Aspects of improvement of software development lifecycle management

József Klespitz*, Miklós Bíró**, Levente Kovács*
* Óbuda University, Budapest, Hungary
** Software Competence Center Hagenberg, Hagenberg, Austria
klespitz.jozsef@phd.uni-obuda.hu, miklos.biro@scch.at, kovacs.levente@nik.uni-obuda.hu

Abstract— **The software development lifecycle management is a crucial aspect of software development, as it defines significantly the effectiveness of development and the maturity of the final product. A suitable management system is especially important in safety critical development, such as the development of medical devices, as these developments have to be well documented to prove the safe operation of the given instrument. This paper prepares a case study to help a company improving their existing lifecycle management system by establishing a group of requirements. In particular, the paper focuses on medical device software development aspects. The answering of these questions allows achieving an optimal choice among the possible management systems either by providing a quantitative comparison or by highlighting the grounds for exclusion.**

Keywords: software development lifecycle, application lifecycle management, software development process improvement

I. INTRODUCTION

The high quality of developed software is important for many companies, but it is crucial for safety-critical software developing teams as they are bound by directives. These directives may contain general recommendations regarding the software development such as IEC 61508 standard [1]. Nevertheless, the different fields have more specified directives from which the most relevant are the ISO 26262 for road vehicles [2], DO-178B for airborne systems [3] and IEC 62304:2006 for medical devices [4]. The fulfilments of these standards are necessary for an equipment to be launched.

In case of medical devices both the Medical Device Directive (MDD) responsible for the European market, and the US Food and Drug Administration (FDA) responsible for the USA market accept the IEC 62304:2006. In case of a medical device, which is the main scope of this research, many more standards have to be complied. Such standards are the quality management standard ISO 13485 [5], the risk management standard ISO/IEC 14971:2007 [6] or ISO/IEC 12207 standard for software [7] and ISO/IEC 15288 for systems [8] [9]. The need of compliance with these standards means the most relevant difference between safety-critical and non-regulated software development.

To fulfil the requirements arisen by these standards the development process has to be well documented. The effort to accomplish this heavy documentation need can be highly reduced by establishing a suitable application lifecycle management (ALM) system. Most commonly plan driven software development is applied for such development; although this is not required by standards and directives [10]. Although there are trials to introduce other approaches, such as agile development, yet these are still uncommon [11]. Therefore, ALM systems on the market target mostly companies with plan-driven software development method (Fig. 1.) [12, 13], but these systems use different approaches by accentuating different components, such as requirements, testing, etc. A particular system component is the traceability of which creation and maintenance is done by the users, but can be highly supported by the used system.

The numerous ALM systems on the market raise the need of choosing them properly according to the need of the developer team. The goal of this paper is to organize the arising questions and give a case study when introducing an existing ALM system. The final aim is to create an ALM system, which ensures the needs both for developers and managers, while it supports traceability [14] in an automatic and ubiquitous way [15]. An important aspect focuses on supporting the reuse of as much code as possible, keeping the possibility to establish a product line in development [16].

The paper is continued by presenting the examined environment. Afterward, the aspects of comparisons are discussed in detail. Finally, the conclusions and future work ideas are listed.

II. ENVIRONMENT OF APPLICATION

The requirements are collected for a medium enterprise case, with special focus on medical device development. It is assumed that the company has an existing ALM system, having former experience on it. Although medical device case is discussed, the requirements are not restricted to medical device developers, but they might be beneficial for any team who develops safety-critical software. The whole software development lifecycle is examined, so the requirements will not relate only to the chosen software but to its application environment (and possible extensions) as well.

