International Journal of Aeronautical and Space Sciences
https://doi.org/10.1007/s42405-023-00663-8

ORIGINAL PAPER

Software Safety Assurance in Non-airborne GPS-based Landing Aids

Daniel Jendraszyk[1] · Thomas Dautermann[2]

Received: 7 October 2022 / Revised: 11 July 2023 / Accepted: 24 August 2023
© The Author(s) 2023

Abstract
This paper investigates the software standard DO-278A from the RTCA according to assurance level 3 to provide a concept for approving software of ground-based navigation aids. For this purpose, related literature and standards were reviewed and evaluated for their applicability for proposing such a concept. The resulting approval concept shows our approach for conducting the development process according to DO-278A based on a traditional one and focusing on an activity schedule. Therefore, the paper offers preliminary considerations at first to show the relation between system and software development. In addition, corresponding standards that are related to the development are presented as well. The concept is segmented in three main phases for software planning, realization, and verification, which are subdivided and scheduled in specific activities. Each activity consists of an outline describing the contents expected by DO-278A and our approach to organizing them. This paper shows our conceptional approach to obtain an approval for software according to DO-278A. This concept is prepared to approve a radio navigation aid.

Keywords GNSS · GBAS · Navigation aid · Aviation · Certification · Software safety

1 Introduction

Since 1939, radio transmitters of instrument landing system have provided precision final approach guidance to aircraft intending to land in less than optimal weather conditions [1]. This more than 80 year old technology consists of several very-high-frequency antennas located at two installations for each runway as well as the associated hardware. The instrument landing system is to be succeeded by landing aids based on Global Navigation Satellite Systems (GNSS) for improving accuracy integrity as well as maintenance cost and resilience. These systems are augmented by differential corrections from the Satellite-Based Augmentation System (SBAS) [2] or the Ground-Based Augmentation System (GBAS) [3].

✉ Thomas Dautermann
thomas.dautermann@dlr.de

Daniel Jendraszyk
daniel.jendraszyk@unibw.de

1 Faculty of Electrical Engineering and Information Technology, Institute for Microelectronics and Integrated Circuits, University of the Bundeswehr Munich, Neubiberg, Germany

2 Deutsches Zentrum für Luft- und Raumfahrt, Brunswick, Lower Saxony, Germany

These modern systems rely heavily on computer technology to generate, transmit, and apply augmentation data to provide highly precise and safe guidance. Therefore, it is important that the processing software and hardware fulfill a preset level of reliability and integrity to ensure the continuous safety during aircraft approach and landing.

In the 1980s, the utilization of software in airborne systems has grown rapidly. Therefore, the need arose to make a document supplying guidance on airworthy software. As a result, the standard DO-178 "Software Considerations in Airborne Systems and Equipment Certification" was created. It deals with issues that are important for producing software for safety–critical aviation-related systems, e.g., certification-related aspects [4, p. 1], [5].

Around 1996, a cooperation of European Organization for Civil Aviation Equipment (EUROCAE) working groups and Radio Technical Commission for Aeronautics (RTCA) special committees were assigned to ascertain the significance of software safety in ground-based systems for Communication, Navigation and Surveillance/Air Traffic Management (CNS/ATM). Then, they established a team to develop material for these systems. That team reviewed the applicability of DO-178B (Software Considerations in Airborne Systems and Equipment Certification) to those systems and developed DO-278 (Software Integrity Assurance Considerations

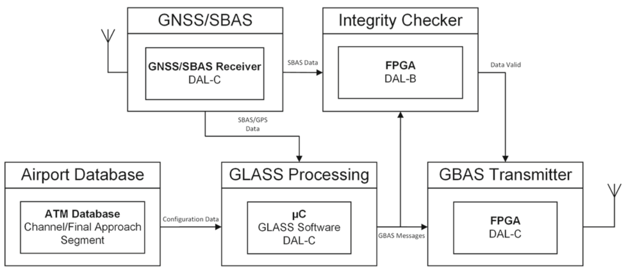

Fig. 1 Proposed system concept for a GLASS transceiver

for CNS/ATM Systems) [6]. The development was based on principles like the following: First, the original guidance for obtaining certification of airborne equipment should not be changed to develop the new standard. Second, the guidance for software integrity assurance of airborne and non-airborne systems should be the same. Third, approval and certification mechanisms should not be changed, but regard should be given to gaps in regulatory and certification requirements [6, p. A-2, A-3], [7].

