# Towards Embedded System Agile Development Challenging Verification, Validation and Accreditation : Application in a Healthcare Company

Clément Duffau, Bartosz Grabiec, Mireille Blay-Fornarino

# Towards Embedded System Agile Development Challenging Verification, Validation and Accreditation

## Application in a Healthcare Company

Clément Duffau
AXONIC and I3S
Université Côte d'Azur, CNRS
Sophia Antipolis, France
duffau@i3s.unice.fr

Bartosz Grabiec
AXONIC
Sophia Antipolis, France
bgrabiec@axonic.fr

Mireille Blay-Fornarino
I3S
Université Côte d'Azur, CNRS
Sophia Antipolis, France
blay@i3s.unice.fr

***Abstract*—When Agile development meets critical embedded systems, verification, validation and accreditation activities are impacted. Challenges such as tests increase or accreditation documents production have to be managed in terms of time and resources. In this paper, we highlight these challenges and present a continuous integration ecosystem that aims to tackle these issues. We report on how this approach has been applied in a research and development healthcare company named AXONIC.**

***Index Terms*—agile development; embedded systems; justification; VV&A; continuous integration**

## I. Introduction

In many domains with a high level of risk of injury or damage to health there is a strong need to ensure that a system satisfies formal quality requirements defined by a certification authority. This is one of the main reasons why Verification and Validation (V&V) activities are followed by justification activities dedicated to produce documents required for Accreditation (VV&A). In the classical V-model, VV&A activities occur at the end of development process. Even if each stage of development requires traceability and production of justification documents, testing is performed only once at the end of the process. Introduction of Agility into a project aims to improve quality of the product and client involvement at constant cost and time. It relies on iterative and incremental development cycles. Consequently, V&V activities performed once in V-model were moved to multiple executions of these activities in agile development. At each development cycle, the amount of testing and justification documents that needs to be managed increases. In order to meet V&V requirements, all tests, including those from previous iterations, must be replayed. The elements of justification must be expanded and modified. This introduces complexity into their management. A consequence is an increase in costs, at least in terms of human resources and length of testing time. However, optimization of the costs of these activities is crucial for companies, especially for R&D entities, which do not have a lot of V&V resources and need to focus on innovation to survive. Moreover, with limited human resources, quality improvement with Agility usage remains to be demonstrated. The remainder of this paper is organized as follows. In the next section we discuss the underlying challenges in this context. We rely on our experience in healthcare systems production within AXONIC, a R&D company. It develops a hardware and software platform to address different pathologies bounded to the nervous system. Sections III and IV present the solutions implemented to meet these challenges. Section V concludes the paper and briefly discusses future work.

## II. VV&A Challenges in Agile Development

### A. Increase in testing activities

Usage of embedded systems is growing very quickly [1]. More and more critical embedded systems are developed *e.g.*, the emergence of self-driven cars and the increase of embedded systems in healthcare. The importance of testing them becomes a big challenge [2]. It is not only the matter of high-level software testing but also of embedded software, hardware and their integrity. But, at the same time, companies need to reduce their time-to-market and consequently their development life-cycle while conforming to accreditation requirements.

Adopting an agile development process in a critical industry still *increases the cost of V&V activities (Challenge C1)*. Kaisti and al. analyze the 12 principles of the Agile manifesto in the context of embedded systems, and underline that "agile methods might offer solutions for embedded software development, but the methods need to concentrate on the embedded domain-specific requirements" [3]. Indeed, embedded systems require tests on each part of the system (*e.g.*, mechanical, hardware, software) and therefore manual or semi-manual V&V activities.

