ISSN: 2583-9888
DOI: doi.org/10.51219/JAIMLD/suresh-sureddi/375

URF PUBLISHERS
connect with research world

Journal of Artificial Intelligence, Machine Learning and Data Science

https://urfpublishers.com/journal/artificial-intelligence

Vol: 1 & Iss: 1

Research Article

Adopting Agile Methodologies and Frameworks in Automotive Industry

Suresh Sureddi*

Citation: Sureddi S. Adopting Agile Methodologies and Frameworks in Automotive Industry. J Artif Intell *Mach Learn & Data Sci 2022*, 1(1), 1727-1731. DOI: doi.org/10.51219/JAIMLD/suresh-sureddi/375

Received: 02 March, 2022; **Accepted:** 18 March, 2022; **Published:** 20 March, 2022

***Corresponding author:** Suresh Sureddi, USA, E-mail: ssureddi@gmail.com

ABSTRACT

The automotive industry is going through a significant transformation due to advancements in technologies like connected cars (V2X technologies), artificial intelligence, autonomous driving and cloud technologies. Due to increasing complexity and functionalities, industry is moving to the concept of software defined vehicle. However, many automotive OEMs are still following the traditional process for building the software for ECUs (Electronic Control Units) over a period of 2-to-3-year life cycle. Especially, in an ECU (Electronic Control Unit), for e.g., Head unit, ADAS (Advanced driver assistant system) or telematics, there is a huge software with numerous requirements and functionalities. Cost of identifying and fixing the defects close to or post SOP (Start of production) involves huge cost and causes program delays. In fact, this would have a significant impact causing delays to vehicle builds. OEMs shall adopt to agile methodology along with the Tier-1 supplier and the product should be built and tested continuously to ensure delivery of a high-quality product. This paper details the pain areas experienced by OEMs due to following the traditional practices and challenges to adopt to agile for software development and then details the benefits of using AGILE and hybrid models to overcome those pain areas and challenges.

Keywords: AGILE, SCRUM, Waterfall, Hybrid model, Automotive, autonomous, cloud, V2X

1. Introduction

With the rise of connected and autonomous vehicles in the automotive industry, Software defined Vehicles (SDV) have become more common. But in automotive industry, it was always believed that software development must be in sync with hardware development and that was the main reason for going ahead with a software development plan which is driven by vehicle build milestone plan instead of following an independent plan which follows agile methodologies. Also, in today's world, with the increase in complexity of the software, need to shorten software release cycles and pressure to cut developmental costs had led to a situation where traditional plan driven developments are failing. Also, continuous updates will become essential for future systems because vehicles are in use for 10 to 15 years on average and information technology is rapidly changing. As a result of rapid change in information technology, Software-over-the-air updates have emerged as an efficient way, through which software updates can be released to the vehicles over-the-air network.

The current development processes are inefficient because of the characteristics of the V-model, which is the current state of the art model for the development of automotive software components. V-model can be considered as an extension of waterfall model, in which the execution of processes happens in a sequential manner.

The V-Model is characterized by a linear development approach, in which implemented processes follow an iterative procedure on system and subsystem levels and incremental approaches on component and unit levels. This implies that at the beginning of the development project, the requirements must be detailed and complete. After the development, the software is tested and validated against the specific requirements in a sequential order[1]. Of course, the late changes are being handled through DCR (Design change requests) process, but the number of DCRs are overwhelming during the later stage of the development cycle which affects the quality of the product.

Due to the above-mentioned reasons, there is a need for OEMs and suppliers to move out of traditional processes, methods, tools and adopt to Agile software methodologies. Agile methodology helps to cope with changing requirements, shorter time to market and faster release cycles. So far, Agile methodologies were successfully implemented in other industries. However, agile methods often need to be adopted to a specific context and are in regulated domains, such as automotive, only used to a certain extent or not at all[2]. The most used and fastest growing is Scrum. A study that the fear of the unknown or lack of understanding of the new processes, as well as the fear of running into regulatory problems and losing certifications, are providing barriers that constrain the adoption of agile methods in the automotive domain[2]. This paper first discusses the challenges with traditional models and to the adoption of agile methods in automotive software development and then details the solutions that have been developed to overcome the barriers for an adoption of agile methods in automotive software development.

