ARCHITECTURE DEVELOPMENT OF WEB BASED APPLICATION FOR A CONSTRUCTION COST PREDICTION

KANIK, Tomasz, (PL)

Abstract. This paper describes the idea and first steps of development of intended commercial application for construction cost prediction. Study of different architecture models and development techniques is described at the very beginning. Then pros and cons of OSS and commercial tools are analysed. Based on that the future shape of project is envisioned and described up to current state. Server-based environment solutions are described too.

1 Overview of main requirements for an application

It should provide construction cost pre-calculation for non-professional builder. During building process real cost calculation is performed by systematically filling real price cost according to created model. It is based on free web technologies. Interface is user friendly. A hierarchical structure of aggregate items of construction allow different, independent price range calculation. Availability of tender creation for multiple group of construction items and their distribution by different channels – for example by email. Summarized report based on available data.

2 Establishing of the overall structure of an application

Architectural design represents an early stage of the system design process. It represents the link between specification and design processes, often carried out in parallel with some specification activities, which involve identifying of major system components and their communications. Right architecture design prevents to disestablish the system, or to reduce it efficiency by dynamic specification changes [1].

Design process has few, practical rules. First is to localize operations to minimise sub-system communication and increase system performance. Second is to use a layered

architecture with critical assets in inner layers. Third, critical component isolation. The last, fourth, is to include redundant and self-contained components in the architecture for better maintainability [2].

The architectural model of a system may conform to a generic architectural model or style. An awareness of these styles can simplify the problem of defining system architectures. However, most of large systems are heterogeneous and do not follow a single architectural style.

Analysis of different model characteristic show that three-tier model is the best fit for our requirements. Three-tier is a client–server architecture in which the user interface, functional process logic ("business rules"), computer data storage and data access are developed and maintained as independent modules, most often on separate platforms. Apart from the usual advantages of modular software with well-defined interfaces, the three-tier architecture is intended to allow any of the three tiers to be upgraded or replaced independently as requirements or technology change. For example, a change of operating system in the presentation tier would only affect the user interface code. Idea of the three-tier model is shown in Fig.1.

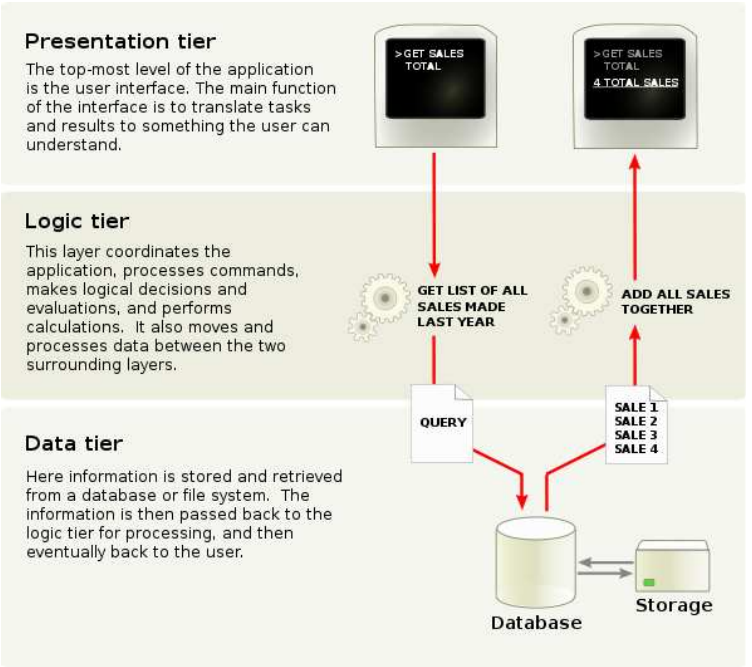

Figure 1: Visual overview of a Three-tiered application [2]

3 Specific application development methodology

The methodology choice we made is fundamentally for future model development by succeeded team. That is why we performed robust sensitivity analysis, which show that the right choice is Incremental Model. It is most appropriate, because requirements for future development are not well understood and expectations rapidly changing.