On the other hand, even if the costs can be managed, there is still a *risk of bottleneck in testing process due to the explosion of V&V activities and needed human resources (C2)*. A common way to handle this problem in Agile development is to

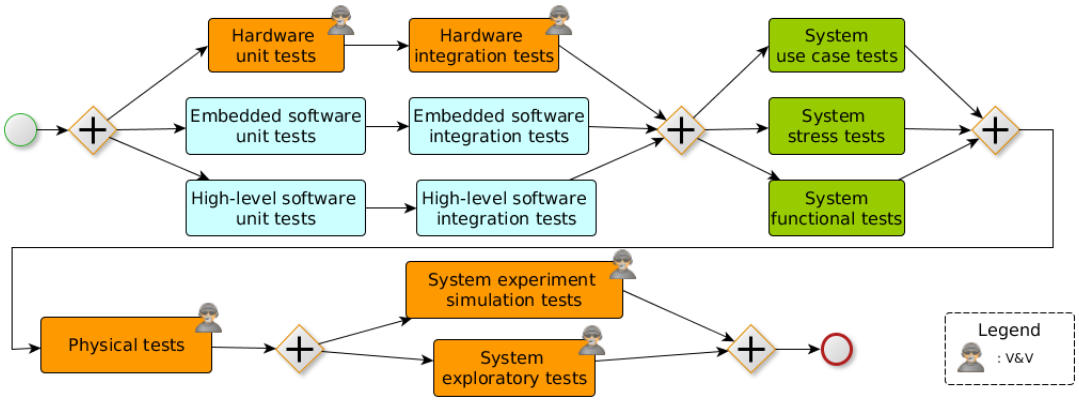


Fig. 1. AXONIC V&V process excerpt

adopt Continuous Integration (CI). CI is an internal practice of development by merging all software artifacts together [4]. In particular, "principle of continuous integration applies as well to testing, which should also be a continuous activity during the development process" [5]. In the context of embedded systems this applies equally to all of their software and hardware components as well as to the whole integrated system. Different techniques to test embedded software tackle these problems. Some of them are platform dependent (*e.g.*, Android emulator, Arduino simulator), others are really difficult to customize to specific hardware like hardware mocking [6]. In a R&D company in the field of embedded systems, the hardware evolves very quickly. The former issues may then lead to difficulties. A dedicated workforce, that is not that of the V&V, is then necessary to maintain the mocking software, which is ultimately very expensive. And what is worse, in a critical context, validation by mocking is insufficient; tests on the end-to-end products are indispensable. Thus, in the trend of the DevOps movement [7], that aims to align the management of the IT infrastructure and the development cycle, it is important to propose global integration processes that avoid these pitfalls.

The Agility increases the importance and quantity of tests. Therefore, it requires an automation of the whole process of integration of software, hardware and end-to-end tests.

### B. Tsunami of justifications for accreditation

Agility is based on successive iterations during which new features are added. The justification activities (*e.g.*, risks identification and mitigation, architectural design reviews, tests results summary) are therefore strongly impacted.

At each iteration, V&V activities are executed and corresponding justifications have to be produced. However, this leads to a multiple revisions of justification artifacts, *e.g.*, input data, technical specifications, design. We can compare it to a tsunami. At the beginning of the project, we are in the epicenter, and with each successive iteration, the mass of artifacts becomes bigger and bigger and is difficult to manage. *Production and management of the justification tsunami* is a challenge *(C3)*. The justification artifacts have to be internally reviewed and signed by quality board to ensure the quality of the products regarding to standards (*e.g.*, ISO 13485 : Medical devices - Quality management systems, ISO 62304 : Medical device software - Software life cycle processes). With Agility, the *review activities surge (C4)*.

### C. Evolution of stakeholders activities

Moreover, Agility requires significant involvement of stakeholders. At the beginning of an iteration, Product Owners (PO) [8] specify new features with the client and decide scope for iteration with technical leaders. At the end of an iteration, POs validate all the features and then deliver the product to the client. This involvement introduces a high dependency to POs. *Overlapping between different POs activities at the end of an iteration leads them to unsustainable pace (C5).*