Hence, this offers the possibility to consider DO-178C [4] related contents in DO-278A contexts as well. Since there are more users of DO-178C than of DO-278A and they are very similar [8, p. 55], the consideration of DO-178C-related contents is useful.

We proposed a simplified GNSS-based landing system providing GLS' approaches using SbaS (GLASS) in [9]. It rebroadcasts the correction and integrity data from SBAS via a GBAS compliant data link. This enables an aircraft equipped with a Global Positioning System Landing System (GLS) to perform Localizer Performance with Vertical guidance (LPV) approaches. In general, commercial transport is not equipped with the capability to fly SBAS LPV approaches. The manufacturers do not provide this option due to their primary focus on precision landing systems. Equipage among the commercial aircraft fleet is 12% for GLS [10]. LPV capability is much less present, as only the Airbus A350 and A220 are offered with this option. Thus, many such GLS airline aircraft can benefit from already published SBAS-based approach procedures.

The system concept for such a transceiver is shown in Fig. 1. It mainly consists of Commercial-Off-The-Shelf (COTS) components like a GNSS/SBAS receiver, database, and GBAS transmitter. However, the GLASS processing and integrity verification unit are custom components. The GNSS/SBAS receiver provides position and integrity data to the GLASS processing and integrity verification unit. Once the GLASS processing unit has converted that data, the integrity verification unit compares these data with its source. If the converted data are approved, the GBAS transmitter can send it. In addition, the Development Assurance Level (DAL) denotes the rigor required for developing the system component [11, p. 39]. DAL A is the most rigorous level and DAL E is the least rigorous one [11, p. 88–96].

Innovative systems fill a gap in the already existing system standardization. Therefore, it can be necessary to fall back on general-purpose standards like Annex Ten to the Convention of Chicago [12], that comprehensively defines the operating conditions of radio navigation aids. Moreover, these systems are unknown to certification or approval authorities that must approve them before putting them into operation. However, for software developed in an aviation context, the RTCA has published the general guideline DO-278A and regarding software in non-airborne systems that impacts aircraft safety. It is considered by approval authorities as an acceptable means of compliance for software safety. Here, taken the GLASS system as an example, we develop and assess the software approval concept for a ground-based navigation aid in preparation for certification.

2 Preliminary Considerations and Corresponding Standards

Software development is not a self-contained activity as it depends on constraints like the proposed processor, the concomitant specific compiler tool chain, and interfaces to other components. Moreover, further standards, such as for developing hardware [13], qualifying software tools [14], and conducting safety assessments [15], must be considered as well depending on the specific project.

These relations are outlined in Fig. 2. The artifact on the left side represents national and international regulations and standards for aviation that must be fulfilled. For this purpose, there are standards for system development, which are responsible to fulfill regulations and conducting the Safety Assessment Process (SAP), among others. Since the system development allocates functionality to hardware and software, there are respective standards that are represented by the means of the center standards. On the right side are artifacts connected to the software standard representing further documents that are directly related to it.

Since DO-278A is an aviation-related standard, a corresponding SAP must be conducted to get system-related software requirements as the assurance level (AL) for software development. The AL assigns the level of rigor for which a software component needs to be developed [6, p. 140]. DO-278A offers six ALs, where AL1 denotes components whose malfunction results in catastrophic failure conditions, whereas AL6 denotes components that do not affect aircraft safety [6, p. 14].

As shown in Table 1, the assurance level has a great influence on the effort to obtain an approval.