The Incremental model is the combination of linear and iterative methodologies with the primary objective of each being to reduce inherent project risk by breaking a project into smaller segments and providing more ease-of-change during the development process. Our idea is that the initial software concept, requirements analysis, design of architecture and system core are defined using the waterfall approach, followed by iterative prototyping, which culminates in implementation of the final prototype (i.e., working system) [3].

Strengths of this solution: Potential exists for exploiting knowledge gained in an early increment as later increments are developed. Moderate control is maintained over the life of the project through the use of written documentation and the formal review and approval/sign-off by the user and information technology management at designated major milestones. Stakeholders can be given concrete evidence of project status throughout the life cycle. Helps to mitigate integration and architectural risks earlier in the project. Allows delivery of a series of implementations that are gradually more complete and can go into production more quickly as incremental releases. Gradual implementation provides the ability to monitor the effect of incremental changes, isolate issues and make adjustments before the organization is negatively impacted [4].

Weaknesses found : When utilizing a series of mini-Waterfalls for a small part of the system before moving on to the next increment, there is usually a lack of overall consideration of the business problem and technical requirements for the overall system. Since some modules will be completed much earlier than others, well-defined interfaces are required. Difficult problems tend to be pushed to the future to demonstrate early success to management [4].

4 Software modelling tool for creating the application architecture

It sounds funny or silly for someone who never created a large project, that it is not so easy to find good tool with support of UML2, MDA or BPMN models. Open source solution mostly don't cover any of above standards at all. Simple applications created by different teams, groups or people are good for learning and can cover requirements of semester project, but it is all. That is why, after long search and many tries our project architecture was created using commercial solution of IBM company - Rational Software Modeler. Experience we gather is expressed bellow, in description of each software we use.

Dia is a free and open source general-purpose diagramming software. It uses a controlled

single document interface similar to GIMP. Dia has a modular design with several shape packages available for different needs: flowchart, UML, network diagrams, circuit diagrams, and more. It does not restrict symbols and connectors from various categories from being placed together [5].

It is really smart and useful tool when you need to draw overhead diagram, but drawing large amount of UML diagrams with it is road across hell. There is no project which glues different diagrams together and no creator or hot key for frequently used operations is available.

Umbrello UML Modeller is a free software UML diagram application available natively for Unix. It handles all the standard UML diagram types. It can reverse engineer code written in C++, IDL, Pascal/Delphi, Ada, Python, and Java, as well as import XMI files generated by external tools and export to various programming languages. It allows simple collaborative development by exporting to DocBook and XHTML formats [6].

An idea of creating code from diagrams or back to them sounds good but reality shows it is still under development, moreover in non-native environment (non-KDE) the application freezes few times and after process was killed, the project had broken structure. Some problems with grouping of objects and changes of interaction between diagram occurred.

StarUML is an open source UML tool. The stated goal of the project was to replace larger, commercial applications such as Rational Rose and Borland's Together. Now it is no longer under development. StarUML was written in Delphi, which is one of the reasons why it is no longer maintained [7].

This tool nearly meets our requirements, except it is currently missing object, package, timing and interaction overview diagrams. It is huge mistake that project was canceled and no successor seems to appear. We were positively surprised that manipulation with diagrams was really good and number of properties allow to specify exactly what we want. Maybe some changes in IDE can be done, which increase ergonomic model space use.

Enterprise Architect, (EA), commercial software produced by Sparx Systems, is a collaborative modelling, design and management platform based on UML 2.1 and related standards (i.e. BPMN, XMI). Agile, intuitive and extensible with fully integrated, powerful domain specific high-end features at a fraction of the cost of many competitors. An enterprise wide solution for visualizing, analysing, modelling, testing and maintaining a wide range of systems, software, processes and architectures [8].