In the context of AXONIC, the Association for The Advancement of Medical Instrumentation provides a guide to adapt accreditation activities in an Agile lifecycle. The proposed process underlines the multiplication of V&V activities due to their repetitiveness during iterations [9, Figure 4]. Moreover, the neurostimulation platform needs to be adaptive to a pathology and user context sensitive. For example, when obesity is treated with an implanted device, specific parameters have to be taken into account in the software outcoming from clinical studies. To support and ensure consistency of product variability, AXONIC's software platform is designed as a Dynamic Software Product Line (DSPL) [10]. We present the global V&V process in Figure 1 that we follow during each iteration. An iteration is 2 weeks long. For automatic activities, they are launched between 1 and 10 times per day during the iteration. This corresponds to about 500 tests of the integrated system per day. AXONIC POs are clinicians that are in charge of delivery and client relations; they test the products on patients; they study the state of the art in order to justify some results and propose new features. In addition to these external activities, they take care about iteration scoping, write specifications and acceptance criteria, and validate the product before delivery. In the face of the multiple aspects of their work, solutions have to be found to optimize their work.

## III. Continuous Integration From Software to End-to-end Systems

We can automate software V&V activities relying on CI thanks to dedicated platforms [11]. The CI platforms trigger tasks like compilation, testing and packaging. To target embedded systems, dealing only with software tasks is not sufficient. A CI platform has also to take into account tasks on the end-to-end product (*e.g.*, performance tests, safety procedure tests). Answer to this issue with manual testing on some specific parts clashes with the increasing costs (*C1*) and the risk of resource overload (*C2*). Thus, we have to improve the IT infrastructure to reduce the set of manual activities and support the production and execution of end-to-end tests.

In the context of AXONIC, as shown in Figure 1, we moved from several manual activities to a more and more automated testing environment. The blue activities are automated thanks to common software testing frameworks (*e.g.*, Junit, CMock, Unity, Mockito). The orange ones are for the time being manual. We are working to partly automate them thanks to a dedicated home-made framework. The green activities correspond to test activities on the end-to-end product. They deal with physical devices and we aggregate tools to automate the deployment (flashing) of the code on them. Thanks to that approach, embedded software and high level software can be tested in a controlled environment. They are integrated in the CI platform. V&V can now execute test on the complete system or part of it depending on the stage of V&V process.

The overview of this approach is shown in Figure 2. At the bottom of the figure, the *development activities* follow and use common guidelines and tools. These activities correspond to blue activities in Figure 1. In the middle of the figure, *V&V execution activities* corresponding to get and deploy binary code of the embedded software on the hardware platform and then tests are automatically executed by the CI platform. These activities correspond to green activities in Figure 1.

To optimize POs activities (*C5*), we develop a Domain Specific Language (DSL), in the *Test Scenario framework*, that supports PO to define acceptance test scenarios regarding to each new feature. V&V engineers complements these tests using their own DSL. Thus, at the beginning of an iteration, during specification analysis, acceptance tests are defined by PO. Following Test Driven Development approach [12], these tests are executed during the entire iteration and automatically checked for the delivery. This approach reduces ambiguousness of acceptance tests because expressed directly by the PO. We also note that developers are helped by this test approach which further enhances the multidisciplinary team. These activities are placed at the top left corner of Figure 2.

Thanks to this CI ecosystem, we observed reduction of the cost in time and human resources on V&V activities, but we did not conduct yet a complete empirical evaluation. However, how to produce artifacts associated with V&V activities regarding to Accreditation?

