MIPRO 2016, May 30 - June 3, 2016, Opatija, Croatia

Visual representation of predictions in software development based on software metrics history data

Blažo Popović*, Adis Balota * and Dženan Strujić **
* Faculty for Information Technology, University Mediterranean, Podgorica, Montenegro
** Nilex Development, Nilex AB I Helsingborg, Helsingborg, Sweden
popovic.blazo@gmail.com, adis.balota@unimediteran.net, dzenan.strujic@nilex.se

Abstract - Software who is being developed for more than a year, is categorized in the group of large projects, and is entering the area of critical and risky in terms of their successful completion. Because of those reasons, they require constant monitoring by the software manager. In order to perform adequate monitoring of developed solution and its projects, there are a number of software metrics that provide these information in numerical form, but very few of them is displayed to software manager in visual form. Predictions for future solution development, and its visual representation, based on historical data gathered from the same system, are usually not included in such systems. These kind of systems must have the ability to predict state of the developed system in the future, and adapt its output upon having analyzed his software metrics history data. In this paper, we propose a intelligent system that will analyze software metrics history data of the solution, and make predictions and visual outputs to support software manager decisions. The results show that using this decision support tool, as a form of intelligent system, helps software managers in their decision making during project management, and in reducing overall project risk.

I. INTRODUCTION

One of the most important tasks of the software manager in project management is risk management throughout all stages of its development [1, 2]. In order to successfully monitor the risk in all individual phases of the software product development it is necessary to have appropriate measures, in respect of which it is possible to quantify the size of the risk [1, 2]. These measures are in software engineering reflected in the form of software metrics.

In today's software engineering, there are a number of development tools and environments that are used for software design and software metrics presentation [5, 17-20]. Those tools helps the software manager significantly in project management. Graphic representation and visualization of software metrics parameters offers the possibility that large amount of numbers that is not in systematic form, become shaped, categorized and graphically or visually expressed.

With development of tools for graphical representation of software metrics parameters, there is a possibility for more efficient risk management and control of software projects [2]. Previous approaches to solving the aforementioned problems give limited results [8-16], and that fact requires that software managers use one or more tools for software metrics representation and analysis [2], in order to see the state of the developed software code. In addition, these tools and models do not offer fully graphical interpretation of its numbers in a way that meets the demand of software managers in real-time.

Recognizing the need of software manager in real-time for risk management in project they lead, there is an open possibility to create decision support system in form of an prototype, intelligent rule-based expert system. This intelligent system will have the ability to analyze software metric history data, represent them in the visual form, and make predictions for future software development based on this data. With this system that analyses history software metric data gathered from the system that is being developed, software managers will have opportunity to anticipate and cope with the risk they face, and more easily predict future software development on the project based on it.

Paper has the following structure. In section II, we will give an overview of software metrics in enterprise systems, and introduce the metrics that will be monitored, collected and used in proposed rule-based expert system. Section III will make an overview of visualization techniques used for graphical representation of data, and give the details of the visualization tool that we will use in synergy with our rule-based expert system. In section IV we will introduce prototype of proposed rule-based expert system, its architecture, knowledge base rules, facts, inference engine, explanation facility, with its prediction and visualization capabilities. Section V will give the discussion of the proposed solution, results, and limitation of the same. Paper ends with conclusion in section VI.

II. SOFTWARE METRICS IN ENTERPRISE SYSTEMS

Software metrics is of great importance in the life of development of a software product. Using software metrics in various stages of development of software product can provide a clear picture to the software manager about state and condition of project that he leads. Additionally, software metrics provides information about software product, process, what kind of effort is invested, what effort should be made to complete the launched project or product, what is the quality of code that developers are producing, and finally, he gets the information needed to guide the process of software testing.

Software metrics must be used in project management so that it can be properly performed, and because of risk that he wears with itself. Appropriate software metrics will provide inputs to software managers to make decisions about how to further advance the development of software products. There are a large number of categories in which one can classify software metrics. One of these categorizations is done in [1, 2].

Two basic software metrics are SLOC and function points. The former, in its purist form, counts the number of the typed code lines, and later represents functional user requirements in numerical form as function points [3]. In enterprise projects, these metrics is not suitable because software is written in object oriented manner. Because of that fact, appropriate object oriented software metrics has to be employed. The most widely used set of object oriented measures is proposed by Chidamber and Kemerer [2, 4], and those measures include: WMC (weighted methods per class), DIT (depth of inheritance tree), NOC (number of children), CBO (coupling between object classes) and RFC (response for a class).

