Maintenance-Ready Web Application Development

Ion IVAN, Narcis-Cosmin LUCA, Mihai Liviu DESPA, Eduard BUDACU
Bucharest University of Economic Studies, Romania
ionivan@ase.ro, narciscosmin@gmail.com, mihai.despa@yahoo.com, eduard.budacu@gmail.com

The current paper tackles the subject of developing maintenance-ready web applications. Maintenance is presented as a core stage in a web application's lifecycle. The concept of maintenance-ready is defined in the context of web application development. Web application maintenance tasks types are enunciated and suitable task types are identified for further analysis. The research hypothesis is formulated based on a direct link between tackling maintenance in the development stage and reducing overall maintenance costs. A live maintenance-ready web application is presented and maintenance related aspects are highlighted. The web application's features, that render it maintenance-ready, are emphasize. The cost of designing and building the web-application to be maintenance-ready are disclosed. The savings in maintenance development effort facilitated by maintenance ready features are also disclosed. Maintenance data is collected from 40 projects implemented by a web development company. Homogeneity and diversity of collected data is evaluated. A data sample is presented and the size and comprehensive nature of the entire dataset is depicted. Research hypothesis are validated and conclusions are formulated on the topic of developing maintenance-ready web applications. The limits of the research process which represented the basis for the current paper are enunciated. Future research topics are submitted for debate.

Keywords: *Software maintenance, Web Development, Maintenance-ready Applications, Maintenance Cost*

1 Research premises and hypothesis

Web application have to be designed to handle a large number of users as one of the advantages of being online is that fact that they are accessible by anyone with an Internet connection. The higher the number of protective users for the web application, the higher the probability of generating downtime [1] or surfacing functionality issues. As web applications grow and attract evermore user traffic their maintenance process needs to handle a vaster array of issues [1] thus growing evermore complex and costly.

Maintenance is an important stage in a web application's lifecycle. Maintenance used to be regarded as repair work [2] but it is now increasingly being considered as improvement work [3]. When developing a web application, one might either choose to focus on specifications, functionality and deadlines and deal with maintenance in due time or, thoroughly plan for the maintenance stage even from the start. The need to keep development costs down and meet strict deadlines compels most development teams to opt for the first option. Choosing the first option will help keep development costs down but will eventually generate cascading maintenance costs. Increasing maintenance costs might prove a challenge in the context of ever-increasing efforts aimed at modifying existing code. 50% of the global software population is engaged in modifying existing applications rather than writing new applications [4].

If the maintenance stage of an application's life cycle is considered during its development stage and its design and architecture are implemented so that the end product will be easily maintainable the result is labeled as a maintenance-ready web application. By developing the web application with considerable consideration to maintenance its complexity, cost and development time will increase. However, maintenance costs and development time will decrease. So, the main focus of the current research is to determine if building maintenance-ready web applications

DOI: 10.12948/issn14531305/20.3.2016.03

generates cost savings in maintenance that could compensate for the additional cost generated by additional time invested in building an easily maintainable application.

Thus the research hypothesis states that by developing maintenance-ready applications the time needed to implement maintenance tasks decreases and therefore the overall maintenance cost is considerably reduced.

According to [ISO/IEC 14764] maintenance is focused on four main objectives:

- **corrective** which entails emergency fixes and routine debugging [5] consisting of reactive modification of a software product performed after delivery to correct discovered problems; corrective maintenance tasks deal with bugs and issues that were missed by the testing team; reducing the number of corrective maintenance task is rather difficult as the testing process is error prone and no matter how much additional effort or resources you invest, it will always have considerable limitations; the major advantage in maintaining web application is that corrective code can be released and made available for all the users instantaneous [6];
- **adaptive** concerning software or hardware system changes [5] which consists of modification of a software product performed after delivery to keep a software product usable in a changed or changing environment; adaptive tasks include changes made to the web application in order to keep it up to date with changes made to the hosting environment, operating system, web server, database engine or supporting technologies; reducing the number of adaptive maintenance tasks is difficult to achieve as its almost impossible to foresee future technological changes and plan for them;
- **perfective** related to user enhancement, improved documentation and computational efficiency [5] which implies modification of a software product after delivery to improve performance or maintainability; perfective maintenance tasks often include changes requested by the web application owner, the marketing department or sales department; perfective maintenance tasks can be reduced by making the web application more customizable;
- **preventive** which tackles modification of a software product after delivery to detect and correct latent faults in the software product before they become effective faults; preventive tasks include checking the access logs and the error logs on a regular basis, making code and database backups, monitoring server load and retesting core functionality as often as possible; reducing preventive maintenance tasks is rather hard to accomplish as performing them more frequently often increases their effectiveness.