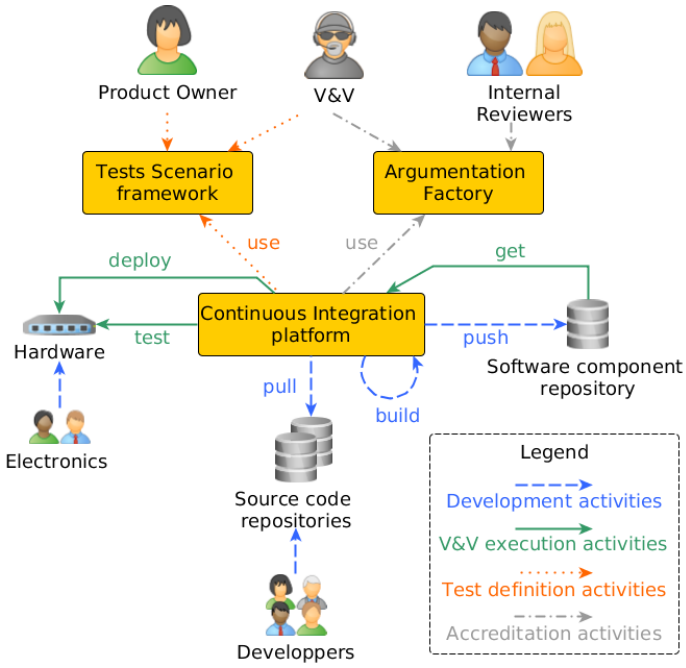


Fig. 2. Continuous Integration (CI) ecosystem

## IV. Continuous Justification Integration

Our CI ecosystem tackles some scalability issues due to the introduction of Agile methods in development of embedded systems. Accreditation requires specific documents to approve a system that must be produced according to V&V activities. Thus, a CI ecosystem worsens justification tsunami (*C3*) and increases review activities (*C4*). In the context of AXONIC, justification documents are traceability matrices between requirements and tests, end of iteration testing reports, bill of materials, etc. For traceability and navigability purposes, all these documents have to be stuck together from ticketing system defining features to test results. This tsunami of documents must be structured to be useful to internal reviewers and, in the future, to external certification authorities. To achieve this structuring, it is necessary to adequately react to V&V activities and to use dedicated justification canvas.

We thus propose to automatically get the test results including all needed environment information, and aggregate all these data in summary documents, *i.e.*, also filter what is essential for accreditation purpose. In the context of AXONIC, we have to keep the latest test results but not the whole set of results during the iterations.

To structure documents, we propose an approach based on argumentation research in critical domains. In [13], Polascek proposed a new kind of diagrams: the argumentation diagrams. These diagrams are derived from the argumentation model outlined by Toulmin [14] to take into account critical systems. They encapsulate reasoning steps followed by experts to achieve an objective, regarding to a standard. We take this new concept and design a tool: the Argumentation Factory [15] that is able to structure justification elements in a diagram. An

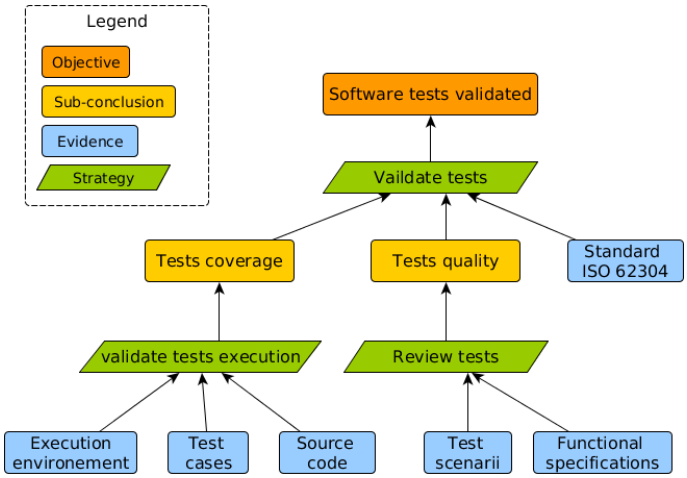


Fig. 3. Example of Argumentation Diagram associated to high-level software unit test activity

example of these diagrams in the context of AXONIC is given in Figure 3. This diagram represents the quality justification of high-level software tests in term of code coverage and features coverage.

The main tool for accreditation purpose is the Argumentation Factory plugged in the CI ecosystem on each V&V activities as shown in Figure 2 in the top right corner. We developed a plug-in for the CI platform that permit to attach justification context to each testing activity. Thanks to these contextual meta-data, test results justification are aggregated into a diagram. These diagrams are not just a way to aggregate reasoning but also explain reasoning to an expert, and discuss about the relevance of certain tests. In this context, quality reviews can lead to detect false negative test cases, replay tests, or add a manual test. Thanks to this approach we keep human in the loop only for what matters, where human expertises is formally required *(C4)*.

These structured diagrams are used to generate the justification documents needed regarding standards. These diagrams capture a lot of details and are an easy way to have the big picture of the state of V&V of a product. At the end of an iteration, for justification purpose, we focus only on V&V results that matters for accreditation purposes. Argumentation Factory supports these kinds of filter policies for justification documents, but also keeps the whole story in the diagram [15].