Definitely it is not intuitive software, despite of the claims on EA homepage. It is a mess of creators and views. It takes a while to prepare environment to do exactly what we want. Maybe it is good that so many futures are provided but it makes this software less practical, no one looks from the viewpoint of designer who has no time to try gadgets but have a lot of work to do. Object manipulation and diagrams interaction are not so good as it should be.

IBM Rational Software Modeller (RSM) made by IBM's Rational Software division, is a UML 2.0-based visual modelling and design tool. Rational Software Modeller is built on the Eclipse open-source software framework and is used for visual modelling and model-

driven development (MDD) with UML for creating applications and web services. RSM is engineered as a plugin that sits on top of the open-source Eclipse development platform [9].

There are not many application I can say are nearly ideal, but RSM and their advanced version RSA (IBM Rational Software Architect) looks like dream of all software architects. It is multi-platform software, has integrated most of commonly used modelling standards, has ergonomic modelling space, has good integration between different diagrams, allows describe model in amount of properties exactly as it is needed. Using this software is a pleasure, our project is well documented and every time we need to check or specify some details we can easily find it.

5 Choosing programming language

Language choice is a topic filled with opinions. Most programmers have a favourite language, and most programmers have languages they hate. Programming in a given language engenders similar responses to doing plumbing repairs with a given box of tools. In general the goal of computing should be to produce the most powerful and reliable tools possible with the least amount of effort and resources consumed. That is why our choice of language is made based on overall goals and requires various language to be reviewed. Bellow, popular languages we consider to use are reviewed in few words.

Java is a multi-platform language that is especially useful in networking. Of course, the most famous usage of Java is on the web, with Java applets, but Java is also used to build stand-alone cross-platform programs. Since it resembles C++ in syntax and structure, learning Java is usually quite easy for most C++ programmers. Java offers the advantages provided by object-oriented programming, such as reusability; on the other hand, it can be difficult to write highly efficient code in Java, and Swing, its primary user interface, is notoriously slow. Nevertheless, Java has increased in speed in recent years, and version 1.5 offers some new features for making programming easier [10].

PHP is a common language for website design that is sometimes used as a scripting language in *nix. PHP is designed for rapid website development, and as a result it contains features that make it easy to link to databases, generate HTTP headers, and so forth. As a scripting language, it contains a relatively simple set of basic components that allow the programmer to quickly get up to speed, though it does not have more sophisticated object-oriented features [10].

C♯ is a multi-paradigm programming language encompassing imperative, functional, generic, object-oriented (class-based), and component-oriented programming disciplines. It is one of the programming languages designed for the Common Language Infrastructure. By design, C♯ is the programming language that most directly reflects the underlying Common Language Infrastructure (CLI). Most of its intrinsic types correspond to value-types implemented by the CLI framework. However, the language specification does not state the code generation requirements of the compiler: that is, it does not state that a C♯ compiler must target a Common Language Runtime, or generate Common Intermediate Language (CIL),

or generate any other specific format. Theoretically, a C♯ compiler could generate machine code like traditional compilers of C++ or FORTRAN [11].

Ruby is a dynamic, reflective, general purpose object-oriented programming language that combines syntax inspired by Perl with Smalltalk-like features. It was influenced primarily by Perl, Smalltalk, Eiffel, and Lisp. Ruby supports multiple programming paradigms, including functional, object oriented, imperative and reflective. It also has a dynamic type system and automatic memory management; it is therefore similar in varying respects to Python, Perl, Lisp, Dylan, Pike, and CLU [12].

Python is a general-purpose high-level programming language whose design philosophy emphasizes code readability. Python aims to combine "remarkable power with very clear syntax", and its standard library is large and comprehensive. Its use of indentation for block delimiters is unusual among popular programming languages. Python supports multiple programming paradigms, primarily but not limited to object oriented, imperative and, to a lesser extent, functional programming styles. It features a fully dynamic type system and automatic memory management, similar to that of Scheme, Ruby, Perl, and Tcl. Like other dynamic languages, Python is often used as a scripting language, but is also used in a wide range of non-scripting contexts [13].

