Demonstration of Augmented Lifecycle Space in Heterogeneous Environment

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

Abstract — **Safety-critical and other high reliability software developments cannot be created without additional support tools. The aim of these so called application lifecycle management systems are to make it possible to track every step of the development process and support the documentation created for customers and regulatory bodies. In such developments, it is crucial to prove that traceability is complete and there is no contradiction between the artifacts. In former papers, the idea of the general approach called Augmented Lifecycle Space was presented which can effectively support the detection of traceability gaps and inconsistencies. Furthermore, the application of this method was demonstrated in homogeneous systems. In this paper, the applicability of this approach is presented in heterogeneous environment where the theoretical problems are partially already solved and it is possible to give insight of the technical difficulties.**

Keywords: software development life cycle, application life cycle management, software development process improvement

I. INTRODUCTION

High quality software is required in many different application fields, such as in telecommunication, information technology, or in the financial sector. The need for quality is motivated in these sectors mostly by the expected high financial loss in case of malfunction. Still, these developments are not bounded by as many directives and standards as the safety-critical software development [1, 2, 3].

On the other hand, safety-critical applications are hazardous due to their nature and it is both ethical and legal concerns to keep the risk of these systems as low as reasonably possible (ALARP) or rather as low as reasonably achievable (ALARA) [4]. The concerning directives may specify general safety requirements, such as IEC 61508 standard [5], which gives recommendation according to functional safety of electronic systems. Furthermore, different fields have their own specific directives from which the most relevant are the ISO 26262 for road vehicles [6], DO-178C for airborne systems [7] and IEC 62304 for medical devices [8]. The requirements in these standards have to be fulfilled, otherwise the product cannot be lunched. Interestingly, the manufacturer have to state the compliance to the related standards and directives with all of the consequences. Naturally, certification bodies are involved to proof this compliance as independent third party.

In case of medical devices, which is the main scope of this research, the European Medical Device Directive (MDD) and the Federal Food and Drug Administration (FDA) of the United Sates are the important regulatory bodies responsible for the local directives. Furthermore, the developed device has to comply with various standards such as the quality management standards ISO 13485 [9], the risk management standard ISO/IEC 14971 [10] or ISO/IEC 12207 standard [11] for software, and ISO/IEC 15288 standard [12] for systems. Together, these standards and directives define the corner stones of development, including processes.

Traceability is emphasized by many of the standards and its need is recognized even by the agile community [13]. By definition, it is dependency relationship of any kind. This means, that the derivation of development items can be marked with various methods: A textual references does the task, but it is always better to have more expressed relationship. The continuous maintenance of traceability is one of the most demanding task in case of complex developments with extensive database [14]. Similarly, the detection of inconsistencies can be difficult considering ten thousands of development items.

In order to ease the burden and make traceability and consistency more ubiquitous the Augmented Lifecycle Space (ALS) approach was introduced. The main goal is to provide a general method which can be used mostly in heterogeneous application lifecycle management systems to fill traceability gaps and correct inconsistencies by providing automatic workflow generation.

In previous paper [15] the execution steps of ALS approach was discussed while in paper [16] the practical implementation of ALS method was presented. However, the real benefits of this approach are more remarkable in heterogeneous systems where the general transparency of system is worse. Therefore, the aim of this paper is to demonstrate the efficiency of the idea in such environment and to give some idea not only about the theoretical difficulties but also the technical ones.

Many others has already tried to solve traceability and consistency related problems [17, 18, 19, 20]. The common point was in these concepts to create a technology to solve the related problems. However, the adaptation of these technologies is cumbersome and error prone. Our approach is unique, as it defines only a concept which can be utilized to solve traceability and consistency deficiencies. Thus, companies can tailor to fit maximally to their needs considering their processes, software tools and habits. This generality is the main strength of ALS method.

978-1-5386-4772-1/18/$31.00 ©2018 IEEE

The paper is structured as the following: First the importance of traceability and consistency is discussed together with the classification of ALM systems. Afterwards, the general introduction of ALS approach is presented. This is followed by explaining the application environment together with the used solutions. Finally, the advantages and disadvantages of presented solution is discussed, followed by our future goals.

II. Environment of Application

The main target of this research consists enterprises in the field of medical device developments, but other (typically safety-critical) domains can profit from it as well. The presented idea is most effective for medium or small companies. For large companies it might be too demanding, so in their case it is advised to apply it on team/department level for best result.