We can therefore conclude that building maintenance-ready web applications can be achieved through reducing the effort allocated to perfective tasks by integrating higher levels of customization and parametrization. Reducing the effort invested in a particular task translates into completing that task with as few development hours as possible. Perfective maintenance tasks are usually requested by the web applications owner or any other entity associated with the application owner like the sales, marketing, logistics or accounting department. An effective way of reducing the number of development hours, and therefore reducing the maintenance costs, is to provide the application owner with the tools to perform the tasks himself in a user friendly way. Such tools are obtained by increasing the parametrization and customization level of the web application.

2. Maintenance-ready web applications

Increasing parametrization and customization translates into providing the end-user and the web application's administrator with the option to change and adjust the functionality and the layout without actually performing code changes or by performing code changes that only require minimum expertise. In order to showcase the concept of maintenance-ready web applications the ALFA application is presented. The application name is anonymized in order to comply with legal constrains enforced by the application owner on the development team.

The ALFA application was designed with high emphasis on parametrization and customization as initial requirements were vague and the need to often change the content structure was outlined by the application owner. The solution that the development team came up with was to allow full customization of the structure and content by implementing a flexible grid-like backbone for the main pages. The grid is depicted in Figure 1.

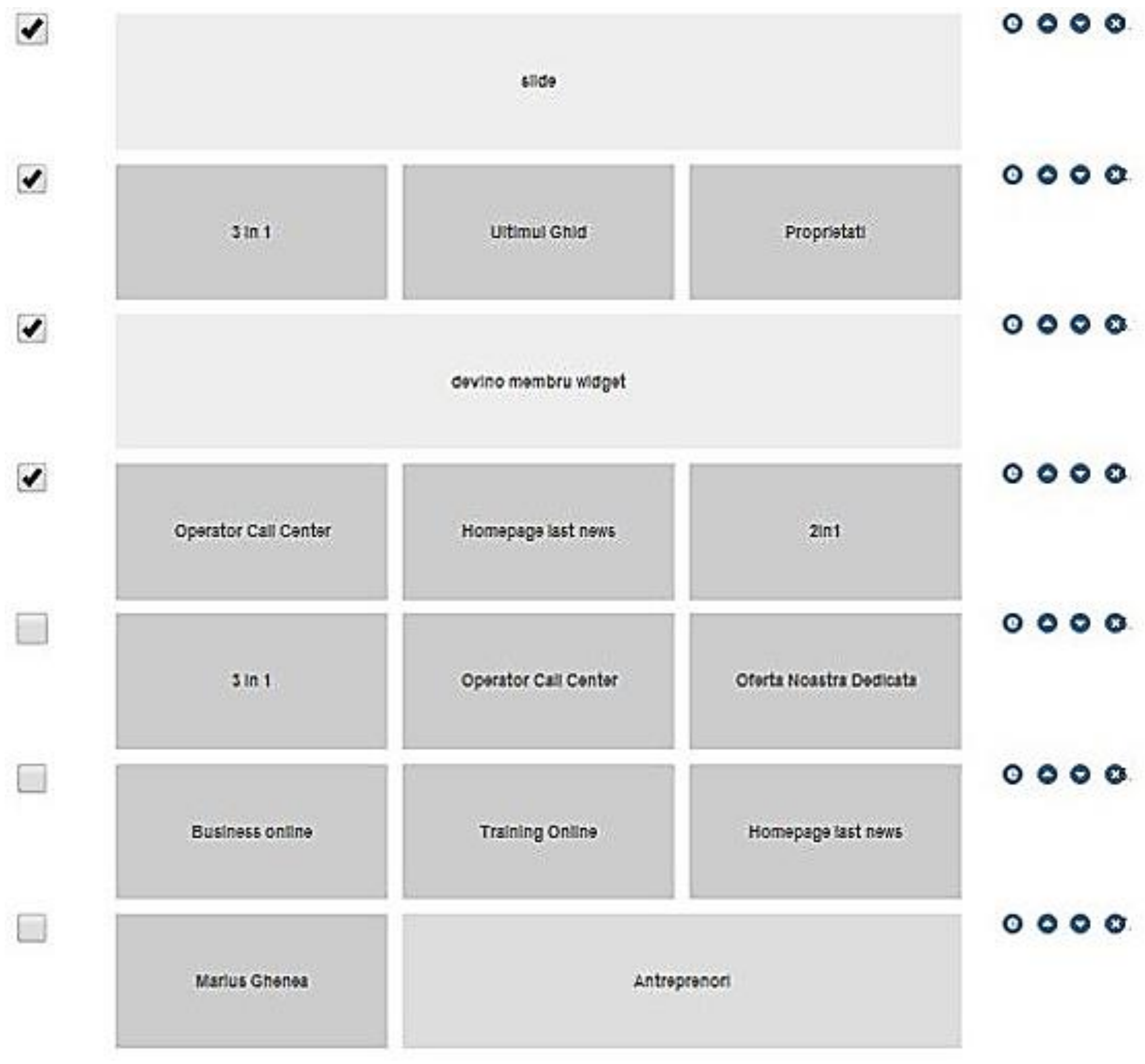

Fig. 1. ALFA application grid system

The grid system was manageable by row and the application administrator was able to decide which row was displayed to the users by simply checking and unchecking the left side checkboxes. Rows can be edited, deleted or moved up and down within the application page by using the buttons presented in the right side of Figure 1. Each row had a unit based structure. Each row could be built by using one, two or three rows. The first row in Figure 1 is built using one unit. The second row in Figure 2 is built by using three units. The last row in Figure 1 is built by using two units. The content in each unit is fully manageable by the web application's administrator with the help of a WYSIWYG editor. The editor allows the administrator to easily insert text, pictures and videos but also facilitates introducing basic code snippets. Each unit can be saved and used multiple times.

