IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Safety Critical Systems and Agile Methodology for Avionics Software Development

Chiranjeevi Aradhya

Sr Software Engineer

Collins Aerospace, United States of America

Abstract

Complex and critical elements of any modern aircraft are aircraft communication, navigation and flight control systems, and many other functions. Current avionic systems are more and more computer oriented, and software can be attributed to a growing percentage of device complexity. An airplane protection software error may lead to tragic incidents, such as many fatalities and aircraft losses. Certification agencies recognise the submission of compliance report for RTCA document DO-178B/C for software creation to show compliance with airworthiness specifications. The production of Avionics software is generally complex and historically depends on a rigorous planning process, with early setting of comprehensive requirement and late manufacturing of working software. This can result in a lot of reproaching and risk of budget overruns and schedule adjustments and software bugs being fixed. This raises the issue of the use of agile techniques in the production of avionics software. Based on the findings of two activities: an industrial literature review with the implementation of agile methodology in a DO-178B/C context, and a professional evaluation of the DO-178B/C objectives, an agile development framework is presented where Scrum is enhanced to achieve the DO-178B/C goals. Applying agile practices can help produce working applications on a daily basis and be able to adapt to changes, thereby lowering the risk of budget and schedule overruns.

Keywords: Avionics, Certification, Safety critical software, DO-178B/C, Software Life-Cycle, Agile, Scrum

Introduction

Modern aircraft are playing a key role in avionic systems. These systems provide operational assistance to pilots in all phases of the flight and under all weather conditions, including communications, navigation and aircraft control. When failure of a system can lead to loss of life, substantial damage to property or environmental damage, it is vital to safety [1]. The flight control system, which controls a plane's attitude and, ultimately, the flight path that follows, is an example of a safety-critical avionic system. Safety-critical systems not confined to the avionics domain, such as process control [2], medical equipment [3] and the automotive sector [4] are other major fields of expertise. Present avionic systems are more machine oriented and more functions are added as applications are introduced. The European Aviation Safety Agency (EASA) and Federal Aviation Administration ("FAA") accredited agencies agree to apply RTCA document DO-178B/C [5] as a means of improving avionics software, in order to ensure compliance. Document DO-178B/C calls for many expensive and time-consuming safety goals to be accomplished in general [6]. Traditionally, the avionics industry uses the V model or a version of it as a software development life-cycle model. This corresponds perfectly to DO-178B/C when looking at the data items for the life cycle to be produced. However there are inconveniences too. For instance, until late in the life cycle, no working software is created.

Errors found at this stage can result in a great deal of reworking of previous activities and increase the risk of overrunning the budget and schedule [7]. Likewise, changes in late-stage specifications could also result in a lot of rework with like results. The use of agile techniques could solve these problems. The problem, however is that the uncertainty of an agile process does not seem to fit the rigour imposed by DO-178B/C. For example, agile development considers it more important than following a plan to adapt to change, whereas DO-178B/C is strictly plan driven. The key issue of this research is how agile processes can be modified to be used in a DO-178B/C avionics production process.

Brief about DO-178B/CC

The certification software approvals of avionic device software by certification agencies, such as EASA/FAA, are regulated in document DO-178B/C. DO-178B/C distinguishes between five (A-E) software levels based on a state of failure that could result from software misconduct. Software is rated to the (highest level A if unintended software behaviour, typically with loss of aircraft, can cause a catastrophic failure of the aircraft. The effect of incorrect software activity gradually does not impact protection at the lower software level (level E).

DO-178B/C is a process-oriented specification based on proof of the efficient results of the different software development activities. The software planning process, identifying and organising the activities of all software development processes (2) that generate a software product and (3) that ensure accuracy of the

software product and confidence in the software processes and in their performance, is categorised in three separate categories: DO-178B/C does not deal with system life-cycle processes, but it specifies the interaction with system processes, including an examination of system protection.

The guidance DO-178B/C shall be given by (1) defining the goals for the software life cycle process, (2) describing tasks which are a means of achieving the goals and (3) describing evidence as data items in order to show that the goals have been met. A specific life cycle software or technique is not recommended by DO-178B/C. A project for software development determines its life cycle by defining a series of processes and their sequence. The normal sequence of specifications, design, coding and integration through the software development processes.

Security standards and authorities relationship

For all organisations, DO-178B/C is certainly the most important norm. Level C is the most common in applications on all levels of DO-178B/C (60 percent of the respondents). As a result, the coverage of data items is very high. If asked about how well the external evaluator communicates, 50% report interacting with the evaluator at all project levels. The remaining records are lower. The average estimate (including all examinations and testing) for the cost of testing is 40% of the overall budget of the project.