Due to the use of multi-tier model with not only web pages but back-end system too, we decide to use Java language as the one, which meet our requirements. It is important for us to choose language, which is not on margin or end-route of their life cycle – future perspective is of the same importance as today ability the language has. For now, Apache Wicket is used as web framework and support of Hibernate Technologies for base tier is planning. Implementation of other/new framework or technologies will depend on future requests or needs.

6 Choosing database engine

Project specification says that application have to be based on free technologies, it means that no commercial database engine can be used. That is, why there is no commercial engine description. Language chosen for project development (Java) allows to use any kind of currently known database, so we are not obliged to make choice based on language restrictions. Some of database engines, we consider to use, are described bellow.

HSQLDB (Hyper Structured Query Language Database) is a relational database management system written in Java. It has a JDBC driver and supports a large subset of SQL-92, SQL-99, and SQL:2003 standards. It offers a fast, small (around 600 kilobytes in the standard version) database engine which offers both in-memory and disk-based tables. Embedded and server modes are available [14].

MySQL is a relational database management system (RDBMS) that runs as a server providing multi-user access to a number of databases. It is currently the most popular open source database server in existence. On top of that, it is very commonly used in conjunction with PHP scripts to create powerful and dynamic server-side applications. MySQL has

been criticized in the past for not supporting all the features of other popular and more expensive DataBase Management Systems. However, MySQL continues to improve with each release, and it has become widely popular with individuals and businesses of many different sizes [15].

Firebird is a relational database offering many ANSI SQL standard features that runs on Linux, Windows, and a variety of Unix platforms. Firebird offers excellent concurrency, high performance, and powerful language support for stored procedures and triggers. The Firebird Project is a commercially independent project of C and C++ programmers, technical advisers and supporters developing and enhancing a multi-platform relational database management system based on the source code released by Inprise Corp (now known as Borland Software Corp) [16].

PostgreSQL is a powerful, open source object-relational database system. It has architecture that has earned it a strong reputation for reliability, data integrity, and correctness. It runs on all major operating systems, including Linux, UNIX, and Windows. It is fully ACID compliant, has full support for foreign keys, joins, views, triggers, and stored procedures (in multiple languages). It includes most of SQL:2008 data types. It also supports storage of binary large objects, including pictures, sounds, or video. It has native programming interfaces for C/C++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, among others, and exceptional documentation [17].

A database engine we are looking for will have long term influence on future of the project, even if database independent technology, like Hibernate, is used. HSQL is the database for development, tests, single client solution and generally small projects. Last few years Firebird is used in large governments project, mainly for it is performance, but it have weaker support from community. MySQL, well known database for PHP and other applications installed on LAMP servers, has good support and years of development. Even considering that all, we feel that it is still not what we looking for. There is lot of non standard functions, not so good performance and scalability and not enough security too. At the end we realize it, our choice should be PostgreSQL. It is database engine for large and critical projects in government and commercial solutions, many times compared to Oracle or DB2. PostgreSQL has very good performance with large amount of data, has mechanisms to maintain the database and is prepared for long up-time work.

7 J2EE application server

GlassFish is an open source application server project led by Sun Microsystems (Oracle now) for the Java EE platform. The proprietary version is called Sun GlassFish Enterprise Server. GlassFish is free software, dual-licensed under two free software licences: the Common Development and Distribution License (CDDL) and the GNU General Public License (GPL) with the classpath exception. GlassFish is based on source code released by Sun and Oracle Corporation's TopLink persistence system. It uses a derivative of Apache Tomcat as the

servlet container for serving Web content, with an added component called Grizzly which uses Java NIO for scalability and speed [18].