978-1-4673-8520-6/15/$31.00©2015 IEEE

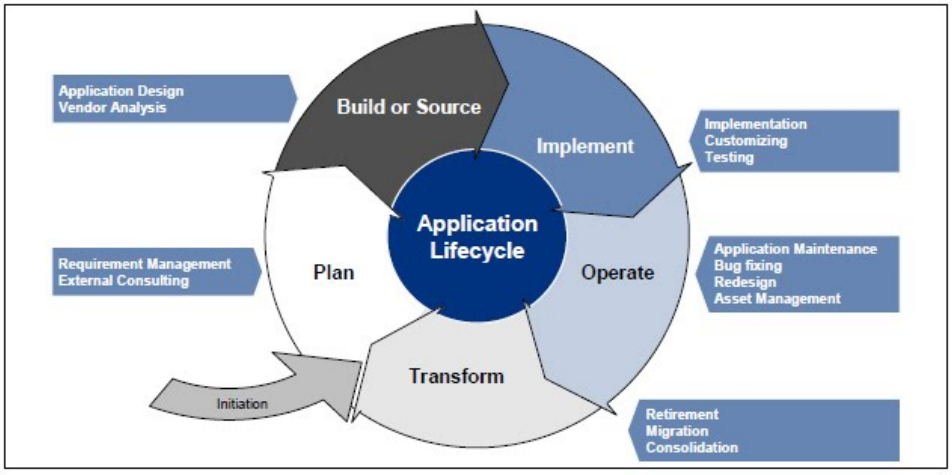

Figure 1. Application lifecycle management systems [26]

III. ASPECTS OF COMPARISON

In this chapter every aspect that helps the decision of choosing between ALM systems are discussed. These aspects are partially selected from the features from different ALM systems and partially collected through discussions with experts and relevant people.

The first group of requirements discuss the environment where the ALM system works. The ALM system could have multiple software components. The system might be web-based, in this case the client is a web browser and the system itself is installed on a web server. Otherwise, a local client could be installed, while the master is installed on a suitable server. In this comparison the web-based solutions are superior as only the install of a single instance is necessary, easing the system maintenance. Moreover, the web-based solution can be accessed anytime, anywhere in case of an active internet connection with a proper authentication. On the other hand, locally installed software might provide more features which could effectively enhance the effectiveness of work. Furthermore, dedicated software components are usually faster and could perform the necessary commands more effectively.

Similarly the created data can be stored either on an own server or on a remote server. The security is an emerging question in this case. The locally stored information could be defended by the owner company, while in case of remote storage the retailer of data storage ensures it. This latter question cannot be answered objectively; it should be an agreement of the project owners. The communication between the different components is basically defined according to the above described issues. Still it has importance, as tool integration or external access may require access to this communication channel. Known, legal access to the communication channel is required for further functional extension. If a system setup could be installed in different ways (e.g. a web-based version and a client-server version) their performance has to be compared. Both setups have to be examined and they should be tested with relevant amount of data. The handling of large data should be tested practically in every case.

The second group of requirements refers to the financial aspects of ALM system. The price of ALM system may be either a personal licence or a floating licence. Usually the floating licence is more expensive as in this case the number of simultaneously working people is limited only by the number of licence. Moreover, a machine licence has to be purchased as it is common to run different tasks (scripts) automatically, usually outside of working hours. Some ALM systems have built-in timeouts to filter inactive users (e.g. those who has finished working but did not checked out). This timeout should not hinder the work neither for people nor for the automatic scripts by interrupting their work sequence. When calculating the price the number of realised features by the ALM has to be considered. (E.g. Polarion® ALM™ can be bought to handle requirements or to handle testing or as a complete system.) The missing features could be developed by the owner company or by third party, which development cost has to be added to the total cost of the establishment of ALM system.

It is assumed that the company has an existing ALM system, so they have an existing database and prior experience. Therefore it is important to be able to migrate the existing data base to the new system. This supposes new ALM system interfaces: it is capable to import database from common formats (such as a CSV file or a ReqIF file). Moreover, the time and resource need of migration should be also considered. The system has to be able to provide necessary outputs as well: for further migration purposes it should support the previously mentioned formats and it should be able to satisfy the documentation needs. This means that the necessary parts should be able to be printed out in a structured way and simultaneously if possible. Moreover, certain documents has to be prepared and kept, the system has to be able to export these documents as well. Simultaneous document export is a preference.