Challenges and Problems

Verification and certification challenges include: (1) sufficient resources, infrastructure and skill/personnel, (2) high quality customer contact, including specifications and feedback input and (3) proof of compliance with the certification authority requirements of DO-178B/C. Requirements management (frequent adjustment, insufficient requirements, vague requirements and additional new requirements) and late detection of the problems/defaults as well as project costs are the highest issues for this software development process.

DO-178B/C-Aligned Agile Approach

As previously stated, DO-178B/C does not recommend a specific software life cycle model. Software life cycles such as the waterfall, the V model, the incremental and the spiral can be described, but agile approaches can also be implemented. The Scrum Frame is considered an appropriate agile framework, and can be used as a baseline. In general it has a large range of educational tools, industrial expertise and accessible research literature, the most widely used agile framework within the software industry. To develop an avionics programme, Scrum will have to be extended to allow the delivery, in accordance with DO-178B/C, of all necessary data products.

Various stages of Scrum

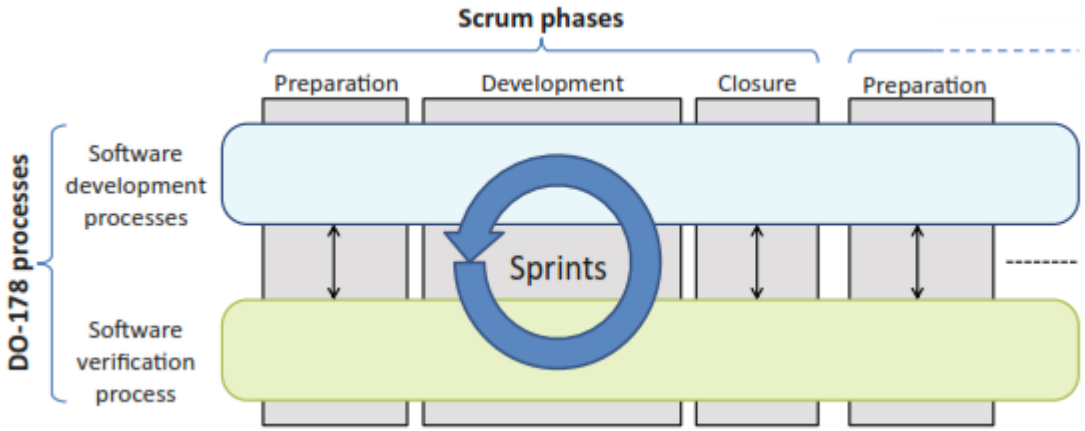

Figure 1: DO-178B/C with Scrum

I use the terms Planning, Development, Closure in this paper, which are also often used for example, [8]. The Scrum phases are added to the DO-178B/C software creation and checking processes as shown in Figure 1, enables the processing of agile approaches. The planning and architecture tasks are carried out during the preparation phase. The strategy concept in Scrum is a little wider than the DO-178B/C concept. Scrum covers the concept of the next release of software on the basis of the known backlog, system requirements review and user history creation. The tasks of architecture determine the framework of the programme. The functionality of a new release and checking for new or modified code are created during the development process. The programme is designed to incorporate, integrate and validate the source code during a Sprints series. The software release including system testing, final documentation and release is planned during the Closure process. Until the finished release of software, the process of Planning, Development and closure is repeated.

Table 1. Activities during Preparation phase.

DO-178B/C	Inputs	Activities	Outputs
Software requirements	Allocated system requirements, software level	Define system features and prepares user stories.	HLRs, trace data
Software design	Software high level requirements (HLRs)	Establish or refine software architecture, including partitioning concept	Software architecture, trace data
Software verification	HLRs, software architecture, trace data	Define test cases for HLRs. Verify all outputs	HLR test cases, verification results

Table 2. Activities during Development phase

DO-178B/C	Inputs	Activities	Outputs
Software design	HLRs, software architecture, trace data	Define software Low-level requirements (LLRs) by conditions and associated actions	LLRs, trace data
Software coding	LLRs	Produce code for the	Source code
Integration	Source code	Perform continuous	Executable
Software verification	HLRs, HLR test cases, software architecture, LLRs, source code, executable object code, trace data	Establish test cases for LLRs. Produce test code for HLRs and LLRs. Execute (automated) tests. Verify all outputs	HLR test procedures, HLR test results, LLR test cases, LLR test procedures, LLR test results,

Table1 and Table2 describe the activities of preparation and development phase. A sufficient number of the options should be completed to warrant a release on commencement of the closure phase. All data items that exist (see outputs in Tables 1 and 2) will be changed during the closing phase. Other processes than software creation and software testing are used to generate the remaining data items needed to comply with DO-178B/C. The software configuration process for example creates the Software Control Index and the Software Completion Summary is generated by the certification liaison process.

