PSAA AerospaceEurope Conference2021
WARSAW, POLAND | 23 -26 NOVEMBER 2021
CEAS Council of European Aerospace Societies

Application of Agile Approach for Development of the Avionics Safety Critical Systems

Pawel Zakrzewski[1], Janusz Narkiewicz[2]

[1]GE Aviation
[2]Warsaw University of Technology

Abstract

This paper demonstrates one of the methods of incremental and iterative product development. It outlines the main challenges and their solutions for Agile application in safety critical systems development at aeronautical domain. Achieved results stem from industrial practice and have been validated during avionics projects related to cockpit displays development in GE Aviation.

Keywords: Avionics, Software, Agile, Safety Critical

1. Introduction

Over last 10 years Agile methodology became very popular project management approach, mainly for the software development. The Agile method is the iterative process of delivering the product in stages to allow collection of customer feedback that will be implemented later. It does not require deep customer involvement to establish the requirements at the first stage of the process.

It proved to be effective in the rapidly growing digital industry. However, so far it has not been applied successfully for development of the safety critical and highly regulated systems, which are common in aerospace industry.

Agile application for safety critical systems development would improve change management, quality (inspection) and provide better control, sustain engineering team hands-on contact with SW development and code writing. It is also cost effective.

However, Agile frameworks do not address the notion of requirements traceability [1], focusing mostly on the delivery of functional features. As such, Agile methodology does not identify documentation creation as an essential part of the project while it is a necessary part of the avionics product certification. Also, the typical rate of requirements change in safety-critical domains, seems to be much lower than the rate of change seen in other industries. For this reason, popular frameworks seem to have limited application for the safety critical aviation products.

Almost all avionics software is qualified as safety-critical and is developed according to the DO-178C standard. This standard defines the goals to be achieved and the requirements that should be met, as a necessary condition for the final system certification [2]. However, it doesn’t define the process of the software development

Creation and implementation of Agile framework that addresses all the safety critical software requirements and limitations, that is fully compatible with existing regulations and approvable by certification authorities would be a breaking point, for the aerospace industry. According to a successful track record in digital industry, Agile methods would potentially allow to reduce time and cost of the new products development which are the key constrains for the new technologies’ introduction in avionics.

2. Frameworks and methods considered

2.1 SCRUM

Scrum is the most popular framework adopted by the digital industry and is an essential representation of Agile fundamentals. It defines components of the process, teams (with roles assigned to individuals), events, artefacts and rules. Every component of the framework has a defined role and should be executed properly to successfully develop the product in a consistent way. The development work is broken down into Sprints (figure 1) - the goals that need to be achieved during each iteration, in timeboxed periods no longer than one month [3]. Progress is tracked daily, in the form of short meetings focused on defining daily goals and removing potential obstacles. Every Sprint is summarized by the Sprint review where the accomplished work is demonstrated.