Since focus of our paper are enterprise projects developed in Microsoft Visual Studio 2013 development environment, we will observe enterprise metrics that is produced by its plug in Code Metrics Viewer [5]. Code Metrics Viewer offers following metrics to developers and software managers: maintainability index (MI), cyclomatic complexity (CC), depth of inheritance (DIT), class coupling (CBO), and lines of code (LOC). Their values and interpretation can be found in [6].

Following the good software manager practice, every software managers want their developers to produce code following their established coding standard, that will guarantee high-quality software components, from methods to custom types. In order to establish such routine and discipline, Code Metrics Viewer gives to developers the possibility to measure their code quality, so that software manager is satisfied with code quality and coding standard. This is also true for software manager, who can view, collect and analyze software metrics data from projects and solution in numerical form. In large enterprise projects, number of produced code lines grows constantly. Number of methods per project and per solution, including number of developed types by project and per solution grows day by day. In our previous work [7], we have demonstrated that growth of the methods and types per project and per solution in enterprise development does not increases its complexity, does not make it more difficult for maintenance, testing, or refactoring. This was all true if software manager is aware of the need for employment of adequate software metric and need for established coding standard for developers. Rules in coding standard employed MI, CC and LOC values of software metric that could not be broken by developers during software development.

However, today software metric tools do not provide means to try to predict the development process based on collected software metrics history data. This is the first important fact in which lies the idea, and basis for building for our prototype intelligent rule-based expert system that we will propose in section IV.

III. SOFTWARE VISUALIZATION TECHNIQUES

The field of software visualization is very broad. Visualization of software has found its application in everyday software engineering, various algorithm creation and representation, presentation of object-oriented code, software diagnostic, just to name a few.

Software visualization has advanced from two-dimensional graphics, over three-dimensional representation [8], up to virtual environments [9]. The biggest challenge in the field of software visualization is how to represent object oriented way of thinking, that is, how to represent classes, objects, interfaces, inheritance, polymorphism and other object oriented pillars. For this particular matter, there has been a significant amount of research and propositions in the science filed, and some of the ideas can be observed in [8, 10-16]. The bottom line of all this work is to find a way to represent software in visual way, so that software managers can visually analyze the state of the developed software product, and the quality of the produced software code.

Following the science field in software visualization, the software industry and practice developed tools for representing software visualization. In the domain of the C# programing language and its natural development environment Microsoft Visual Studio 2013, these tools include: Atomique [17], Nitriq [18], CodeIt.Right [19], NDepend [20]. All these tools have their own way of analyzing software code based on software metrics, and represent it in a visual or graphical form. However, it has been noticed that none of them includes any kind of prediction module for future software development in the developed solution.

Further enhancement and help in graphical representation and visualization on web sites and web pages has been brought by several graphical libraries. These libraries include: D3.js [21], FusionCharts.js [22], jqPlot [23], Google Charts [24], and Highcharts [25], just to name a few. In our proposed solution, we will use Highcharts library for drawing graphics. All mentioned libraries have the ability to draw different kinds of charts to web pages, tablets or mobile phones. These charts include: line charts, bar charts, area charts, column charts, pie charts, scatter charts, bubble charts, synchronized charts, dynamic charts, 3D charts, polar charts, spider web charts, wind rose charts, box plot charts, waterfall charts, gauges, and in particular libraries, heat and tree maps. Above listing of options in libraries makes them the perfect ground for use in visual representation of software metrics history data.

It can be brought to notice that all of these software tools and products do not offer the possibility to try to visualize the predictions of future software development based on software metrics history data. This is the second important fact for building our prototype intelligent rule-based system.

IV. PROPOSED SOLUTION

In our previous work [7], we demonstrated that usage of software metrics in enterprise development is of crucial importance in success of leading, monitoring and evaluating quality in software development, which in turn gives the software manager the precise control of the developed project and solution in whole. Further more, since software metrics gives information what is going on with the developed system, it of great importance for software manager to collect it, analyze it, and make business decisions after that. In the same work, we proposed software metrics that had to be used in the developed systems, which represented the development standards that had to be satisfied by all developers. After collecting software metrics history data from the system, we made the analysis and proved that development standards were respected by the developers, and there was no increase of complexity of the system aside from fact that solution was growing day by day in size of number of new methods and types per project, and per whole solution. The analysis in our work [7] did not include any kind of future software development predictions or recommendations.