As it was already mentioned in the introduction, a product has to fulfil many standards and directives before market launch. The proof of compliance has to be prepared which means a significant burden and needs massive human effort [21]. To make this task easier, to support the everyday work and to help keeping the documentation organised so called Application Lifecycle Management tools (ALM) are used, altogether they are called ALM system. (Often referred as product lifecycle management system which is somewhat different in scope.)

Such system is a collection of software tools dedicated to support a certain field of development (including but not limited to requirement management, test management, process management, workflow management, and many others). It is self-evident that these programs may origin from a single manufacturer or multiple manufacturers. Nowadays, the tool providers are focusing on creating not only dedicated solutions but also integrating them into a system. This way the overall visibility is increased, the navigation is easier and everyone can get efficiently the interested information and only that. Generally speaking, the usability of such systems is improved.

This question is important for a certain vendor to let companies buy most of the products from them. Moreover, the significance of cooperation has been realized by many and this kind of integration can be found more and more often between tools from different providers. This integration possibility and mutual collaboration is a must for effective and precise operation. If the tools in the system can share data with each other and the artifacts (items and their relationships stored in the ALM system) are accessible for each tools requiring it, then we are speaking about a homogeneous network. On the other hand, when the information is isolated and it cannot be reached directly by other tools, then we are speaking about a heterogeneous system. From the description it is clear that almost every practical realization is somewhere in between these categories. In practice, when most of the tools are integrated into the system then it is discussed as homogeneous otherwise as a heterogeneous environment. Companies are now aware that homogeneous systems are more beneficial, yet heterogeneous system still exists. The reason for this can be various, it can be a historical remnant, it can be fear from depending of a single vendor, or they can be just simply unmotivated for improvement.

One of the crucial problems of heterogeneous ALM systems is to provide satisfactory traceability and to maintain consistency in a provable way. Traceability is the expression of decomposition, implementation and reintegration of system artifacts by keeping their dependencies meanwhile.

The importance of traceability is shown well, by showing the fact that most of the important standards are prescribing it. The already mention standard DO-178C [7] in avionics and IEC 62304 [8] in the medical domain requires bilateral traceability together with Automotive SPICE. Other mentionable standards and directives can be found in [21]. Ironically, managers and stakeholders not question the relevance of traceability, but still it is a common opinion among them that it is mostly a burden with little real benefits. Moreover, developers, who can really utilize the benefits of traceability are the ones mostly agitating against it. The reason behind this could be that they are also the ones who are responsible to create and maintain it which is a really demanding task for sure.

Automotive SPICE shall be highlighted as from version 3.0 this is the first current standard prescribing not only traceability but also consistency. Although, it seems to be self-evident to check whether artifacts are not contradicting each other, but the systemic analysis and automatized exploration is not a common practice. More can be read about traceability in [22, 23] from Gotel at al.

It seems to be trivial to solve these problems, at least for the first sight. However, when analysed them more closely the problems are really challenging. Namely, the task to find a relationship between two items is self-evident. On the other hand, proving that there is no need for any connection between two chosen artifacts is really difficult. This existence – none-existence problem was also discussed by Biro at al. [15]. Finding a result for these problems is informative for both the manufacturers and also for certification bodies.

Companies can profit from finding uncovered relationships thus they can not only improve their processes to avoid these mistakes but they can also improve software quality by eliminating these traceability gaps and inconsistencies as source of errors. Moreover, assessors (certification bodies) benefit from this solution as well. They have a limited time to explore the unique structure of database of each manufacturer with different workproducts and technology. Thus, they cannot fulfil completely their role at the moment, namely to find any deviation by only sampling some hundreds of artifacts. On the other hand, with a system responsible for finding any traceability gap and inconsistency they would be able to state without doubt.

In order to give an automatic solution for the above mentioned problems the idea of Augmented Lifecycle Space (ALS) was created [15].

III. Augmented Lifecycle Space approach

The lifecycle space is defined as a set of artifacts together with their relationships which exists in the chosen environment. Practically, this environment means the (restricted) dataset from the chosen tools involved in the investigation. (It shall be highlighted that it is not necessary to apply ALS approach on every artifact, but it can be limited to a subset with special interest. However, this subset shall also be –mostly– complete.)