1.1. Traditional automotive software development:

In automotive software development, the process starts with the specification of the desired complete vehicle characteristics in a downward movement on the left side of the V, called component specification. Based on this component specification, SRS (Requirement), SyARS (System architecture requirement specification), design and development are performed in sequential order as shown in left side of the below diagram.

And then the validation against the specification is done in a hierarchical order as shown on the right side of the below picture. (Liu et al., 2016). Agren et al. (2019) state that there are many safety and legal concerns in automotive software development, which makes the decomposition of requirements and testing necessary, nevertheless can the usual way, to do it upfront development, cause preventable delays[3].

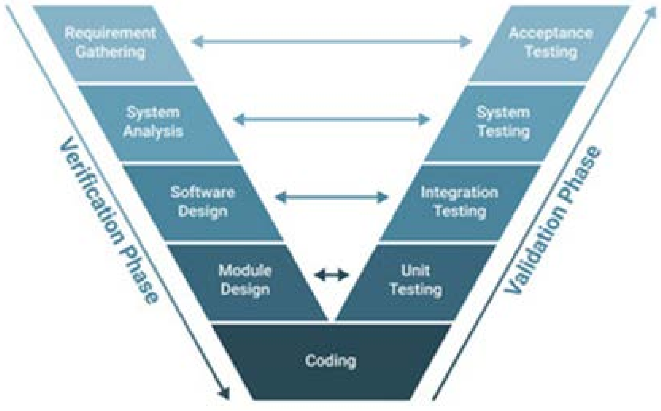

Figure 1: Software life cycle V-Model.

1.2. Agile for Automotive Software development

There are many Agile methodologies which include Crystal methodologies, Dynamic Software development method (DSDM), Feature-Driven Development, Lean software development, Scrum and Extreme programming. One study[2] indicated that agile methods - Continuous Integration, Test Driven Development, Feature Driven Development, Extreme Programming, Scrumban, Kanban and Scrum were all used in an automotive context. Also, it explains that so called hybrid models were used, which combine agile methods with traditional software development processes[4]. Since Kanbhan is more suitable for debugging and maintenance type projects, this article focuses on Scrum, Continuous Integration and hybrid models which are advised for software development process.

1.3. Scrum

Scrum is one of the fast growing and popular frameworks of several agile frameworks, which is extensively used by IT companies. It typically consists of a product owner, scrum master and a development team. The Product Owner gathers all the information on what should be produced from customers or end-users, as well as other stakeholders and translates this into a prioritized list, to assure the team is working towards the desired end goal. The Scrum Master is responsible for supporting and guiding the team towards a successful delivery of work items by following the scrum framework. The Development Team is responsible for the development of the product, under consideration of the Product Owner's requests. Product development is organized by Scrum into several iterations, called sprints.

Sprints are time-boxed typically with the same duration and are driven by the objective to deliver a product increment. As soon as a sprint has come to an end, a new sprint starts. The sprint cycle starts with a stakeholder meeting in which the project vision is agreed on and the Product Backlog is created and prioritized by the Product Owner. The Product backlog is a list of requirements that is prioritized by business value. The sprint cycles begin with a sprint planning meeting to decide which requirements of the Product Backlog will be included in the sprint. During the sprints, there are daily stand-up meetings with 15-min maximum duration, to synchronize the team members. In these meetings the team members share what they have done so far and if they are facing difficulties with something. Towards the end of a sprint, stakeholders and the development team conduct a sprint review meeting to validate the increment that has been delivered and assure the Product Owner approves it. The last step of a sprint is the sprint Retrospective Meeting, which provides an opportunity to the Scrum team to reflect and reevaluate their performance.