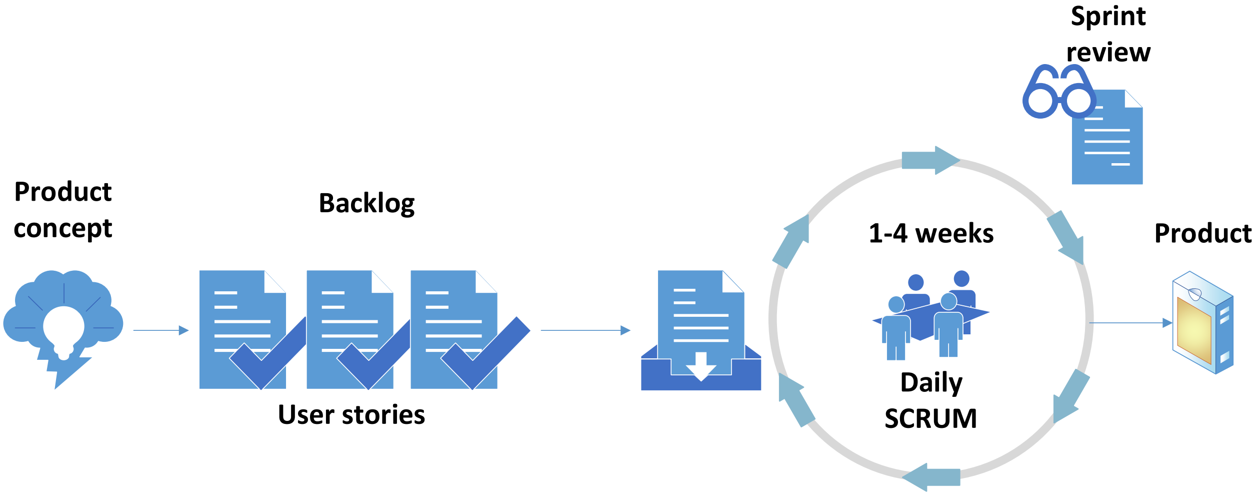

Figure 1 – SCRUM

2.2 The V-model

Software development lifecycle may be also visually represented by the V-model (figure 2). Starting from requirements every step down of the left side of the V provides higher level of details in definition of the product. Implementation phase concludes the requirements and architecture design by writing the code and implementing it within the dedicated hardware [4]. The right side of the V represents the verification and validation process in the lifecycle for the corresponding phases on the left side. The V-Model is a main framework of the ARP4754A Guidelines for Development of Civil Aircraft and Systems [5].

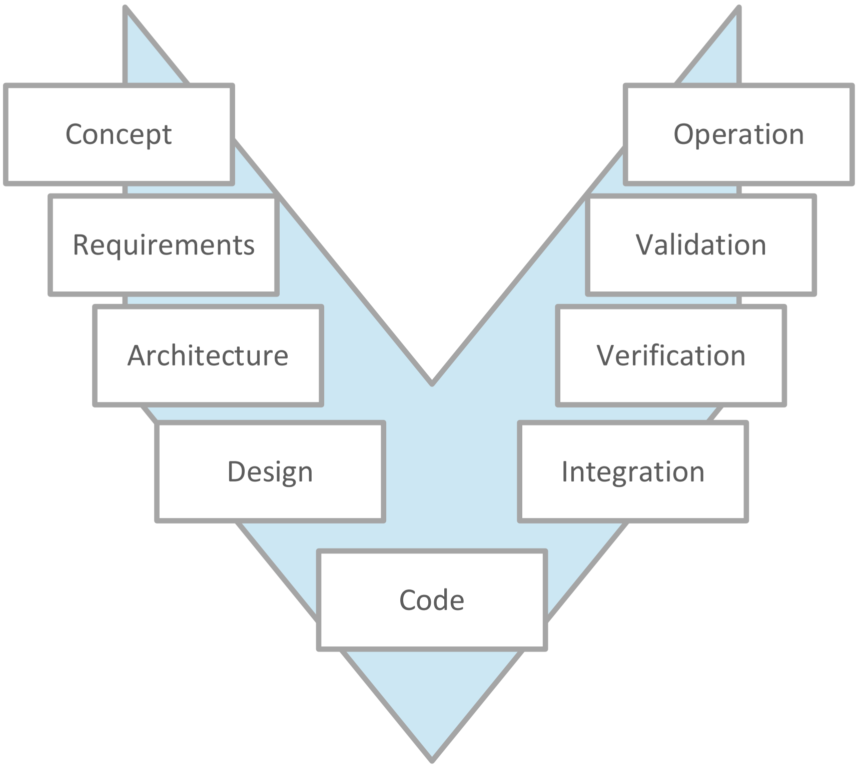

Figure 2 – V-model

2.3 Iterative and incremental development

The background of Iterative and incremental product development is that the system is considered as a composition of parts, and each part is then subjected to a full development cycle – requirements, design, build, test [6]. With every iteration, the system develops incrementally, growing towards the final product. Each iteration creates the opportunity for design modifications and adding new functionalities and should be considered as a waterfall process or V-model within the scope of each iteration. Comparing Iterative and incremental development and Agile based methods both approaches are focused on feedback from each iteration and continuous improvement of a product as well as improvement in team collaboration and process.