The solution in question was developed by 50 developers that were spread all over the world. The leading software manager was not physically present all the time in person for questions and answers about future development decisions and predictions for all other development team leaders, who were leaning on his decisions between development iterations. Since the solution has been developed for more than a one year, it has grown so large in size that controlling and monitoring of the solution had to be delegated to other team leaders, aside from leading software manager. In order to do that, the expertise and experience of the leading software manager had to be caught in some form so that it could provide answers and guidelines to developers team leaders regarding future software development. The solution was found in our proposition to develop of prototype rule-based expert system, who had to catch and represent the logic of human expert, and help in decision making for other team leaders. Since the developers were geographically dispersed, we proposed that the system had to be a web based solution, so that every team leader could easily connect online on system and find the answers that he needed.

After narrowing the problem domain, the requirements for the expert system were elicitated. The proposed prototype of our intelligent system had to have ability to give state of the developed projects and overall solution based on collected software metrics history data. After making the analysis of software metrics history data, they had to be visualized on the web page with some form of visualization technique, where we proposed Highcharts library. The proposed system have to make predictions for next day based on performed analysis on software metrics history data, and number of days before current iteration ends. Prediction of the software development for the next day had to be represented in visual form so that team leader can visually conclude what will probably happened in tomorrow software development, without the need to check the numbers by hand. Proposed intelligent system must have the ability to explain how it has reached its decision on demand by team leader. Ultimately, the proposed expert system must give the visual output to team leader about current level of the risk in developed project in question, after making the predictions for software development.

After analyzing all the requirements for the intelligent system, and knowing the fact that most of the developers in company are using programing language C# to develop the solution, we proposed a solution that is based on Microsoft technologies. Development environment is Microsoft Visual Studio 2013. Since developed solution had to be a web application, it is based on ASP.NET MVC 4.5 and ASP.NET WEB API 2 frameworks. Database system for the intelligent system will be Microsoft SQL Server 2012 Express Edition. Our choice to go for ASP.NET WEB API 2 framework is made from gathered requirements for intelligent system, and its capabilities for creating RESTfull web services in easy manner, using ordinary http verbs: GET, POST, PUT and DELETE.

Rule-based systems and web-based systems are important part in theory of intelligent systems and artificial intelligence [26, 27], and have found their usage in lots of important works [28-34]. In our solution, we had to make modification and use combination of both in order to create solution that will fit the needs of this real life situation.

The proposed architecture of prototype intelligent rule-based expert system is shown in Figure 1.

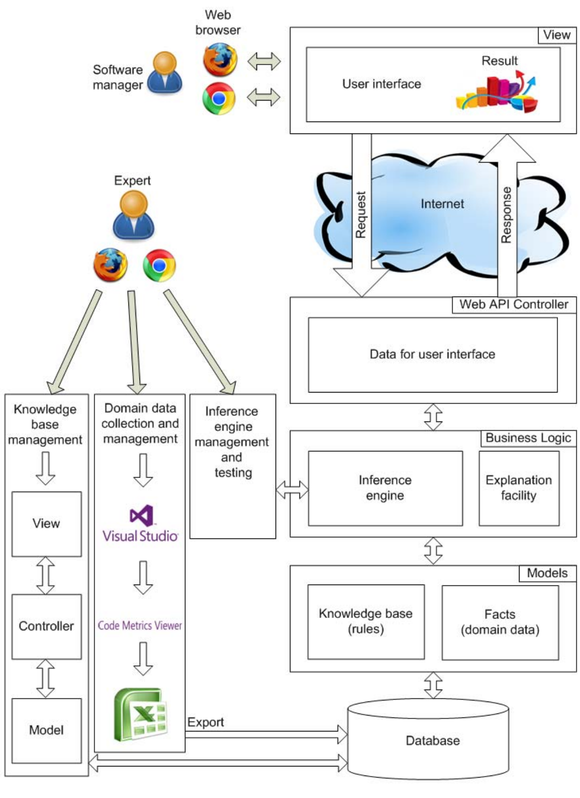

Figure 1. Architecture of prototype (web) rule-base expert system

A. Knowledge engineering

In order to make our prototype of intelligent system to work, knowledge for it had to be acquired from human expert, which in our case was the leading software manager. Capturing the experience, knowledge and reasoning of software manager was done using observations during his work and asking questions how did he reach the decision, interviewing him in situations that required his call, and discussing the various case studies that were appearing during software development process. Knowledge was then represented as a set of typical if-then rules, that could be used later in inference process. Human expert has the ability to insert, edit and test the rules, so that intelligent system can represent his logic as closely as possible. For this purpose, human expert uses web browser as his interface when working on knowledge engineering.