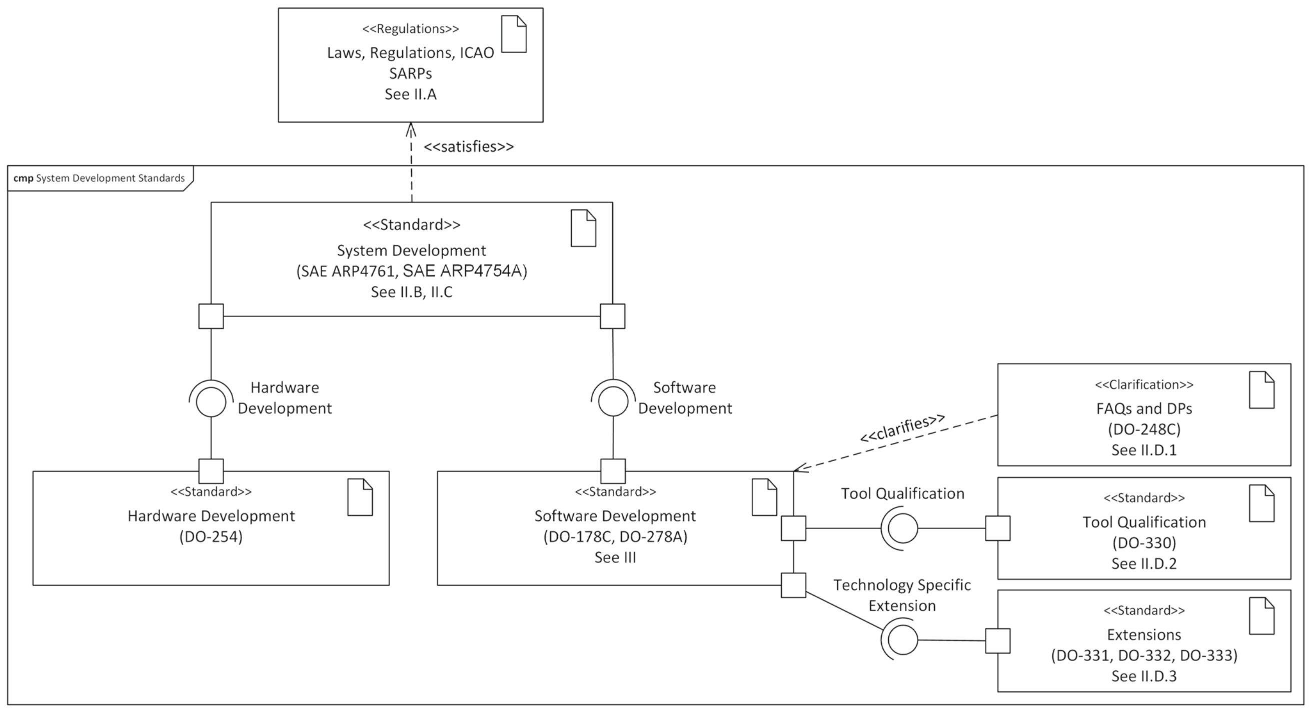

Fig. 2 Component diagram of corresponding standards, based on [8, p. 54]

Table 1 Comparison of approval aspects of assurance levels according to DO-278A. Source: [31, p. 33], extended and adapted to DO-278A

DO-278A aspect	AL1	AL2	AL3	AL4	AL5
Classification of failure condition	Catastrophic	Hazardous	Major	Not associated	Minor
Number of objectives	71	69	62	41	26
Independence	High	Medium	Low	Low	Very low
Low-level requirements	Yes	Yes	Yes	No	No
Statement coverage	Yes	Yes	Yes	No	No
Decision/condition coverage	Yes	Yes	No	No	No
Modified condition/decision coverage	Yes	No	No	No	No
Configuration management	Tight	Tight	Medium	Medium	Low
Source to binary correlation	Yes	No	No	No	No
Requirements correlate to target processor	Yes	Yes	No	No	No
Architecture and algorithm verification	Yes	Yes	Yes	Yes	No
Code reviews	Yes	Yes	Yes	No	No
Software quality assurance transition criteria	Yes	Yes	Yes	Yes	No