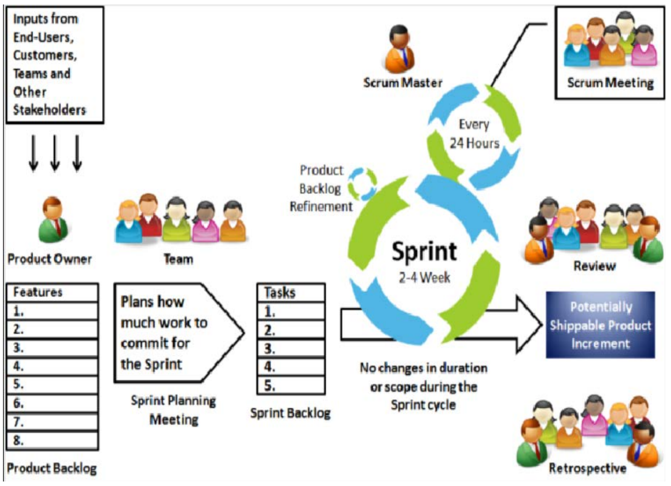
Inputs from End-Users, Customers, Teams and Other Stakeholders
Scrum Master
Scrum Meeting
Every 24 Hours
Product Backlog Refinement
Product Owner
Team
Sprint
2-4 Week
Review
Features
1.
2.
3.
4.
5.
6.
7.
8.
Product Backlog
Plans how much work to commit for the Sprint
Sprint Planning Meeting
Tasks
1.
2.
3.
4.
5.
Sprint Backlog
No changes in duration or scope during the Sprint cycle
Potentially Shippable Product Increment
Retrospective

Figure 2: Scrum Software Development Process Phases.

1.4. Continuous Integration

Continuous Integration has become popular in software development as one of the Extreme Programming practices. Continuous integration implies that software developers integrate code frequently into a central repository at least daily[5].

Continuous Integration helps to improve the quality and speed at which automotive software-based innovations are delivered to the customers. the incremental integration of smaller work products can replace a larger and more complex final integration and helps to achieve transparency about the finished content of the release as well raising awareness of dependencies.

1.5. Hybrid models

Hybrid model is considered as any combination of agile or traditional (plan-driven) approaches that an organizational unit adopts and customizes to its own context needs[6]. As the automotive domain is dealing with safety-critical software, it is strictly regulated. In line with this a common practice for software development is the V-Model. This situation has led to the creation of hybrid models that focus on speeding up the development phase and implement the best principles of agile methods like Scrum inside the V-Model methodology to assure the required level of quality and safety[7].

As Automotive Software Performance Improvement and Capability dEtermination (ASPICE) is a major software process development standard for car manufacturers, it builds the basis on which automotive software development processes are assessed, in terms of software and process quality[8]. In line with this, there are so called traceability matrices which show to what extent the best practices of automotive SPICE are covered within hybrid models[9].

2. Reasons for reluctance to adoption of Agile methods in Software development

This section provides insights into potential constraints for adopting agile methodologies in automotive software development:

2.1. Challenges for adoption of Scrum in Automotive Software Development:

In this context, the main challenges identified are, safety related issues organizational cultural challenges and knowledge related challenges:

2.1.1. Safety related issues: the high criticality level of in-vehicle software results in Safety-related challenges, which includes life-critical as well. The quality assessment standard Automotive SPICE (A-SPICE) is commonly used by automotive manufacturers[10] to assure the high level of quality for reliability, functionality, efficiency and other characteristics. In line with this, multiple barriers when applying A-SPICE to Scrum are recognized. Among these barriers were the assurance of a correspondence between agile process and A-SPICE software engineering processes. One literature state that in regard of Scrum this means that, on the one hand, legal requirements, standards and production requirements must be taken into account and, on the other hand, arrangements with software developers, who urge to act more freely without being restricted by guidelines, has to be made[11]. Additionally, one research points out that scaled agile development of safety-critical systems is dependent on traceability and the ability to prove regulatory compliance at any point in the development process[12].

2.1.2. Knowledge related challenges: Scrum development teams need to have effective and efficient testing processes, as well as defect detection mechanisms. It shows that the real projects implementation showed that insufficient experience of developing manual and automated test cases results in escaping defects. Those defects are discovered afterwards by the system team, but with a delay of potentially several iterations until the complete system is integrated and tested. The new discovered critical bugs then need to be fixed as soon as possible, which affects the planning for the following iterations, so that a domino effect can be created that may affects the objective of the entire program as well.

2.1.3. Organization challenges: It's believed that Scrum methodology can lead to obsoletion of some jobs, for example, team lead role is not needed anymore because scrum master and product owner manages the deliverables along with the development team.