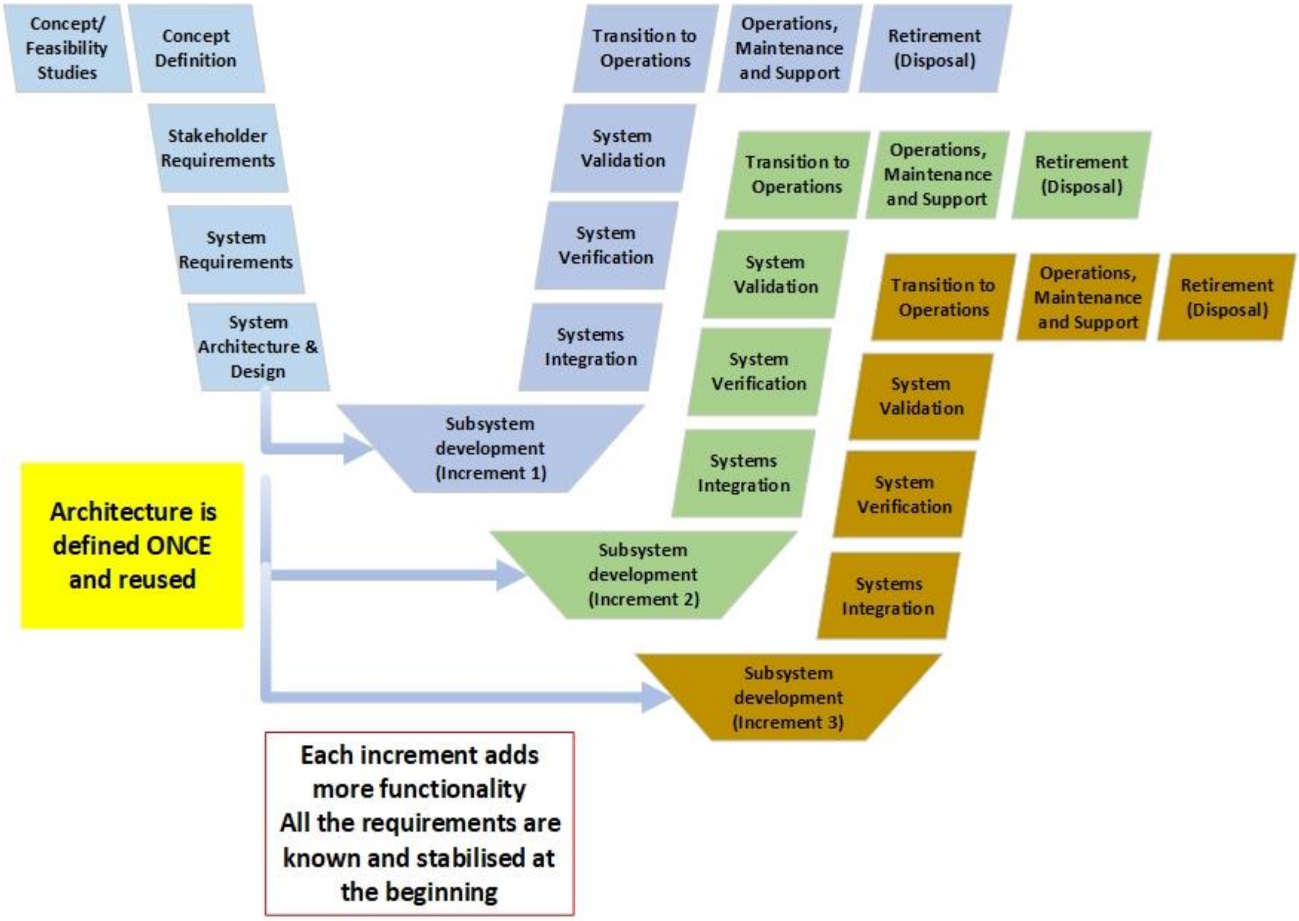
Concept/ Feasibility Studies
Concept Definition
Stakeholder Requirements
System Requirements
System Architecture & Design
Subsystem development (Increment 1)
Systems Integration
System Verification
System Validation
Transition to Operations
Operations, Maintenance and Support
Retirement (Disposal)
Subsystem development (Increment 2)
Systems Integration
System Verification
System Validation
Transition to Operations
Operations, Maintenance and Support
Retirement (Disposal)
Subsystem development (Increment 3)
Systems Integration
System Verification
System Validation
Transition to Operations
Operations, Maintenance and Support
Retirement (Disposal)
Architecture is defined ONCE and reused
Each increment adds more functionality All the requirements are known and stabilised at the beginning