2.1 Legal Considerations

Depending on the country in which the radio navigation aid should be taken into operation, legal regulations must be considered. As an example, the German law regulates in § 4 of the German Air Traffic Control Equipment and Device Type Approval Regulation (FSMusterzulV, see [16]) that the official journal News for Aviators issues device requirements for air navigation services. For instance, this official journal published the Notification concerning the Requirements for Type-Certification of GBAS Ground Facilities as Aeronautical Radio Navigation Stations, which refers to several other standards like Society of Automotive Engineers (SAE) Aerospace Recommended Practice (ARP) 4761 for conducting the SAP and ED-109 for software safety [17].

2.2 Safety Assessment Process

Before the software life cycle starts, the SAP needs to be started as it allocates an AL for each software component.

However, as mentioned in [6, p. 7], a complete system life cycle process description is not part of DO-278A, but can be found in other industry documents as the following. Typically, SAE ARP4754A "Guidelines for Development of Civil Aircraft and Systems" [11] and SAE ARP4761 "Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment" [15] can be used for system life cycle processes [18, p. 14], [8, p. 14, 33–34].

The SAP according to SAE ARP4761 is a process that includes generation and verification of requirements. It starts with a Functional Hazard Assessment (FHA) for identifying and classifying failure conditions of single or combined functions including rationale therefore. As a result, corresponding safety objectives will be established. Then, the Preliminary System Safety Assessment (PSSA) examines the proposed system architecture to determine how failures can cause functional hazards identified by the FHA. Moreover, it establishes safety requirements to meet safety objectives that are identified by the FHA. For that purpose, the PSSA usually consists of a Fault Tree Analysis, Dependence Diagram, or Markov Analysis, and should include a Common Cause Analysis. After system implementation, safety objectives and requirements established by FHA and PSSA will be evaluated by the System Safety Assessment (SSA) whether they are fulfilled. The SSA use the same methods as the PSSA. However, it focuses on the verification that the objectives allocated by FHA and PSSA are met [15, p. 12, 15].

2.3 Assurance-Level Determination Procedure

The assurance level can be basically determined analog to the procedure that is indicated in ED-114A, see [19, p. 24–25]. That procedure consists of the following three steps:

1. Determination of Risk Budgets
   In the context of radio navigation aids, the International Civil Aviation Organization (ICAO) GNSS Standards and Recommended Practices (SARPs, s. a. [12, p. 3-72, APP B-122, APP B-125]) define the allowed risk budgets for flight operations or ground equipment. They are further divided in integrity and continuity risks.
2. Relation to Failure Classifications
   Based on the determined risk, the European Union Aviation Safety Agency (EASA) provides in Certification Specification (CS) 25 the relation between quantitative risks and qualitative failure classifications in Acceptable Means of Compliance (AMC) 1309. See [20, p. 2-F-40 to 2-F-78].
3. Allocation of Assurance Levels
   The ALs will be allocated according to their quantitative failure classification in [6, p. 14]. See also the classification of failure conditions in Table 1.

Additionally, if the software can be partitioned in isolated components during the SAP, they can be assigned with different assurance levels. The independence can be achieved if each software component will be executed on separate hardware or the software provides provisions for ensuring independence of its components [6, p. 11, 16–17].

2.4 Standards Belonging to DO-278A

The "core documents" DO-178C and DO-278A are intended to be technology-independent. Accordingly, there are other documents built upon or related to them. At first, there is DO-248C for clarification containing Frequently Asked Questions (FAQs) and Discussion Papers (DPs). Second, if software tools are needed during development, DO-330 supplies guidance on their qualification. Finally, three technology-specific supplements are extending the core documents for model-based development (DO-331), for object-oriented technologies (DO-332), and for formal methods (DO-333) [8, p. 54–55].