JBoss Application Server (or JBoss AS) is free software/open-source Java EE-based application server. Because it is Java-based, the JBoss application server operates cross-platform: it is usable on any operating system that supports Java. JBoss AS was developed by JBoss, now a division of Red Hat [19].

Apache Tomcat is an open source servlet container developed by the Apache Software Foundation (ASF). Tomcat implements the Java Servlet and the JavaServer Pages (JSP) specifications from Sun Microsystems, and provides a "pure Java" HTTP web server environment for Java code to run. Apache Tomcat powers numerous large-scale, mission-critical web applications across a diverse range of industries and organizations [20].

No special requirements are formed for application server for now, because it can be changed without influence on most of the project. We would like to create one EAR file with our project and make it available on all application servers, which support this file format. For development environment Apache TomCat is chosen for it's simple configuration, management and good support from their community.

8 Development IDE

There is no specification or recommendation for development environment. Only two most popular IDE are presented, because the use of other environments is imperceptible. Someone can say that Visual Studio is recently used to develop Java programs, and it is true, but not for web-based development. That small difference cause that only two, well known IDE's, are used in real development. It is thanks to provided support and functionality, which help programmers to make their job and not to think about technical dependencies.

NetBeans refers to both a platform framework for Java desktop applications, and an integrated development environment (IDE) for developing with Java, JavaScript, PHP, Python, Ruby, Groovy, C, C++, Scala, Clojure, and others (for a complete overview, visit the netbeans website). The NetBeans IDE is written in Java and runs everywhere where a JVM is installed, including Windows, Mac OS, Linux, and Solaris. A JDK is required for Java development functionality, but it is not required for development in other programming languages. The NetBeans Platform allows applications to be developed from a set of modular software components called modules. Applications based on the NetBeans platform (including the NetBeans IDE) can be extended by third party developers [21].

Eclipse is a multi-language software development environment comprising an integrated development environment (IDE) and an extensible plug-in system. It is written primarily in Java and can be used to develop applications in Java and, by means of various plug-ins, other languages including C, C++, COBOL, Python, Perl and PHP. The IDE is often called Eclipse ADT for Ada, Eclipse CDT for C/C++, Eclipse JDT for Java and Eclipse PDT for PHP. The initial codebase originated from VisualAge. In its default form it is meant for Java developers, consisting of the Java Development Tools (JDT). Users can extend

its capabilities by installing plug-ins written for the Eclipse software framework, such as development toolkits for other programming languages, and can write and contribute their own plug-in modules. Released under the terms of the Eclipse Public License, Eclipse is free and open source software [22].

Our team chosen Eclipse as development environment used for this project. This decision is among based on their support for advanced SVN plugins, Ant building script, remote library import and good integration with development application server selected before. Moreover Rational Software Modeler is based on Eclipse, thanks to this our team has good, past experience with that environment.

9 Current state of project and development

After all preparation we created well documented architecture model. Based on that model, development work is started and first prototype is being prepared. Simulation of production environment is created based on Debian system and TomCat application server. There is remote access enabled to this machine for any team member – it is for test, how developed programs behave in real simulated environment. Source code is stored in SVN repository, team members IDE's are prepared to remotely communicates with it. This works as back-up and helps to keep source updated during different modules development process. Simple database model is kept on the server, to help the members to provide tests on their local machines too, thanks to VPN tunnelling.

10 Conclusions

All steps described before show how to make reliable and well-considered decision on large scale project. Few of this step seems to be not important at the beginning but in the future can cause large changes, even cause the project rebuild. Technologies in use today may change tomorrow, so we have to provide mechanisms to implement any of them, even if we do not have their specification now. This article is not the receipt for a good project, it just puts lights on tools and technologies available today but which may not be well recognised.