In the above stated environment the ALS approach shall consists the following steps in a general case [15]:

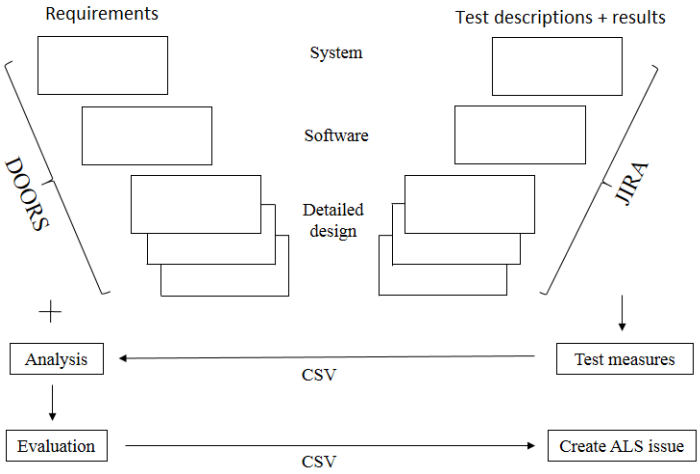

Figure 1. Execution process of analysis

1. Categorize all existing artifacts of the homogenous or heterogeneous tool environment according to the elements of the chosen model containing traceability requirements (e.g. requirement, architecture element, test case, etc.).
2. Analyze the existing relationships (links) and the artifacts in the system and identify those which are missing but should exist according to the traceability requirements of the model.
3. If one of the two artifacts necessary for a required relationship is missing, automatically augment the system with the corresponding artifact whose links will be initially missing of course.
4. Analyze the relationships (links) of the augmented system. If a relationship (link) required by the model is missing, then automatically generate the task of the workflow for bridging the relationship gap.
5. Execute the relationship gap bridging workflow generated in the previous step involving manual intervention if necessary.

As it was demonstrated in [16] it is possible to customize the above shown steps for homogeneous ALM system. Moreover, the result was a set of workflows generated automatically which makes it possible to also fix the identified deficiencies. This application had an ultimate weakness: It has neglected every technology related problems and provided a solution with a single tool which is not ideal for a relevant part of artifacts. Therefore, it was inevitable to present the applicability with multiple connected tools as well.

IV. Heterogeneous Application

Automotive SPICE was used as a base in this research due to its considerations regarding both traceability and consistency. As this is only a demonstration project a complete system was not set up. Instead, only the important steps were modelled. From the original V-model of ASPICE system requirements, software requirements and software detailed design were created in the decomposition phase. As a counterpart, unit tests, software qualification tests and system qualifications tests were created in the integration part.

In order to create a heterogeneous system IBM Rational DOORS and Atlassian JIRA were used. The former one is a mature requirement management tool with still increasing number of users. The latter one is a trending development tool with significantly increasing user base, which is also acknowledged by independent advisory companies [24]. Thus, it seems worthy to analyse their integration possibility with ALS approach.

To fit their original purpose, requirements were all stored in DOORS. System and software requirements were covered with one-one formal modules. For software detailed design three independent formal modules were created. In each of these formal modules (next to the explanatory structure) the same inscriptions were created: a unique name (generated by the program), description of the artefact, an optional comment and the JIRA identifiers of the related test cases are stored information for a single entity. The relationships with the other tool items are created this way which is acceptable as it is unequivocal. The drag-and-drop feature of DOORS is utilized to create the linking between requirements which is a direct link, and the referenced artefact can be reached directly.

The test cases are stored in a similar way in JIRA. One-one JIRA projects are responsible for the system and software qualification test cases and three projects are responsible for the corresponding detailed design artifacts. Here, the test cases are stored as issues, which have a unique identifier (generated by the program), a description containing the test description, a test cases status, a test case alignment (whether it is inbound or out-of-bound test) and also the unique identifier of the referred DOORS objects is also stored.

Thus requirements and test cases alone can be considered as one-one homogeneous system, but the integration of these solution results a heterogeneous one.

A standalone project is created in JIRA which is responsible for the augmentation. This project is responsible for storing the automatically generated workflows which should be followed to fix the traceability gaps and inconsistencies. This project is completely independent of the other projects (excluding the references to other objects) to exclude it from the corporal processes thus decreasing the burden caused by false positive findings or findings with little to no relevance.

Three types of analysis were executed on this system. Requirements were analysed for having no parent and/or child requirements, requirements were checked to have at least one inbound and one out-of-bound test case and it was also checked that the chronology of linked items is suitable (e.g. requirement is modified later than its parent and it is sooner modified than its test case). These checks need information sharing between the two databases which raise some technical problems. (About the implementation of ALS in homogeneous environment and the connection of above mentioned checks to the ALS method are detailed in [16].)