2.4.1 Supporting Information (DO-248C)

The supporting information document contains FAQs, DPs, and rationale for both the industry and authorities. There is no further guidance, but clarification related to the core documents. The FAQ section supplies concise explanations in less than two pages, whereas longer ones are covered in the DP section. To support comprehending the core documents, background information is provided in a rationale section. DO-248C recommends being used by looking for keywords in appendix C or references of the corresponding core documents sections in appendix D [18, p. 1–2], [21].

2.4.2 Software Tool Qualification (DO-330)

Software tools like compilers or code generators are widely used in the context of software development. They can improve system safety. By way of contrast, they can adversely affect system safety if they are erroneous. For this reason, tool qualification should ensure their functional correctness. More precisely, tool qualification is required if processes of the core documents will be replaced by such a tool and its output will not be verified as mentioned by them [14, p. 1, 5], [22], [6, p. 88].

The following criteria [6, p. 89] are established to consider the potential effect on software safety including examples [8, p. 320,322]:

- Criterion 1
  Tools affecting the resulting software directly like code generators or modeling tools.
- Criterion 2

Table 2 Correlation of assurance level and software tool criteria with tool qualification level

Assurance level	Criteria 1	Criteria 2	Criteria 3
AL1	TQL-1	TQL-4	TQL-5
AL2	TQL-2	TQL-4	TQL-5
AL3	TQL-3	TQL-5	TQL-5
AL4	TQL-4	TQL-5	TQL-5
AL5	TQL-4	TQL-5	TQL-5

See [14, p. D-9]

Tools for verification automatization that can fail to detect an error, whose output justifies reducing verification or development processes. For instance, that applies for code analyzers replacing source code review or overflow detectors during software design.

- Criterion 3
  Tools that can fail to detect an error like test case generators and code analyzers.

These criteria lead together with the assurance level to the Tool Qualification Level (TQL), as shown in Table 2. The TQL results in several objectives that must be satisfied to obtain the approval for using the tool. DO-330 offers up to 76 objectives in total and the more rigorous the TQL, the more objectives need to be satisfied.

2.4.3 Supplements (DO-331, DO-332, DO-333)

As already mentioned, there are three supplements that extend the core documents. They add objectives for technology-specific guidance.

At first, DO-331 [23, 24] supplements the core documents with considerations for model-based development. This technique uses an abstract model that represents system aspects and performs development or verification activities on it [23, p. 82]. The reduction of development time, cost, and human errors as a result of using models and qualified code generators is the main reason for applying model-based development [8, p. 345].

Then, the core documents can be supplemented with DO-332 [25] to consider object-oriented technology and related techniques. Object-oriented technologies are based on the concept of developing and programming software based on objects [25, p. 3]. Owing to their advantages like strong tool support and possible cost savings as well as are already unitized in safety–critical applications, their utilization in aviation can be preferable [8, p. 360].

Finally, there is DO-333 [26] for supplementing the core documents for the utilization of formal methods. The application of formal methods comprises two activities, namely formal modeling and formal analysis. A formal model is a mathematical representation of system aspects. Additionally, a formal analysis describes the utilization of mathematical methods to ensure that a defined behavior is achieved. Altogether, formal methods can be outlined as mathematical reasoning of correctness and behavior of modeled system aspects. For this reason, the software development and verification process can benefit from utilizing formal methods [8, pp. 371–372], [26, p. 1, 58].

3 Approval Concept for GLASS

The approval concept for GLASS is based on the traditional "five-plans-and-three-standards" approach provided in [8]. This means that project-specific development plans and standards will be implemented as offered by DO-278A. In addition, there is a freely accessible template set (see [27]), which is suitable for this approach. It outlines many documents of an example project by containing tables of contents. Besides this traditional approach, it is also possible to use modern agile software development processes if the DO-278A objectives are addressed adequately, but this can lead to challenges with the approval authority [8, p. 74].