Literatúra

[1] SOMMERVILLE I.: *lectures of Software Engineering*, University of Southern California, 2008

[2] Wikipedia, `http://en.wikipedia.org/wiki/Multitier_architecture`, 2010-06-06

[3] Wikipedia, `http://en.wikipedia.org/wiki/Software_development_methodologies`, 2010-06-06

[4] Centers for medicare and medicaid services: *Selecting a development approach*, Washington, Rev. 2008-03-27

[5] Wikipedia, http://en.wikipedia.org/wiki/Dia_(software), 2010-06-06

[6] Wikipedia, http://en.wikipedia.org/wiki/Umbrello_UML_Modeller, 2010-06-06]

[7] Wikipedia, http://en.wikipedia.org/wiki/StarUML, 2010-06-06]

[8] Sparx Systems Pty Ltd, http://www.sparxsystems.com/products/ea/features.html, 2010-06-06

[9] Wikipedia, http://en.wikipedia.org/wiki/IBM_Rational_Software_Modeler, 2010-06-06

[10] SUH E., *The Tower of Babel – An Exploration of Programming Languages*, http://www.cprogramming.com/langs.html, 2010-06-07

[11] Wikipedia, http://en.wikipedia.org/wiki/C_Sharp_(programming_language), 2010-06-07

[12] Wikipedia, http://en.wikipedia.org/wiki/Ruby_(programming_language), 2010-06-07

[13] Wikipedia, http://en.wikipedia.org/wiki/Python_(programming_language), 2010-06-07

[14] Wikipedia, http://en.wikipedia.org/wiki/HSQLDB, 2010-06-08

[15] Erack Network, http://www.tizag.com/mysqlTutorial/, 2010-06-08

[16] Firebird Foundation Incorporated, http://www.firebirdsql.org, 2010-06-08

[17] PostgreSQL Global Development Group, http://www.postgresql.org, 2010-06-08

[18] Oracle Corporation, http://www.glassfish.org, 2010-06-08

[19] Red Hat, Inc., http://www.jboss.com, 2010-06-08

[20] The Apache Software Foundation, http://tomcat.apache.org, 2010-06-08

[21] Oracle Corporation, http://netbeans.org, 2010-06-08

[22] The Eclipse Foundation, http://www.eclipse.org/resources/, 2010-06-08

Contact

Tomasz KANIK (Bc),
Fakulta riadenia a informatiky,
Univerzitná 8215/1, 010 26 Žilina,
tkanik@o2.pl

Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
1.–4. júla 2010, Žilina, Slovensko

Organizátori: Miloš Šrámek, Spoločnosť pre otvorené informačné technológie
Tatiana Šrámková, Katedra fyziky, FEI STU Bratislava
Michal Kaukič, Aleš Kozubík, Tomáš Majer, Žilinská univerzita
Lýdia Gábrišová, Ľubica Michálková, Žilinská univerzita
Juraj Bednár, Digmia, Slovensko
Miloslav Ofúkaný, GeoCommunity, Slovensko
Peter Mráz, Kremnica
Slavko Fedorik, SOŠ elektrotechnická, Poprad
Peter Štrba, Spojená škola/Gymnázium M. Galandu, Turčianske Teplice
Ladislav Ševčovič, FEI, Technická univerzita v Košiciach

Editori: Michal Kaukič
Miloš Šrámek
Slavko Fedorik
Ladislav Ševčovič

Recenzenti: Mgr. Juraj Bednár
Mgr. Rudolf Blaško, PhD.
RNDr. Ján Buša, CSc.
Ing. Slavko Fedorik
Ing. Karol Grondžák, PhD.
Mgr. Michal Kaukič, CSc.
Ing. Tomáš Kliment
RNDr. Aleš Kozubík, PhD.
Mgr. Juraj Michálek
doc. RNDr. Štefan Peško, CSc.
Ing. Pavel Stříž, PhD.
RNDr. Ladislav Ševčovič
Ing. Michal Žarnay, PhD.

Vydavateľ: Spoločnosť pre otvorené informačné technológie – SOIT, Bratislava

ISBN 978-80-970457-0-8

Sadzba programom pdfTEX Ladislav Ševčovič