Let it see, how these check can be executed efficiently in the mentioned setup: First of all, DOORS has an own script language called DXL (DOORS eXtension Language). This supports effectively the creation of checks similar to the above mentioned ones. Thus, the backbone of the analysis was implemented via DXL scripts. A sample pseudo code can be read below for finding missing links:

```
Filter f = hasLinks (linkFilterOutgoing,
"AnalyzedModule")      //For      system
requirement
Filter   f   =   hasLinks(linkFilterBoth,
"AnalyzedModule")     //For     software
requirements
Filter f = hasLinks(linkFilterIncoming,
"AnalyzedModule")     //   For   software
detailed design
isEmpty(f)
```

Yet, some information is unavailable from only the DOORS database. For example, the test results are stored only in JIRA and such information shall be shared. In order to solve this, a simple CSV file was created. This file is created by DOORS and it contains the references for every relevant test cases. This file is then overwritten by JIRA and the test results are concatenated to every required test identifiers together with the test alignment. This modified CSV file is sent back to DOORS where the analysis (for each cases) can be now completed. The process is shown on Figure 1. for better understanding.

The found deficiencies trigger the generation of workflows based on the general practice (Fig. 2.). The generated workflow is transited to a state which is necessary to fix the found problem. This way, the stakeholders only need to approve them and let them execute with the developers.

Even from this description it is clear that this approach has weaknesses. First of all, it is inefficient with big databases. In this case the creation, modification and communication of CSV file is demanding. Thus, in real life scenarios other methods are recommended, but in this case it is acceptable for demonstration purposes. Secondly, it would be better to affect directly the workflow. Now, independent workflows are created which should be approved and applied for the system. Practically, it is more beneficial to affect directly the ongoing process and right after the occurrence of a deficiency it should be moved to the previous state to prohibit the injection itself. Furthermore, it would be more user friendly to apply it to a selected subsystem. It is rare to find occasions when the full system can be checked and modified. On the other hand, after certain baselines the newly created or modified objects could be checked assuming that the baseline has no deficiencies. Finally, much better results could have been achieved by homogenising the system.

For this purpose Kovair Enterprise Bus (KEB) could have been used. This middleware can bridge between the two tools thus information sharing would be much easier. Although, KEB is already OSLC based [25], but still other OSLC based solutions should be also mentioned as an implementation option. (OSLC is a linked data based standard with significant support from vendors' side for sharing information between the different components of ALM system.) However, this way a more homogeneous system is created which is not much different from the already analysed case [16].

Each solution (homogeneous system, heterogeneous system, homogenized system) can be found in practice, thus it was necessary to mention application for each.

This paper together with [16] has shown the general applicability of ALS method. It is important to highlight, that these are general, demonstrative description, the solution itself shall be always tailored to the need of the company. The reason for this is that the implementation is highly depending on the used tools, the development processes, the everyday practice and the organization of databases. It is impossible to cover all these unique properties with a single solution.

In the future, we are expecting to combine this idea with the use of formal methods which makes it possible to create more sophisticated analysis and to find deficiencies more effectively. The ultimate goal is to discover every deficiencies and to prove the completeness mathematically in the absence of these. Naturally, some of the later versions should be evaluated with industrial parties to get more feedback from the practical usability. However, due to the sensitive nature of data used in the experiment, prudence is inevitable and it makes the evaluation prolonged.

V. Conclusion and Further Work

Traceability and consistency are key issue in order to provide high quality software. Still, the systematic, proven execution is not a standard even in the safety-critical developments. To get rid of this problem the idea of Augmented Lifecycle Space approach was created and it was successfully applied in a homogeneous ALM system with great general transparency.

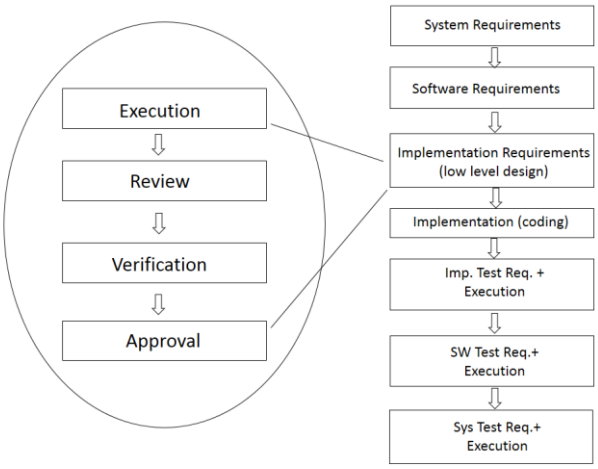

Figure 2. General workflow scheme for development used by ALS method

In this paper an example was shown how this approach can be utilized in a heterogeneous system, where the direct access between every artifact is not guaranteed. A system was presented by using DOORS and JIRA where three analysis were executed: finding missing links, find outdated artifacts and find missing test cases. An example was shown for data sharing between the two databases without homogenizing them. Thus, a practically conceivable application was shown.