The high flexibility in managing the content and structure of the ALFA application reduces significantly the need to involve the development team in the maintenance process. In the ALFA application the cost of implementing the grid system which lead to making the web application maintenance-ready was 9% out of the overall development cost. Designing the ALFA application to be maintenance ready reduced the number of perfective tasks sent to the development team by approximately 70%.

3. Research hypothesis validation

In order to test the research hypothesis data from real life web development projects was used. All the data was collected form the same web development company. Data totals a number of 588 maintenance tasks from 40 different projects collected within a four-month timespan. A sample of the data is presented in Table 1.

Table 1. Maintenance tasks data sample.

Task Name	**Client**	**Project**	**Type**	**Duration (hours)**	**Cost ($)**
PAT Interview Questions Guide	David	PAT	perfective	3	105
Pop up box when certain email addresses sign up	David	PAT	perfective	1	35
Update Value Story completion when a Task is marked as completed on the One Page Plan	Paul	CT	perfective	4	140
SOK Registration process 2	Bjorn	sok2016	perfective	2	70
Monitor is UP: Saleoot com	Greg	saleoot	corrective	2	70
Custom content preview plugin	Erik	Cunning ham	corrective	3	105
Cancel profile Louise Robinson/Taylor	Alex	TLT	perfective	0.5	17
Add Houslow as the location of the job as one of the sub locations for London	Alex	TLT	perfective	1	35
Exhibitors list-SOK2016	Bjorn	sok2016	perfective	1	35
Order form	Karl	mealtek	corrective	9	315
Saleoot Feed changes	Greg	saleoot	adaptive	0.5	17
Infospace one off task	Greg	OSN	perfective	3	105
Medical Malpractice Guide template	Erik	Cunning ham	perfective	6	210

Task Name	Client	Project	Type	Duration (hours)	Cost ($)
Change devzy AWS configuration	Andy	Devzy	preventiv e	3	105
Sabres - Add checkbox to products that appear on homepage	Sely	Sabres	perfective	2	70

Task name column represents the way the task is referred internally by the development team. The task name derives from the email subject in which the task was requested. *Client* column represents the person to which progress on the task needs to be reported. *Project* column represents the name of the project the task belongs to. *Type* column refers to the category of the maintenance task and takes the values corrective, adaptive, perfective and preventive. *Duration* column represents the number of hours spent on a particular task by the development team. *Cost* column represents the amount billed for the time consumed on implementing a particular task. The standard rate is $35 per hour.