Remarks and potential issues

The proposed process seeks to meet certain main challenges, especially requirement management, which we defined in the survey. Breaking in shorter iterations including preparation and assessment ensures that planning can be carried out with revised Sprints and each Sprint offers the requisite details to satisfy DO-178B/C requirements. From research in relation to this, we know that tools are needed to automate the creation of test-driven products and the generation of documentation to ensure time savings and quality and consistency [9].

The use of agile approaches in the avionics software development process promises the widely listed benefits of providing working software, including all of the data items needed by the DO-178B/C and the ability to deal with regular requirement modifications. There are a variety of possible problems, however. Unlike a waterfall or a V-model, HLRs are specified in batches. A sufficient number of HLRs is defined for the following Sprint sequences any time a preparation stage is entered. Without an overview of the entire HLR collection at the very beginning, an unsatisfactory software design might be required during subsequent planning. This could entail drastic (and thus expensive) evaluations. This means that agile projects must also invest in appropriate HLR and system design in sufficient detail early on. An agile method will provide enhanced chances of handling changes if occurring. Another challenge is the potential implications for the safety analysis by identifying derived HLRs late in the development, e.g. after many planning, development, and closure cycles [10]. For example, if derived HLRs include new interfaces which counterfeit earlier claims for independence, a higher levels of software may be appropriate, which could have done more (verification) work in a better way when previously known.

Conclusion

RTCA document DO-178B/C regulates the production of security software essential to the avionics industry. The document emphasizes recorded and traceable testing so that an appropriate level of trust is reached with the performance of software development activities. Our survey, which includes leading players in the European avionics industry, indicated that certification forms a substantial portion of overall production costs (estimated 40 percent). In the survey, other challenges found by this industry include volatility criteria, identification of problems/defects late, and overruns of project costs.

Approving an agile system may be a solution to these problems, in line with other research relevant to the safety industry. At present, the V-model or variations of this life-cycle model, which the avionics industry often uses to coordinate software creation. However, DO-178B/C does not prohibit the use of any specific model and in general; the implementation of an agile system appears to be without obstacles. It is apparent that agile processes, such as Scrum, must be adapted to build and certify avionics software. These approaches

must in particular be generalised to satisfy traceability and documentation criteria. Some of these can be allowed by the use of proper automation software.

On the basis of Scrum, a strategy that benefits from agile approaches and can also fulfil the goals of DO-178B/C was outlined. Some aims of DO-178B/C shall be accomplished by an agile manner; others in particular partial to the verification targets, shall, on the conventional basis, be performed, respectively (management plans, reviews, and analyses). The anticipated advantages of an agile approach include risk reduction, adaptability to evolving requirements and a reduction in development cost overall.

However there are problems which require more study. One is that software requirements are set out in batches; for the following sprint series, adequate software requirements are specified each time. Without an overview of the entire set of software specifications in the early stage of development, software architecture may be insufficiently updated, which would require more careful analysis. To conclude, agile methods can promise to solve some specific avionics problems, but further research and industry testing is still required to validate their applicability and show improvements.

References

[1] Knight, J.C.: Safety critical systems: challenges and directions. In: Proceedings of the 24rd International Conference on Software Engineering, ICSE 2002. IEEE (2002)

[2] Stalhane, T., Myklebust, T., Hanssen, G.K.: The application of Scrum IEC 61508 certifiable software. In Proceedings of ESREL, Helsinki, Finland

[3] Rottier, P.A., Rodrigues, V: Agile development in a medical device company. In: AGILE 2008 Conference (2008)

[4] Hantke, D.: An approach for combining spice and scrum in software development projects. In: Rout, T., O'Connor, R.V., Dorling, A. (eds.) SPICE 2015. CCIS, vol. 526, pp. 233–238. Springer, Cham (2015)

[5] RTCA, DO-178C: Software considerations in airborne systems and equipment certification (2011)

[6] Hilderman, V.: DO-178B Costs Versus Benefits. HighRely Inc., HighRely Whitepaper (2009)

[7] Chenu, E.: Agile and Lean software development for avionic software. Whitepaper, Thales Avionics (2011)

[8] Meunier, V., Destouesse, M., Cros, T.: How to “take credit” of agile principles within a certification context (2008)

[9] Hanssen, Geir K., Haugset, B., Stålhane, T., Myklebust, T., Kulbrandstad, I.: Quality assurance in scrum applied to safety critical software. In: Sharp, H., Hall, T. (eds.) XP 2016. LNBIP, vol. 251, pp. 92–103. Springer, Cham (2016).

[10] VanderLeest, S.H., Buter, A.: Escape the waterfall: agile for aerospace. In: Proceedings of IEEE/AIAA 28th Digital Avionics Systems Conference, DASC 2009, p. 6, (6D3). IEEE (2009).