This way the applicability of ALS method was demonstrated for heterogeneous and homogeneous systems. In the future, it is expected to further develop these examples to be more usable for companies. Also, it is an expectation to use formal methods to prove mathematically the results and proving the completeness of the corrected system.

ACKNOWLEDGMENT

The authors are grateful for the support of Research and Innovation Center of Óbuda University.

The research reported in this paper 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.

Supported by the ÚNKP-17-3-III-OE-779/52 New National Excellence Program of the Ministry of Human Capacities.

REFERENCES

[1] Cooke-Davies, Terence J.; Arzymanow, Andrew. "The maturity of project management in different industries: An investigation into variations between project management models", *International Journal of Project Management*, vol. 21:(6), pp. 471-478, 2013.

[2] Niazi, Mahmood; Wilson, David; Zowghi, Didar. "A maturity model for the implementation of software process improvement: an empirical study." *Journal of Systems and Software,* vol. 74:(2), pp. 155-172, 2005.

[3] Humphrey, Watts S. "Characterizing the software process: a maturity framework." *Software, IEEE,* vol. 5:(2), pp. 73-79, 1988.

[4] Baldwin, R., Cave, M., & Lodge, M. "Understanding regulation: theory, strategy, and practice". Oxford University Press on Demand, pp. 10-33., 2012.

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

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

[7] Radio Technical Commission for Aeronautics, European Organization for Civil Aviation Equipment. *Software Considerations in Airborne Systems and Equipment Certification.* RTCA/DO-178C, 2012.

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

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

[10] International Organization for Standardization. *Medical devices – Application of risk management to medical devices.* ISO 14971:2012

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

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

[13] Ambler, S.: Tracing Your Design. In *Dr. Dobb's Journal: The World of Software Development* (1999)
http://www.drdobbs.com/tracing-your-design/184415675 (accessed on 29.11.2017)

[14] Cleland-Huang, J., Gotel, O. C., Huffman Hayes, J., Mäder, P., & Zisman, A. "Software traceability: trends and future directions". In Proceedings of the on *Future of Software Engineering* pp. 55-69. ACM, 2014.

[15] Bíró, M., Klespitz, J., Gmeiner, J., Illibauer, C., Kovács, L. „Towards Automated Traceability Assessment through Augmented Lifecycle Space". In: European Conference on Software Process Improvement. Springer International Publishing, p. 94-105. 2016.

[16] Klespitz, J., Bíró, M., Kovács, L. „Augmented Lifecycle Space for traceability and consistency enhancement", Invited session paper: „Junior Cybernetics in Applied Sciences", SMC 2016 - IEEE International Conference on Systems, Man, and Cybernetics, Budapest, Hungary, pp. 3973-3977, 2016.

[17] Avdeenko, T., & Pustovalova, N. "The ontology-based approach to support the completeness and consistency of the requirements specification". In *International Siberian Conference on Control and Communications* (SIBCON), 2015 pp. 1-4. IEEE. 2015.

[18] Kaiya, H., Sato, R., Hazeyama, A., Ogata, S., Okubo, T., Tanaka, T. & Washizaki, H. "Preliminary Systematic Literature Review of Software and Systems Traceability". *Procedia Computer Science*, Vol. 112, pp 1141-1150. 2017.

[19] Demuth, A., Kretschmer, R., Egyed, A., & Maes, D. "Introducing Traceability and Consistency Checking for Change Impact Analysis across Engineering Tools in an Automation Solution Company: An Experience Report". In IEEE *International Conference on Software Maintenance and Evolution* (ICSME), pp. 529-538. IEEE. 2016.

[20] Silva, N., & Vieira, M. "Adapting the Orthogonal Defect Classification Taxonomy to the Space Domain". In *International Conference on Computer Safety, Reliability, and Security* pp. 296-308. Springer International Publishing. 2016.

[21] 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.

[22] 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)". *Software and Systems Traceability*, pp. 343-409., Springer London, 2012.

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

[24] Murphy E. T., West M., Mann K. J. Magic Quadrant for Enterprise Agile Planning Tools, *Gartner*, 2017.
https://www.gartner.com/doc/reprints?id=1-3YWBN1Q&ct=170427&st=sb%255Bgartner.com (accessed on 29.11.2017.)

[25] Arthur G., Ryman; Le Hors, Arnaud; Speicher, Steve. "OSLC Resource Shape: A language for defining constraints on Linked Data." *Linked Data on the Web" (LDOW)* 996, 2013.