B. Collecting of Software Metrics History Data

Second important part of our intelligent system is ability to gather software metrics history data from the developed software solution, so that it can be used in the inference process. The production solution is being developed in Microsoft Visual Studio 2013 environment using C# programming language. Leading software manager is using Code Metrics Viewer plug-in for this environment, in order to track the quality of developed software code, represented in numerical form. The process of collecting software metrics history data begins with capturing software metrics data in numerical form represented in Code Metrics Viewer, upon which they are exported to the Microsoft Excel file. Each working day must be represented as one file, which contains the measured software metrics data from the production solution. After the files have been created, the files are being parsed and data is exported to SQL Server database, consolidated and aggregated. In this way, the database has the necessary data, in expert system terms – the facts, that are going to be used in inference process.

C. Inferencing in our prototype intelligent system

Coding standard is the very important in development of software systems. As it has been shown in our previous work [7], proposed parameters for software metrics give the expected results and move the development of the solution in desired solution, without the rise of the overall solution risk growth. The metrics in question is maintenance index (MI), cyclomatic complexity (CC) and lines of code (LOC). Proposed parameters of this metrics are: MI lower limit 50, CC maximum 10, and LOC upper limit 20 and critical limit more than 25. Only external indicator is end of iteration in days, which will help in predictions for software development.

Inferencing mechanism in our system is forward chaining technique, which means that it is using knowledge based rules and data to reach the decision, in our case, the prediction of future software development.

One example of inferencing for a particular project in production solution, based on its software metrics history data is represented in following manner:

IF Quality_MI = High
AND Quality_CC = Low
AND Quality_LOC = Low
THEN CodeQuality = Low

IF IterationEndsInDays > 0
AND IterationEndsInDays < 8
THEN IterationInState = Ending

IF CodeQuality = Low
AND IterationInState = Ending
AND CodeQualityIndicator = “Required”
THEN Prediction is “Refactor”
AND Effort is “Increased”
AND Productivity is “Increased”
AND PotentialForIterationBreach = “High”
AND Risk = “High”

Human expert has the ability to train the inference engine, in order to help the intelligent system to reach his decision making behavior.

D. Explanation facility

Inference engine is hidden from end user of the intelligent system. The human expert is the only one who has the ability to view and modify the rules in knowledge base and to test the inference engine how it reaches its decision. For end user of the system, it is often needed to explain how the system reached its solution that he predicted. In our proposed system beside the inference engine is standing the explanation facility, which will follow the rules that were fired by the inference engine, and based on them will generate most possible human language representation of the reached solution.

Returning to the example in previous section, the explanation facility would give back to end user following text:

“*Known facts for last 7 days: Average code standard for maintenance index is High. Average code standard for cyclomatic complexity is Low. Average code standard for lines of code is Low. Requirements for this iteration: Coding standard. Conclusion: Coding standard violated. Prediction: Action is refactor solution. Effort will be increased. Productivity will be increased. Iteration deadline breach potential is High. Risk is high.*”

Language variables like “refactor”, “increased”, “lower”, “high” will have their graphical/visual representation in the web page in the form of arrows, so that hey can implicitly induce the team leader to make adequate decision.

In this way, decision making of software managers and his fellow team leaders that are managing other developers will be significantly shortened and improved.

E. User Interface

Proposed intelligent system has the user interface in form of the web page so that team leaders can interact with it. User interface of the prototype intelligent system is shown n Figure 2.