As shown in Fig. 3, the approval concept is structured in phases for previous system processes, software planning, software realization, and software verification. There are previous processes on system level prior to software development, such as prototyping and selecting hardware. When finished, hardware characteristics like interrupts or interfaces can be regarded and the plans can be written. After finishing the software plans, the Plan of Software Aspects of Approval (PSAA) can be created as well and submitted to the approval authority. If it is authorized, the software realization processes can be started and their output be reviewed. Finally, the software will be tested and analyzed during the software verification phase and its results will be verified again.

The first two stages in the previous system processes' phase are grayed, because they are not part of the essential approval concept. However, they are important preprocesses to the software development process as they provide system description and approval aspects that are relevant for software development. Afterward, the software planning phase is aligned to complete all plans that can be reused for writing the PSAA at first as it is the first document that will be submitted to the approval authority. After submitting the PSAA to the approval authority, additional plans can be created that are not essential to the approval process. For instance, that could be an internal test equipment scheduling plan. When the PSAA is approved, the development activities of requirements, design, and code can be performed in the realization phase based on the plans. This includes already respective reviews. Eventually, the software can be tested during the

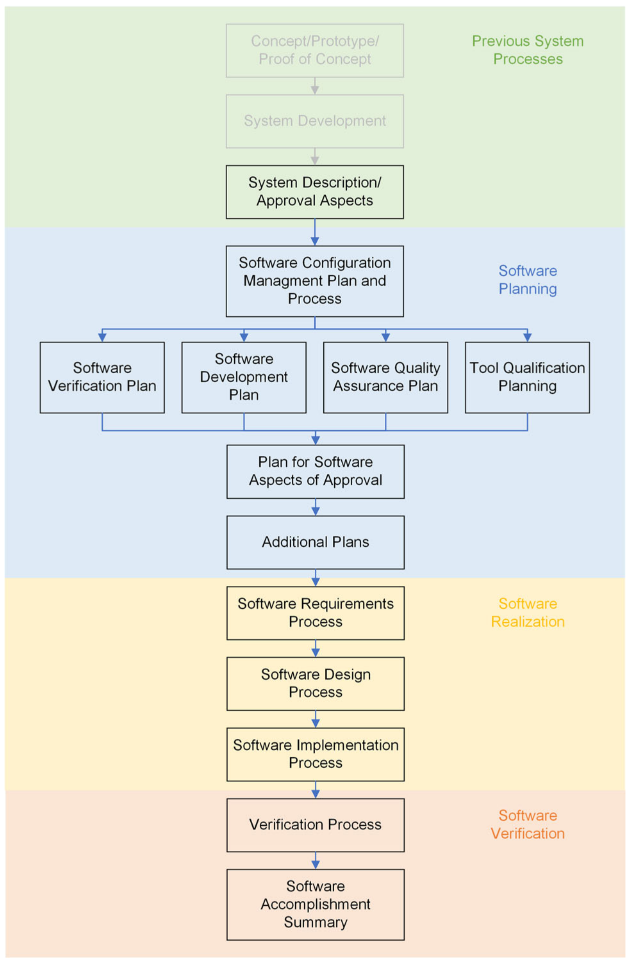
Concept/Prototype/ Proof of Concept
Previous System Processes
System Development
System Description/ Approval Aspects
Software Configuration Managment Plan and Process
Software Planning
Software Verification Plan
Software Development Plan
Software Quality Assurance Plan
Tool Qualification Planning
Plan for Software Aspects of Approval
Additional Plans
Software Requirements Process
Software Realization
Software Design Process
Software Implementation Process
Verification Process
Software Verification
Software Accomplishment Summary