The automated documentation generation is also favoured. The new ALM system has to handle in similar or in an improved way the project and repository structures. This means that the ALM system should be able to model the applied development process (plan-driven and other models as well) while it has the possibility to handle branching (version control), product line design and has to support reuse as well. This has to be complemented by supporting help to prove compliance with directives and it has to provide system modelling tools as well. The support of product development line raises the need of branching not only the software version

(or even device versions) but the documentation as well. Ideally, the documentation of a new product or software version arises with little to no effort.

The usability of the system is a marginal question. The use of the system should be easily overviewed. Hence, most of the features must be intuitive, so a familiar editorial interface is required; this means word like interface for requirement editing or excel like interface to handle tests. The aim is to spend as little effort as possible to learn its use. In development and especially during documentation many repetitive patterns and forms can be recognized. To decrease the work effort templates are created and the ALM system should support the use of these. Involving many people in a project, it is inevitable to face with encounter that should be supported by the editorial interfaces. Hence, the real-time access and parallel editing is required. Furthermore, a practical locking mechanism is expected together with a conflict solver.

Certain software development tools have extensions which make possible to communicate with specific ALM systems. For example the Matlab®-Simulink® has a toolbox which makes possible to connect Simulink blocks with IBM® Rational® DOORS® requirements. Such possibilities have to be explored and considered according to the used development tools. Otherwise, the effort to create connection between the development tool and the ALM has to be considered.

Authentication is a must in such systems, but it is expected to distinguish people according to their rights. This requires specific and restricted views. Optionally these restricted views can be configurable.

The customization is another significant question. These ALM systems are created for general use; hence, customization is inevitable. Many different workflow models exist, which should be configurable. Ideally, the system enables dynamic workflow definition not hindering the development. Furthermore, default tags and other identifiers for different artefacts rarely characterize perfectly the stored objects. Therefore, enter of new (custom) properties are needed. The same rule stands for people's rights as well. Moreover, the view should not only be settable centrally according to the existing rights, but it has to be allowed for people to create their own views. Usually, this means to set up a rule-base for filters, where the filtered properties must contain the default properties and user made (custom) properties as well. The application of new functions or views and creation of new workflow items must be simple. Ideally, the validity of introduced modification can be chosen among certain project(s) or the whole repository. Again due to the general nature of ALMs missing features are inevitable. If the ALM has strong scripting possibilities these deficiencies can be bypassed. The effectiveness and easiness of scripting must be an important decision point.

The development process requires creating baselines not only from software but from documentation point of view as well. It has to be examined if the ALM supports base lining of different statuses of the project. Furthermore, it could be important to preserve a certain status of the project which raises the need to store a certain state of it, basically creating a snapshot from it.

The importance of traceability was proved by many researches [19, 20, 21, 22], but usually it takes high effort to provide the bilateral (bidirectional) traceability [15, 23, 24]. An ideal ALM system supports the creation and maintenance of traceability in a ubiquitous and automatic way. It is expected from the system to link artefacts automatically (or at least with minimal effort), to have built-in tools checking traceability defects (e.g. by searching for components without link). Lacking these features, the system has to provide the possibility to realize them via scripting. Moreover, analysis with semantic heuristics might be applied to find traceability problems.

The system might have (or can be extended) with various automatic features similarly to the traceability. Hence, a test system might be integrated into the ALM system. This test system might check the requirements automatically for existing valid connection with a verification test, might evaluate (or import external) test results such as static and dynamic code analysis, and might handle the scheduling of these tests. It has to support the change management by providing a modification claim or ticket system to follow software changes and it has to keep tracking the changes of entries; thus providing a history. The changes should be visible and searchable.

The human interaction has to be enhanced as well. It was experienced that the ticket system requires an internal (in-system) communication channel (through chat or comment section); thus the professional discussions are more effective and also searchable. The same applies for managing the tasks of people. It must be clear for the manager to see the responsibilities and performance of a worker, while the developers have to see the problems to be solved with their own responsibilities as well. The realization of this feature could be various ranging from a visualized workflow diagram with participants or by simply sending messages. Either way the system has to provide clear information for everyone.

Finally, the system has to provide measures for the leadership. This might relate to the project(s) and to people as well, and can be extracted from system information. The visualization is important for the faster and easier understanding. Moreover, the visualization system can be extended with different state marks characterizing the actually running processes. Such notation is used in Kanban approach [17], which can be effectively used in software development as well [18].