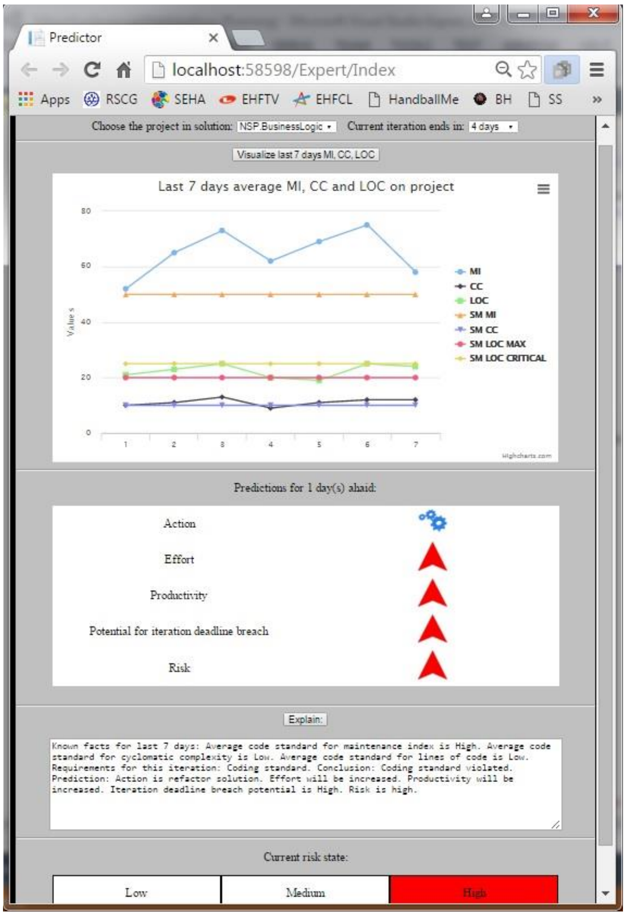

Figure 2. Interface for prototype rule-based expert system

User interface is fairly simple. It is a web page that consists of the following elements: dropdown box for choosing the project in developed solution and dropdown box for iteration end; button bellow it to trigger the visual representation of the current state of the project in terms of average MI, CC and LOC using Highcharts library; bellow the this graph there is visual representation of predictions made by inference engine, and a button for explaining how was the decision reached; lowest element of the page has the visual representation of the current risk on the examined project. All elements were written in HTML code.

V. Discussion Of The Solution And Results

The proposed intelligent system is in prototype phase when speaking in terms of expert system building methodology. Therefore, it is to young to give the precise picture of its ability to replace in full the human expert, our leading software manager. The thorough and exhaustive testing and validation of the expert system is yet to come in future months. Current testing of our prototype by human expert also shows that existing rules will have to be modified using fuzzy logic in the future, so that it can overcome their current strict boundaries. The limitation factor is also time and availability of the human expert and the developers team leaders who have lot of work to cover in their everyday work. In order to move on the development and testing of the prototype, it was given to 7 team leaders to try to work with it in their limited free time on local machines. After spending some time with the prototype, team leaders were given the questionnaire to answer questions regrading usefulness of the proposed rule-based expert system. The results are given in Figure 3.

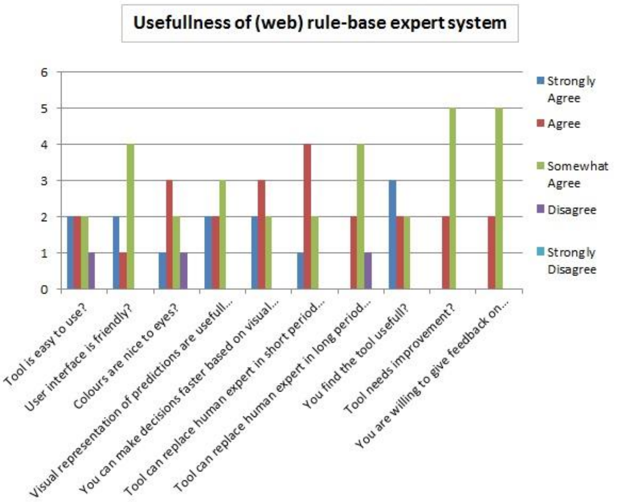

Figure 3. Questionnaire results from team leaders

The questionnaire included 10 questions with five level scale: strongly agree, agree, somewhat agree, disagree and strongly disagree. Questions that were given to team leaders are following: 1) Tool is easy to use?, 2) User interface is friendly?, 3) Colors are nice to eyes?, 4) Visual representation of predictions are useful to you?, 5) You can make decisions faster based on visual output?, 6) Tool can replace human expert in short period of time?, 7) Tool can replace human expert in long period of time?, 8)You find the tool useful?, 9) Tool needs improvement?, 10) You are willing to give feedback on improvement?