Figure 3 – Iterative development

2.4 Feature Driven Development (FDD)

Feature Driven Development is one of the forms of the iterative development process. Functionalities of the software are defined as features. Each feature provides the business value and requires a small amount of work – rather weeks than months - to be implemented. Development team leader or a software lead selects the group of features that are to be developed during each iteration. Iterations are usually planned for 2 weeks and if the feature would take longer than 2 weeks it should be broken down on smaller parts to fit into the 2 weeks rhythm. It is very similar to Scrum. Main differences are initial step is designing an overall project model which is the starting point for the feature list creation [6], engagement of the end user in the process during progress reporting and more focus on documentation creation however it is still far from the aerospace standards. FDD based projects are unpopular for being out of schedule [7].

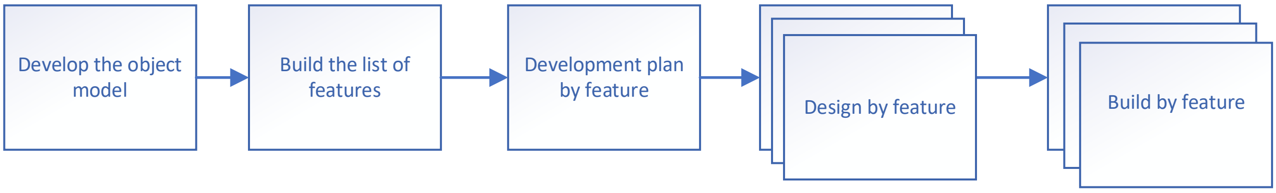

Figure 4 – Iterative development

3. Proposed method

3.1 Lifecycle

The development lifecycle in the proposed method identifies the phases of the software creation process and defines the entry and exit criteria for the transition between each of the lifecycle phases. The main activities for the software lifecycle are

- Initial Analysis & Planning

The phase of analysis and planning is a crucial initial activity, consisting of a generation of development plans and schedules, preliminary analysis of customer requirements including preliminary requirements review, to check their feasibility and generation of the associated requirements-based tests. During the planning phase requirements are decomposed in the way to associate a set of requirements to only one new/modified feature for the project. For each feature a sub phase of the project is created in the schedule. Project contains as many sub-phases in total as the number of all identified features.

- Development Iterations

The development iterations are an essential part of the lifecycle where the biggest benefit from applying Agile approach is observed. Every iteration is focused to deliver one feature/functionality of the product including corresponding increment in the project documentation. In each increment a partial SW lifecycle is applied to the features under development. Iterative Technical Readiness Reviews (TRR) and Software Validation Design Review (SVDR) reviews are held in order to allow assessment of the test readiness and test status at different points during the project.

- Release

In the product release phase after successful completion of the subsequent iterations the remainder of the lifecycle is a series of activities focused to deliver certification artifacts.

Over the period of the development, a number of controlled “releases” (engineering (X), pre-production (Y) and production (Z)) is made.

During this phase the post-iterative ‘for credit’ verification is performed. As demonstrated in Figure 5 before releasing a Software Z release a formal Validation & Verification (V&V) is performed. This phase consisted of a Full Software & System Test of the final release run on the hardware target environment and includes the creation of the remainder of project documents. The phase concludes when the candidate Z release of the software deliverable is generated. All testing of the features implemented and pre-validated in the previous phase is completed. This is delivered to the customer.