Table 2. Task duration and cost aggregated by task type.

Task Type	Number of tasks	Duration (hours)	Cost ($)
corrective	48	73	2.555
adaptive	57	42.5	1.487
perfective	449	826	28.910
preventive	33	60.5	2.117

Corrective maintenance tasks amounted for 73 hours of development and generated a cost of $2.555. Adaptive maintenance tasks amounted for 42.5 hours of development and generated a cost of $1.487. Perfective maintenance tasks amounted for 826 hours of development and generated a cost of $28.910. Preventive maintenance tasks amounted for 60.5 hours of development and generated a cost of $2.117. Perfective maintenance tasks alone generated a maintenance cost that exceeds the maintenance cost generated by all the other maintenance tasks put together.

Maintenance-ready web applications are designed to facilitate the maintenance process and reduce the number and complexity of perfective tasks. Assuming that the overall cut in perfective tasks is consistent to the case of ALFA application it would infer a cost cut in maintenance tasks of $20.237. The cost of developing each of the analyzed web application is presented in Table 3.

Table 3. Project development cost

Project	Cost ($)
ASA	1100
AVZ	27000
BRK	800
BD	1600
CTLS	3100

Project	**Cost ($)**
CB	1600
CO	1300
CT	13000
CNGHM	2700
DVZ	3200
DQS	2300
DSF	1800
ELLT	1100
FITD	7000
GTW	2100
HTGVS	2300
IMM	3100
LDBB	3100
MC	1100
MCR	2700
MLTK	600
MV	1000
NGNT	1300
OSN	4800
PRTS	1500
PAT	12000
RC	8100
RMG	800
sabres	1800
SOOT	1100
SI2	5000
SOK	1600
STC	2300
TC	3200
TLT	1800
TRCTS	1400
WP	3000
WWS	1200
YWA	200

The total cost of developing the 40 web application was $134.700. Assuming that and additional cost of 9%, as determined for the ALFA application, would have transformed the 40 web applications into maintenance-ready web applications, the total cost would have amounted to $146.823 with $12.123 invested in making the web application maintenance-ready. Therefore, the additional cost of making the applications maintenance-ready would have been $12.123 but making the applications maintenance-ready would have saved $20.237 in maintenance costs. Thus the additional cost generated by designing and building an application to be maintenance ready is on average 60% of the amount that will be saved within the maintenance process by reducing the number and complexity of perfective maintenance tasks.

The research hypothesis stating that *by developing maintenance-ready applications the time needed to implement maintenance tasks decreases and therefore the overall maintenance cost is considerably reduced* is therefore validated. Additional research needs to be performed to determine with higher precision the average cost of designing and implementing an application to be maintenance ready. The current research only analyzed one application and data collected form that application was used to set the thresholds for costs in terms of building maintenance-ready functionality and also for determining the threshold for reducing the number and complexity of perfective maintenance tasks. By applying the thresholds to a number of 40 web development projects the research hypothesis is confirmed but future research should focus on validating and optimizing the thresholds for additional costs and decrease in number and complexity of maintenance tasks.