Results from analysis of questionnaire show following facts. On the first question, 57.1% of the team leaders would find the tool easy to use. Same percent of people find the colors used in the web interface appealing. User interface is friendly to 42.9% of examinees. Visual representation of predictions are useful to 57.1% percent of examinees. Five team leaders or 71.4% find that hey can make their decisions faster and that the tool could replace the human expert in short period of time. In this moment, tool is still not ready for long term replacement of the human expert, which can be seen from the fact that only 2 team leaders share the positive opinion about that question. Five team leaders or 71.4% of the total number of the examinees, would find the tool useful in their everyday work. The opinion that tool needs improvement was found with 2 team leaders or 28.6%, and 5 team leaders or 71.4% would somewhat agree with that fact. On the other hand, the 14.3% of the examinees did not find the tool easy to use, in the same percent did not like the colors used in web interface, and the same percent of the team leaders does not think that a tool could make replacement of the human expert in the long period of time. Lastly, all examinees showed similar interest and readiness to give feedback for expert system improvement, that was asked in last question of the questionnaire. Limitation of our proposed prototype of rule-based expert system is the fact that it is produced only for small number of people. In other words, in this moment, only a closed group of users can use it and test its usefulness for everyday work and operations.

VI. Conlusion

Software metrics has an important role in development of large software solutions. Its visualization and visualization of predictions of software development can play vital part of todays software manager tasks. In this paper we proposed the prototype of an rule-based expert system with capabilities of visual representation of predictions in future software development, based on software metrics history data. The results showed that prototype of the rule-based system would bring benefits for team leaders working on the software project. Benefits include visual representation of state of the developed project, visual representation of predictions for future software development, faster decision making of software managers, reduction of the risk in overall project based on visual representation of software metrics history data, which makes decision making more reliable, then analyzing numerical representation of the same. Our future work will be based on continuation toward ending the proposed expert system, and its finalization in terms of production implementation.

References

[1] D.Galin, Software Quality Assurance: From theory to implementation, Pearson Adisson Wesley, ISBN: 0-201-70945-7, 2004.

[2] I. Sommerville, Software Engineering, 9th edition, Addison-Wesley, ISBN-13 978-0-13-703515-1, 2011.

[3] D. Longstreet, Function Point Analysis Training Course, [online], available at: http://www.softwaremetrics.com/Function%20Point%20Training%20Booklet%20New.pdf, [accessed: 21.10.2015. at 23:55].

[4] N. E. Fenton, S. L. Pfleeger, Software Metrics: A Rigorous & Practical Approach, second edition, PWS Publishing Company, ISBN: 0-534-95425-1.

[5] Microsoft MSDN – Code Metrics Values, [online], available at: https://msdn.microsoft.com/en-us/library/bb385914.aspx, [accessed: 25.12.2015. at 13:35h].

[6] Code Metrics Viewer Values and Meaning, [online], available at: https://codemetricsviewer.wordpress.com/2011/06/26/how-to-interpret-received-results/, [accessed: 23.11.2015. at 15:28].

[7] Dz. Strujic, A. Balota, B. Popovic, Software metrics as the backbone of reducing risk in development of Enterprise software solutions, XXI science and business conference YU INFO, Kopaonik, Serbia, Proceedings, pp.151-156, ISBN: 978-86-85525-15-5, 2015.

[8] A. Marcus, L. Feng, J. I. Maletic, 3D Representations for Software Visualization, In: Proceedings of the 2003 ACM symposium on software visualization (SoftVis'03). ACM Press, New York, pp 27-36.

[9] C. Knight, M. Munro, Comprehension with[in] Virtual Environment Visualizations, In: Proceedings of the Seventh IEEE International Workshop on Program Comprehension, pp 4-11, 1999.

[10] J. Maletic, A. Marcus, M. L. Collard, A Task Oriented View of Software Visualization, Proceedings of the First International Workshop on Visualizing Software for Understanding and Analysis (VISSOFT '02), 0-7695-1662-9/02, IEEE, 2002.

[11] T. A. Ball, S. G. Eick, Software Visualization in the Large, IEEE Computer, Vol. 29, No. 4, April 1996, pp. 33-43, 1996.

