National University of Construction and Architecture in the direction of information technology and project management. He received Ph.D. degree in information technology in 2015. Since 2013 is assistant professor Information Technologies department in Kyiv National University of Construction and Architecture, Ukraine.