Furthermore, a study conducted at the Porsche AG, has shown that decisions are still made at higher levels of hierarchy and the development team does not have enough decision power. This is contrary to the idea of Scrum, since the development team is supposed to know best, what decisions are appropriate.

Also, at OEM side, there is a dependency on cross functional teams in a large organization, where the response is typically slow, but the scrum expects quick feedback and action.

2.2. Challenges to adopt to continuous integration methodology in automotive SW development

Challenges in this methodology are mainly due to safety related challenges and cross functional collaboration.

Safety related challenges: It is recognized that, in a large-scale agile software development project, there need to be mechanisms, which always ensure a certain level of functional quality, even if developers are autonomously integrating and changing software code.

2.3. Collaboration challenges

Due to the V-model, which is implicitly suggested by the ISO 26262 functional safety standard, the overall development organization is split into different groups. These different groups, for example divided into different competences, can cause a silo effect that hinders information sharing and synergy. Across organizational boundaries, collaboration challenges derive from the issue that many stakeholders must be considered, including suppliers. Another constraining factor in this context is that current collaboration models between automotive manufacturers and suppliers are commonly based on written specification documents. However, even though strict contract-based collaboration is constraining inter-organizational Continuous Integration, contracts facilitate negotiations between different organizations.

3. Solutions

3.1. Solution for scrum limitations

As A-SPICE is the most popular quality assessment framework for automotive software development, one research has defined hybrid processes incorporating both agile and waterfall aspects[10], which means certain aspects are defined by automotive SPICE and other aspects by Scrum. The goal of this approach is to benefit as much as possible from the increased flexibility and increased development speed of Scrum, but at the same time to assure that legal requirements are met.

(Figure 3) provides the process overview of the hybrid process approach. While the Planning (preparation sprints) and deployment (release sprints) are defined using waterfall processes, to ensure a high level of quality, the development sprints are defined on a Scrum basis. This means the creation of the software requirements and the architecture design are conducted within the preparation sprint. Afterwards a flexible development sprint starts, which is based on Scrum and contains first unit and functional tests. If the retrospective meeting is satisfactory, the release sprint starts and the software is tested based on testing requirements that are recommended by A-SPICE. Also, the issue of insufficient documentation is solved since the creation of minimum required documents starts with the development sprint and is completed within the release sprint.

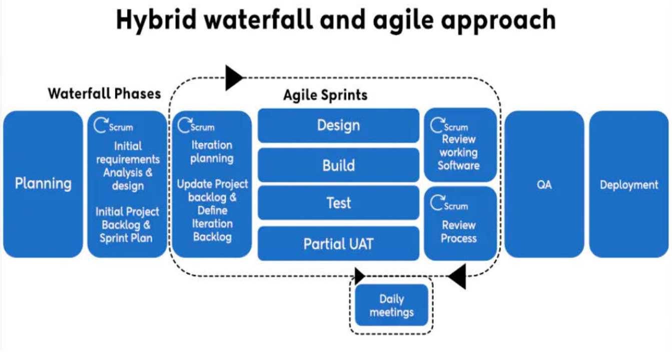

Figure 3: Overview of Potential Hybrid Process[17].

One article provides a way to approach the challenge of additional qualification phases to fix remaining bugs or to finish functionalities that had been planned for the previous sprint and therefore the planned results for the next sprint cannot be fully achieved[11]. The approach that is suggested by Marner et al[11], focuses on a definition of done (DoD) and the incorporation of time-boxed sprints, to increase transparency regarding the content that was finished within an iteration. Another aspect of this approach is an assessment of the status quo at the end of each sprint and the creation of a planning for unfinished requirements for the next sprint. Additionally, the creation of transparency also involves the conduction of retrospectives together with relevant stakeholders and dependent projects. Regarding the issue of the creation of adequate test cases, one study proposes to enable knowledge sharing by exchanging members of the system teams with members of the Scrum teams, so that they can teach their colleagues how to write meaningful manual and automated test cases on team level and how to automate and execute the regression tests[13]. The objective of this approach is that each agile team member will have the competency and necessary skills for system testing as well.