[12] M. Lanza, S. Ducasse, Understanding software evolution using a combination of a software visualization and software metrics, L'objet-8/2002, LMO'2002, pp. 135-149, 2002.

[13] D. Holten, R. Vliegen, J. J. van Vijk, Visualization of Software Metrics using Computer Graphic Techniques, s.l., s.n.

[14] M. Compton, Visualization of Software Metrics, [online], available at: http://marlenacompton.com/wp-content/uploads/2009/05/visualization_of_software_metrics.pdf, [accessed: 04.10.2015. at 11:25h], 2009.

[15] G. Langelier, H. Sahraoui, P. Poulin, Visualization-based Analysis of Quality for Large-scale Software Systems, ASE'05, November 7-11, 2005, Long Beach, California, USA, pp. 214-223, ACM 1-58113-858-X/04/06, 2005.

[16] M. Balzer, O. Deussen, C. Lewerentz, Voronoi Treemaps for the Visualization of Software Metrics, Association for Computing Machinery, pp. 165-172, ACM 1-59593-073-6/05/0005, 2005.

[17] Atomiq, [online], available at: http://www.getatomiq.com/features, [accessed: 03.09.2015. at 14:20h].

[18] Nitriq Code Analysis, [online], available at: http://www.nitriq.com/features, [accessed: 03.09.2015. at 15:30h].

[19] Submain CodeIt.Right, [online], available at: http://submain.com/codeit.right/features.aspx#1, [accessed: 03.09.2015. at 16:15h].

[20] NDepend, [online], available at: http://www.ndepend.com/, [accessed: 03.09.2015. at 17:20h].

[21] D3 Data Driven Documents, [online], available at: http://d3js.org/, [accessed: 07.10.2015 at 01:23h].

[22] Javascript charts for web and mobile, [online], available at: http://www.fusioncharts.com/, [accessed: 08.10.2015. at 00:34h]

[23] jQuery plotting library, [online], available at: http://www.jqplot.com/, [accessed: 09.10.2015. at 23:45h].

[24] Google Charts library, [online], available at: https://developers.google.com/chart/interactive/docs/, [accessed: 12.11.2015. at 23:42h].

[25] Highcharts library, [online], available at: http://www.highcharts.com/, [accessed: 11.11.2015. at 20:15h].

[26] M. Negnevitsky, Artificial Inteligence: A Guide to Inteligent Systems, second edition, Addison-Wesley, ISBN: 0-321-20466-2, 2005.

[27] S. Russel, P. Norvig, Artificial Intelligence: A Modern Aproach, third edition, Prentice Hall, ISBN-13: 978-0-13-604259-4, 2010.

[28] P. Sosnin, Question-Answer Programming and Modeling in Expert Systems, Proceedings of the IASTED International Conference Artificial Inteligence and Applications (AIA 2009), pp. 208-123, Insbruck, Austria, 2009.

[29] M. S. P. Babu, N. V. R. Murty, S. V. N. L. Narayana, A Web Based Tomato Crop System Based on Artificial Intelligence and Machine Learning Algorithms, International Journal of Computer Science and Information Technologies, Vol. 1 (1), 2010, pp. 6-15, ISSN: 0975-9646.

[30] I. M. Dokas, Developing Web Sites For Web Based Expert Systems: A Web Engineering Approach, In: Proceedings of the Information Technologies in Environmental Engineering (ITEE'2005), September 25-27 2005, Otto-von-Guericke-Universitat Magdeburg, pp. 202-217.

[31] N. Askarov, K. Kozhakhmet, L. Atymtayeva, Analysis Of Web-Based Applications For Expert Systems, Computer Modeling and New Technologies, 2011, Vol. 15, No. 4, pp. 41-45.

[32] A. Abraham, Rule Based Expert Systems, Handbook of Measuring System Design, edited by Peter H. Sydenham and Richard Thorn, John Wiley & Sons, Ltd. ISBN: 0-470-02143-8, 2005.

[33] V. Rafe, M. H. Goodarzi, A Novel Web-based Human Advisor Fuzzy Expert System, Journal of Applied Research and Technology, Vol. 11, No. 1, pp. 161-168, 2013.

[34] Y. Duan, J. S. Edwards, M. X. Xu, Web-based Expert Systems: Benefits and Challenges, Information and management, Vol. 42, No. 6, pp. 799-811, 2005.