Another difficulty with Agility and accreditation is the gap between iterations. Evolution of the system implies evolution of the justification documents and consequently leads us to the question of justification versioning, from revision of requirements to non-regression tests and new tests. This is one of the challenges to optimize the quantity of quality reviews by supporting a controlled incrementality. This point remains one of our short-term perspectives. Also, in the next future, the usefulness for end-user should be demonstrated through metrics such as the time to find a specific document or survey to measure the improvement of confidence with our approach.

## V. Conclusion and Perspectives

In this paper, we discussed the issues related to the adoption of agility in the construction of critical embedded systems. In particular, we highlighted the increase in testing and specification activities, and therefore in the associated justifications. To overcome these difficulties, we have enriched a Continuous Integration ecosystem. The originality of this approach lies in the joint coordination of end-to-end product tests and the automatic production of justification documents using argumentation diagrams. In the context of the AXONIC company, which produces neurostimulation systems, we have multiplied the number of all types of tests, including acceptance tests, by 10, in terms of execution bandwidth, while keeping a constant number of dedicated human resources.

We are currently working to enrich this ecosystem by extending the set of automatic test activities. We intend to exploit the very construction of the argumentation diagrams to check their compliance with the standards related to accreditation. For this reason, we define domain specific argumentation canvases and exploit them through the Argumentation Factory. Responsiveness to changes is essential in agile development but complex in a VV&A approach. We already manage addition of new requirements in argumentation diagrams, but, in this case, it is also a question of management of modifications or removals of requirements. The versioning of the argumentation diagrams is a track that we are considering.

## References


[1] X. Mosquet, M. Russo, K. Wagner, H. Zablit, and A. Arora, "Accelerating innovation: New challenges for automakers," *The Boston Consulting Group*, 2014.

[2] B. Broekman and E. Notenboom, *Testing embedded software*. Pearson Education, 2003.

[3] M. Kaisti, V. Rantala, T. Mujunen, S. Hyrynsalmi, K. Könnölä, T. Mäkilä, and T. Lehtonen, "Agile methods for embedded systems development-a literature review and a mapping study," *EURASIP Journal on Embedded Systems*, vol. 2013, no. 1, p. 15, 2013.

[4] M. Fowler and M. Foemmel, "Continuous integration," *Thought-Works, http://www.thoughtworks.com/ContinuousIntegration.pdf*, p. 122, 2006.

[5] G. Booch, "Object oriented design with applications. redwood city," 1991.

[6] M. Karlesky, G. Williams, W. Bereza, and M. Fletcher, "Mocking the embedded world: Test-driven development, continuous integration, and design patterns," in *Proc. Emb. Systems Conf*, 2007, pp. 1518–1532.

[7] M. Httermann, *DevOps for developers*. Apress, 2012.

[8] K. Schwaber, *Agile project management with Scrum*. Microsoft press, 2004.

[9] A. T. AAMI, "Guidance on the use of agile practices in the development of medical device software," *Association for the Advancement of Medical Instrumentation, Arlington, VA*, 2012.

[10] S. Hallsteinsen, M. Hinchey, S. Park, and K. Schmid, "Dynamic software product lines," *Computer*, vol. 41, no. 4, 2008.

[11] V. Pecanac, "Top 8 continuous integration tools," *Dzone / DevOps Zone, https://dzone.com/articles/top-8-continuous-integration-tools*, 2016.

[12] K. Beck, *Test-driven development: by example*. Addison-Wesley Professional, 2003.

[13] T. Polacsek, "Validation, accreditation or certification: a new kind of diagram to provide confidence," in *Research Challenges in Information Science*. IEEE, 2016.

[14] S. E. Toulmin, *The uses of argument*. Cambridge University Press, 2003.

[15] C. Duffau, C. Camillieri, and M. Blay-Fornarino, "Improving confidence in experimental systems through automated construction of argumentation diagrams," in *ICEIS 2017*, vol. 1, 2017, pp. 495–500.