The above mentioned requirements could be fulfilled with a single ALM tool with script extension or by connecting different tools. In latter case the OSLC compatibility has to be analysed as the most current solution connecting tools. According to the final vision every line of code is documented and traced with minimal human interaction through a unified (development) interface.

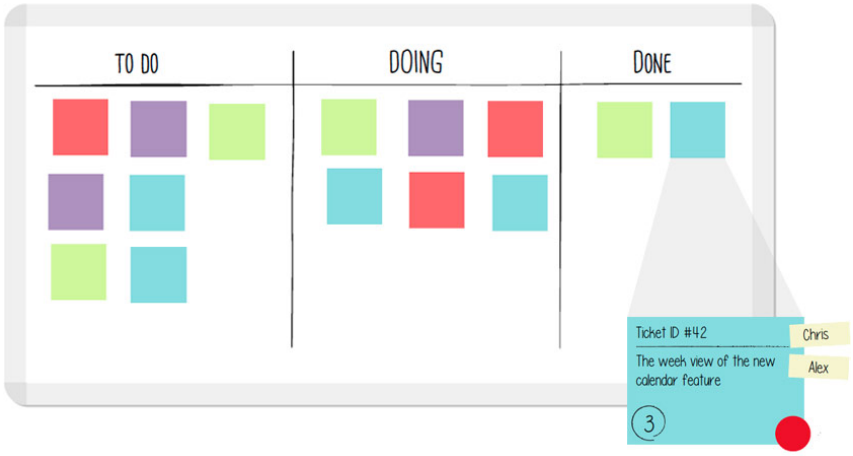

Figure 2. Kanban table with different colors according to the importance of task [27]

IV. Conclusion and Further Work

The heavy documentation need of safety critical software developments raises the need of a well-established software development management system. Various aspects were mentioned when an ALM system is introduced or changed. The explorations of disadvantages are useful as well, because a plan can be created to bypass the deficiencies and accelerates the introduction of the new system. However, these requirements are informal and sometimes subjective. The next step during the evaluation process should be to create objective measures [25] from these requirements. Furthermore, the grounds for refusal have to be clearly clarified as well. After these, a case study will be executed, where the emerging ALM systems will be analysed.

Acknowledgement

The writing of this article has been supported by the Austrian Ministry for Transport, Innovation and Technology, the Federal Ministry of Science, Research and Economy, and the Province of Upper Austria in the frame of the COMET center SCCH."

References

[1] International Electrotechnical Commission. *Functional Safety of Electrical/Electronic/Programmable Electronic Safety-related Systems.* IEC-61508:2010

[2] International Organization for Standardization. *Road vehicles – Functional safety.* ISO 26262:2011

[3] Radio Technical Commission for Aeronautics, European Organization for Civil Aviation Equipment. *Software Considerations in Airborne Systems and Equipment Certification.* RTCA/DO-178B

[4] International Electrotechnical Commission. *Medical device software – Software life cycle processes.* IEC 62304:2006

[5] International Organization for Standardization. *Medical devices – Quality management systems.* ISO 13485:2003

[6] International Organization for Standardization. *Medical devices – Application of risk management to medical devices.* ISO 14971

[7] International Organization for Standardization, International Electrotechnical Commission. *Systems and software engineering – Software life cycle processes.* ISO/IEC 12207:2008

[8] International Organization for Standardization International Electrotechnical Commission. *System and Software Engineering – System life cycle processes.* ISO/IEC 15288:2015

[9] Miklós Biró. "*Functional Safety, Traceability, and Open Services.*" In Software Engineering from Research and Practice Perspective, pp. 73-82, Scientific Papers of the Polish Information Processing Society Scientific Council, 2014.

[10] Martin, McHugh; Fergal , McCaffery,. "*Challenges Experienced by Medical Device Software Organizations while following a Plan-driven SDLC*". European Systems and Software Process Improvement and Innovation Conference EuroSPI Dundalk, Ireland, 2013.