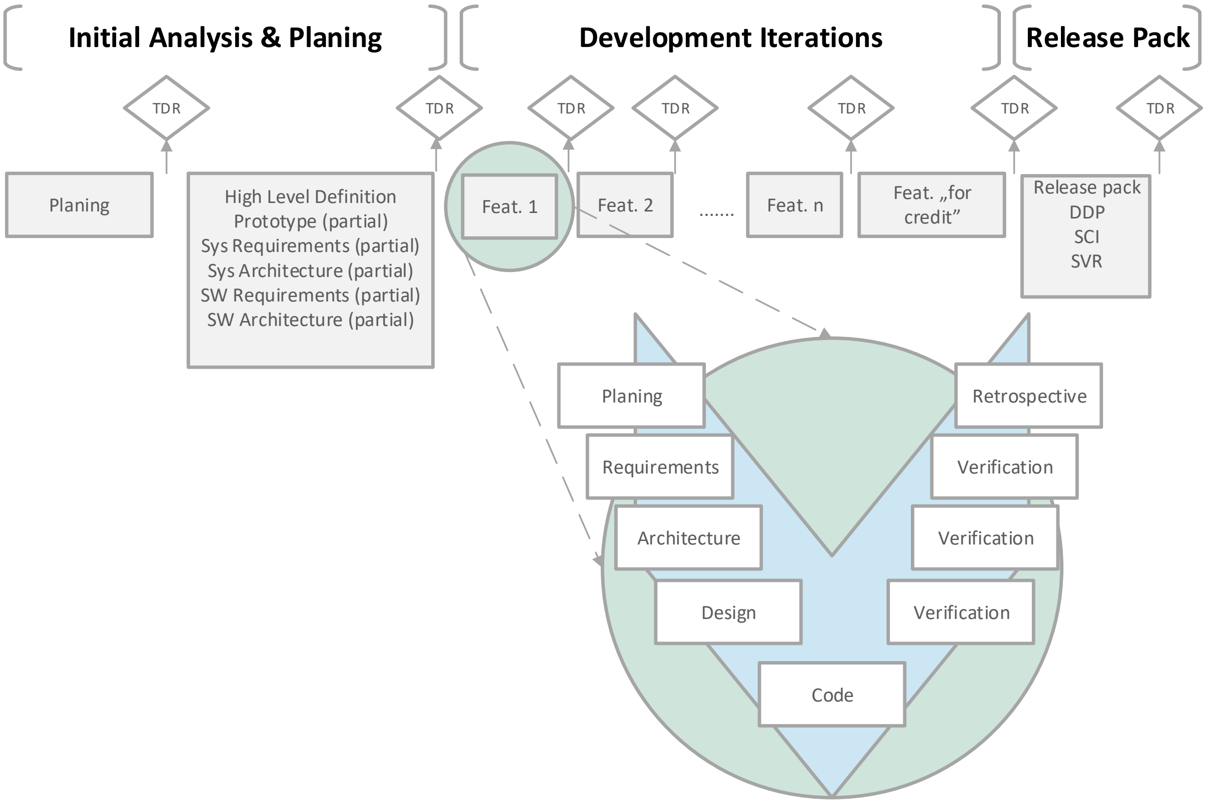

Figure 5 – Software development lifecycle.

4. Conclusions

This method appears to be very successful for the mid-size up to 15 engineers' teams. The main difference in approach in comparison to SCRUM and FDD was elimination of the time bound sprints. It was replaced by a feature focused approach without defined time limiting every iteration. The iteration is finished when feature is developed. It creates stable increment in each iteration. Iterative and incremental element of the method assures that all the artifacts including documentation are provided for each feature at the end of every sprint. It reduces amount of time required to create full "for credit" documents during the release phase. In comparison to typical waterfall approach of the project management methods provides better control over the progress of the project. Delivering small chunks of software allows the team to test more often and potentially fix bugs easier for every feature.

References

[1] Zakrzewski, Brenchley, Narkiewicz SAFETY CRITICAL SOFTWARE DEVELOPMENT METHODOLOGIES IN AVIONICS, Transactions on Aerospace Research, Warsaw Institute of Aviation 2020 (2)

[2] RTCS Software Considerations in Airborne Systems and Equipment Certification DO-178C

[3] Ken Schwaber, Jeff Sutherland, Scrum Guide, scrum.org, 2020

[4] Walden, D., Roedler, G., Forsberg, K., Douglas, H., & Shortell, T. (2015). INCOSE Systems Engineering Handbook: A Guide For System Life Cycle Processes and Activities (Fourth ed.). San Diego, CA: WILEY.

[5] SAE. (2010). Guidelines for Development of Civil Aircraft and Systems. SAE International.

[6] Gary Sheppard, Systems Engineering Management. The University of Warwick (2017)

[7] S. S. Tirumala, S. Ali, A. Babu G. A Hybrid Agile model using SCRUM and Feature Driven Development International Journal of Computer Applications · December 2016