Furthermore, multiple studies suggest adopting agile methods, like Scrum, with an incremental, stepwise approach rather than a big bang approach. One study[2] elaborates that a stepwise approach helps to build acceptance, but another crucial component is a common goal for a team, which has been found to enhance the team moral, productivity and motivation.

3.2. Solutions for continuous integration in Automotive Software Development

For continuous integration in a large-scale agile software development project, an essential element is to maintain a master repository[14].

In this case, a staged development process is recommended, where developer creates a small, short living, development branch and does the changes in the branch. After that the developer initiates a pull request to take over the changes in the master repository. Once the pull request has been triggered, two stages of automated quality gates are triggered, which are a check and a gate. In order to pass the check, at minimum one manual review by another developer has to be conducted. After that, the change is either declined with comments that request an improvement or the change is approved. When the review and all automated tests, which are performed within the check, are acceptable, the gate is triggered and performs additional, more detailed, tests in a virtual merge with the master repository. Finally, if the automated tests of the gate are also acceptable, the change will be merged into the master repository. This process avoids that people commit changes into the repository without any quality checks.

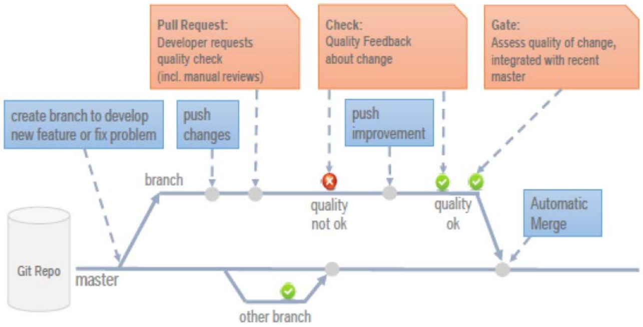
Pull Request:
Developer requests quality check (incl. manual reviews)
Check:
Quality Feedback about change
Gate:
Assess quality of change, integrated with recent master
create branch to develop new feature or fix problem
push changes
push improvement
branch
quality not ok
quality ok
Automatic Merge
Git Repo
master
other branch

Figure 4: Continuous Integration process with Quality Gates (Schlichthärle et al., 2020).

One study suggests multiple guidelines to keep continuous delivery safe[16]. The first guideline advice to perform iterative safety analysis parallel to development, as an up-to-date safety analysis is considered the first prerequisite for continuous safety testing. The objective of the second guideline is to reduce the manual work, in regard of safety case generation, to a minimum, by implementing automated safety test execution and generation. The third and fourth guideline require to handle safety analysis and its results as every other artifact that is required for a build, which means that the analysis must be stored in a central repository in a versioned manner. In that way every change in the analysis can be tracked. Furthermore, there has to be a safety test in every build, to ensure that every build that might be delivered to production has passed a safety check, so that potentially unsafe software is not deployed.

One research concludes that even though strict contracts are providing stability and leverage to the parties, more flexible contracts, so called agile contracts, are needed to improve the supplier collaboration. Furthermore, studies indicate that transparency, which is, in this case, the degree of information that is shared among the organizations operating in the same value-chain, is an enabler for inter-organizational Continuous Integration[16].

4. Conclusion

The objective of this paper is to provide an overview of current challenges of traditional SW development methodologies and adoption to Agile. Also, this paper proposes the solution approaches for the adoption of agile methods in automotive software development. Out of several agile methods, Scrum and Continuous Integration have been found to be applied most frequently in automotive software development[4]. There are various safety related constraints, amongst other things, safety standards like the ISO26262 and the implicitly suggested automotive SPICE quality assessment framework, which demand various requirements and tests[10]. As the adoption of agile methods also involves the people, who may have to operate in a different way than before, also organizational cultural barriers have been discovered. In regard of this, studies indicate that managers are hesitant to provide the operative levels with sufficient decision authority, even though this is needed for agile software development[11]. Nevertheless, there are also various solution approaches to adapt automotive software development processes and incrementally change the corporate culture. In general, the literature for safety challenges of Scrum suggests creating hybrid structures, which are a mix of traditional waterfall processes and agile methods, to make use of the benefits of Scrum, but also ensure that safety requirements are met[10].