[11] Martin, McHugh; Fergal, McCaffery; Valentine, Casey; Minna, Pikkarainen. "*Integrating Agile Practices with a Medical Device SDLC.*" European Systems and Software Process Improvement and Innovation Conference, EuroSPI, Vienna, Austria, 2012.

[12] Sandeep, Chanda; Damien, Foggon. "*Application Lifecycle Management*". In:Beginning ASP. NET 4.5 Databases. Apress, pp. 235-249., 2013.

[13] David, Chapell. "*Application lifecycle management as a business process*". 2008.

[14] Glenn, A. Stout. "*Requirements traceability and the effect on the system development lifecycle (SDLC)*". Systems Development Process Research Paper, pp. 3-17, Spring Cluster, 2001.

[15] Orlena, Gotel; Jane, Cleland-Huang; Jane, Huffman Hayes; Andrea, Zisman; Alexander, Egyed; Paul, Grünbacher; Alex, Dekhtyar; Giulio, Antoniol; Jonathan, Maletic. "*The grand challenge of traceability (v1.0)*". In Software and Systems Traceability, pp. 343-409., Springer London, 2012.

[16] William, Frakes; Carol. Terry. "*Software reuse: metrics and models*". In ACM Computing Surveys (CSUR), Vol. 28. No. 2., pp. 415-435., 1996.

[17] John M., Gross; Kenneth R., McInnis. "*Kanban made simple: demystifying and applying Toyota's legendary manufacturing process*". AMACOM Div American Mgmt Assn, 2003.

[18] Xiaofeng, Wang; Kieran, Conboy; Oisin, Cawley. "*"Leagile" software development: An experience report analysis of the application of lean approaches*"- In agile software development. Journal of Systems and Software, Vol. 85. No. 6: pp. 1287-1299., 2012.

[19] Padmalata, Nistala; Priyanka, Kumari. *Establishing content traceability for software applications: An approach based on structuring and tracking of configuration elements*. In 7th Workshop on Traceability in Emerging Forms of Software Engineering (TEFSE), pp. 68-71., IEEE, 2013.

[20] Bouillon , Elke; Patrick, Mäder; Ilka, Philippow. *A survey on usage scenarios for requirements traceability in practice*. In 19th International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ), vol. 7830 of LNCS, pp. 158-173, Springer Berlin, 2013.

[21] Patrick, Mäder; Orlena, Gotel;Ilka, Philippow. *Motivation matters in the traceability trenches*. In 17th IEEE International Requirements Engineering Conference (RE), pp. 143-148, IEEE, 2009.

[22] Michael C., Panis. *Successful deployment of requirements traceability in a commercial engineering organization...really*. In 18th IEEE International Requirements Engineering Conference (RE), pp. 303-307, IEEE, 2010.

[23] Jane, Cleland-Huang; Olly, Gotel; Jane, Huffman Hayes; Patrick, Mäder; Andrea, Zisman. *Software traceability: trends and future directions*. In: Proceedings of the on Future of Software Engineering. ACM pp. 55-69, 2014.

[24] Orlena, Gotel; Jane, Cleland-Huang, Jane, Huffman Hayes; Andrea, Zisman; Alexander, Egyed; Paul, Grünbacher; Alex, Dekhtyar; Giulio, Antoniol; Jonathan, Maletic; Patrick, Mäder. *Traceability fundamentals*. In J. Cleland-Huang, O. Gotel, and A. Zisman, editors, Software and Systems Traceability, pp. 3-22. Springer, 2012.

[25] Manar, Abu, Talib; Adel, Khelifi; Alain, Abran; Olga, Ormandjieva. "*Techniques for quantitative analysis of software quality throughout the SDLC: The SWEBOK guide coverage*". In Eighth ACIS International Conference on Software Engineering Research, Management and Applications (SERA), pp. 321-328, IEEE, 2010.

[26] Noise Consulting Group, "Managing the application lifecycle from inception to retirement", (last seen Oct. 2015.) http://noisetcd.com/tech-slns.html?alm

[27] LeanKit Inc. "*A (very) short history of Kanban*", (last seen Oct. 2015.) http://leankit.com/kanban/what-is-kanban/