5. Conclusion

Maintenance is an important stage in a web application's lifecycle as it has the role of ensuring proper functioning of the application after deployment on the live environment. Maintenance is a costly process especially when the web application has been developed with no regards to upcoming maintenance operations. Building a web application that is easily maintainable means increasing its customization and parametrization degree, namely making it easy to extend functionality and change content with the least amount of code writing. Increasing the degree of customization and parametrization generates an increase in development costs but induces a decrease in maintenance costs. In the case of the ALFA application the cost increase generated by making the web application maintenance ready was 9%. The decrease in maintenance costs is achieved by reducing the number and complexity of perfective maintenance tasks. In the case of the ALFA application the number of maintenance perfective tasks was decreased by approximately 70%. Thus building maintenance-ready web applications can be achieved through reducing the required number of perfective tasks by integrating higher levels of customization and parametrization. The study on 40 web applications highlighted that the costs generated by designing and building a web application to be maintenance ready is on average 60% of the amount saved later in the maintenance process due to the maintenance-ready characteristics of the web application. Thus building maintenance-ready web applications is cost effective as it saves more on maintenance than it spends on additional development cost. The current research results are limited by the fact that only one application was used as a benchmark for determining the threshold for reducing the number and complexity of perfective maintenance tasks and for additional costs generated by building maintenance-ready functionality. Future research needs to optimize the thresholds for additional costs and decrease in number and complexity of maintenance tasks.

References

[1] S. Pertet and P. Narasimhan, ”Causes of failure in web applications”, Parallel Data Laboratory, Technical Report CMUPDL-05-109. December 2005, available at http://repository.cmu.edu/cgi/viewcontent.cgi?article=1047&context=pdl

[2] S. Takata, F. Kirnura, F. J. A. M. Van Houten, E. Westkamper, M. Shpitalni, D. Ceglarek and J. Lee, ”Maintenance:

changing role in life cycle management" *CIRP Annals-Manufacturing Technology*, vol. 53, no. 2, pp. 643-655, 2004.

[3] S. Eick, T. Graves, A. Karr, J. Marron and A. Mockus, "Does Code Decay? Assessing Evidence from Change Management Data", *IEEE Transactions on Software Engineering*, vol. 27, no. 1, pp. 1-12, 2001.

[4] C. Jones. (2006). The economics of software maintenance in the twenty first century. Unpublished manuscript. Available: http://www.compaid.com/caiinternet/ezine/capersjones-maintenance.pdf.

[5] B. P. Lientz, E. B. Swanson and G. E. Tompkins, "Characteristics of application software maintenance", *Communications of the ACM*, vol. 21, no. 6, pp. 466-471, 1978.

[6] J. Offutt, "Quality attributes of web software applications", *IEEE software*, vol. 19, no. 2, pp. 25-32, 2002.

Ion IVAN has graduated the Faculty of Economic Computation and Economic Cybernetics in 1970. He holds a PhD diploma in Economics from 1978 and he had gone through all didactic positions since 1970 when he joined the staff of the Bucharest University of Economic Studies, teaching assistant in 1970, senior lecturer in 1978, assistant professor in 1991 and full professor in 1993. Currently he is full Professor of Economic Informatics within the Department of Computer Science in Economics at Faculty of Cybernetics, Statistics and Economic Informatics from the Academy of Economic Studies. He is the author of more than 25 books and over 75 journal articles in the field of software quality management, software metrics and informatics audit. His work focuses on the analysis of quality of software applications.

Narcis-Cosmin LUCA has graduated Transilvania University form Brasov in 2007 with a diploma in Mathematics - Informatics. He is vice-president and founding member of the ASURA Association and has extensive experince as a frelance software developer, collaborating with various companies in Romania, Austalia and France. His main field of interest is software development with a focus on building and maintaining web applications.

Mihai Liviu DESPA has graduated the Faculty of Cybernetics, Statistics and Economic Informatics from the Bucharest University of Economic Studies in 2008. He has graduated a Master's Program in Project Management at the Faculty of Management from the Bucharest Academy of Economic Studies in 2010. He has a PHD in Managing innovation oriented software development projects obtained in 2015 at the Economic Informatics PHD School and he is currently Project Manager at GDM Webmedia SRL. His main field of interest is project management for software development

DOI: 10.12948/issn14531305/20.3.2016.03

Eduard Budacu and has graduated the Faculty of Cybernetics, Statistics and Economic Informatics from the Bucharest University of Economic Studies in 2010. He has graduated the SIMPRE - ERP oriented Master's Program from the Bucharest Academy of Economic Studies in 2012. He is currently a PHD Student at the Economic Informatics PHD School. His main field of interest is Agile software development. He is an Agile coach and helps software development teams produce positive change in order to achieve high performance using agile principles and practices. He works with companies to define learning and development strategies for agile transformation.

DOI: 10.12948/issn14531305/20.3.2016.03