As Continuous Integration in automotive software development can only be used under the condition of a guarantee for a certain level of software quality, literature suggests implementing quality gates that ensure that software changes are checked and approved, before they are added to the master repository, which is the stable basis for the software. Next to that, collaboration was also stated to be an important aspect for the efficient usage of Continuous Integration. In that regard, it was advised to increase the level of transparency, in form of shared information, between organizations that operate in the same value chain, as well as the introduction of agile supplier contracts.

Long story short, agile methods cannot be used by the book, but have to be adapted to the automotive software development environment, especially to functional safety standards such as the ISO26262. Even though there is not a strict development model, it is important that at the end of each iteration, there is a product increment, which fulfills specific requirements and is potentially shippable. The literature on agile automotive software development provides different solutions to ensure quality, regulatory compliance, as well as recommendations to improve collaboration internally and externally. Additionally, an important aspect, which seems to be understated in scientific literature, but was emphasized by industry experts, is to establish an organizational culture that understands agile not only as a process but as a mindset. This mindset is anchored within the core values of the agile software development idea and is essential to create an added value for automotive software development, by enabling more flexibility and shorter software release cycles.

5. References

1. Bach J, Langner J, Otten S, Holzäpfel M, Sax E. Data-Driven Development, A Complementing Approach for Automotive Systems Engineering, 2017.
2. Diebold P, Theobald S. How is agile development currently being used in regulated embedded domains? Journal of Software: Evolution and Process, 2018;30:1935.
3. Agren SM, Knauss E, Heldal R, Pelliccione P, Malmqvist G, Boden J. The impact of requirements on systems development speed: a multiple-case study in automotive. Requirements Engineering, 2019;24:315-340.
4. Kugler Maag CIE. Agile in Automotive - State of Practice, 2015.
5. Ståhl D, Bosch J. Modeling continuous integration practice differences in industry software development. Journal of Systems and Software, 2014;87:48-59.
6. Kuhrmann M, Diebold P, Munch J, Tell P, Trektere K, McCaffery F, Prause CR. Hybrid Software Development Approaches in Practice: A European Perspective. IEEE Software, 2019;36:20-31.
7. Takahira RY, Laraia LR, Dias FA, Yu AS, Nascimento PTS, Camargo AS Jr. Scrum and Embedded Software development for the automotive industry. Paper presented at the PICMET 2014 - Portland International Center for Management of Engineering and Technology, Proceedings: Infrastructure and Service Integration, 2014.
8. Diebold P, Zehler T, Richter D. How do agile practices support automotive SPICE compliance? Paper presented at the ACM International Conference Proceeding Series, 2017.
9. Hantke D. An Approach for Combining SPICE and SCRUM in Software Development Projects, Cham, 2015.
10. Komiyama T, Konno S, Watanabe T, Matsui S, Kase M, Igarashi I. Improvement of Agile Software Development Process Based on Automotive SPICE: A Case Study. In: Vol. 1060. Communications in Computer and Information Science, 2019;518-531.
11. Marner K, Theobald S, Wagner S. Real-life challenges in automotive release planning. Paper presented at the Proceedings of the 2019 Federated Conference on Computer Science and Information Systems, FedCSIS, 2019.
12. Steghöfer JP, Knauss E, Horkoff J, Wohlrab R. Challenges of Scaled Agile for Safety-Critical Systems. In: Vol. 11915 LNCS. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2019;350-366.
13. Sandu IA, Salceanu A. System Testing in Agile SW Development of the Electronic Components Based on Software from the Automotive Industry. Paper presented at the 2019 11th International Symposium on Advanced Topics in Electrical Engineering, ATEE, 2019.
14. Schlichthärle S, Becker K, Sperber S. A Domain-Specific Language Based Architecture Modeling 16, 2020.
15. Vost S, Wagner S. Keeping Continuous Deliveries Safe. Paper presented at the 2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C). Approach for Safety Critical Automotive Software Systems, 2017.
16. Van Der Valk R, Pelliccione P, Lago P, Heldal R, Knauss E, Juul J. Transparency and contracts: Continuous integration and delivery in the automotive ecosystem. Paper presented at the Proceedings - International Conference on Software Engineering, 2018.
17. https://www.pmi.org/learning/library/outcomes-hybrid-approach-waterfall-environment-5839