Fig. 3 Approval concept

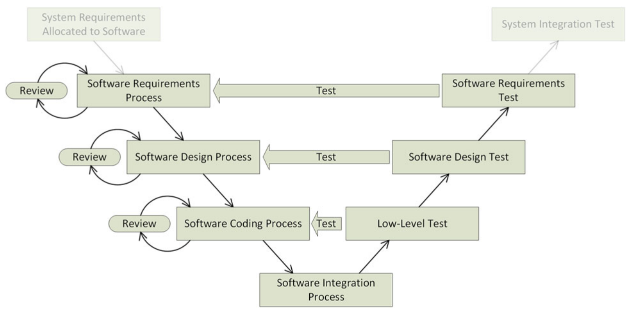

Fig. 4 V life cycle model

verification process. This distribution of the verification process leads to the V life cycle model, as shown in Fig. 4. Finally, the Software Accomplishment Summary can be written and submitted to the approval authority. If it and further requested data are approved by the authority, the software can be taken into operation.

Several data items will be generated during the approval procedure, as shown in Table 3. They are required by DO-278A and their extent depends on the assurance level. Because it is not required to create an independent document

Table 3 Data items to be created

Plan for software aspects of approval	Software development plan
Software verification plan	Software configuration management plan
Software quality assurance plan	Software requirements standards
Software design standards	Software code standards
Software requirements data	Design description
Source code	Executable object code
Software verification cases and procedures	Software verification results
Software life cycle environment configuration index	Software configuration index
Problem reports	Software configuration management records
Software quality assurance records	Software accomplishment summary
Trace data	Adaptation data item file

See also [6, p. 69–83]

for each data item, some data items will be put into one document. The contents and combination of them will be outlined in the following. A detailed description can be found in the 11th section and to be satisfied objectives according to the assurance level are shown in annex A of DO-278A.

3.1 System Description/Approval Aspects

Development processes on system level will be already performed. Therefore, relevant information like high-level system descriptions or proposed system functions allocated to software will be already available. They will be collected and reused for creating the PSAA. This can include diagrams like use case diagrams and state machines. Since the system concept is not final at present, a PSSA or complete system description is not available. For this reason, literature was reviewed and datasheets of related products were investigated. First, the Minimum Operational Performance Standard (MOPS) for GBAS ground stations [28, p. 24, 25], assigns AL2 and AL5 to prevent integrity and continuity violations, respectively. Second, MOPS for SBAS airborne equipment requires at least AL2 for integrity if no AL was assigned during the SAP. Then, according to the AL determination procedure presented in Sect. 2.3, the software must be developed at least under AL3 for integrity and continuity safety. Finally, the majority of investigated datasheets show that their software is developed under AL2 and a minority under AL1 or AL3. In conclusion, software responsible for integrity and continuity must be developed at least under AL3, but there is a strong indication that the integrity aspect shall satisfy AL2. In consequence, an integrity checker is introduced for using the less rigorous AL3. Furthermore, the separate integrity checker introduces redundancy. Eventually, the software approval concept is based on the proposed system concept, as shown in Fig. 1.

3.2 Software Planning Phase

As already mentioned, the software planning phase is aligned to complete all plans that can be reused in the PSAA. Initially, the Software Configuration Management (SCM) Plan should be written and the corresponding SCM process should be started. This ensures that all plans are under configuration management from the beginning. Afterward, the Software Verification Plan (SVP), Software Development Plan (SDP), and the Software Quality Assurance (SQA) Plan can be written including plans for tool qualification, if the use of software tools is planned. These four plans can be written in parallel as there are no scheduling dependencies between them. After that, the PSAA can be written based on the collected data from system development and the already written software plans. Finally, plans that do not affect the PSAA can be created.

3.2.1 Software Configuration Management Plan and Process

The SCM process starts with the planning phase, continues during the software development, and ends with the phase-out of the software [8, p. 86]. For this purpose, the SCM Plan provides methods to be used within the configuration management to satisfy its objectives [6, p. 74]. This includes procedures for configuration identification, baselines and traceability, problem reporting, change control and review, configuration status accounting, software load control, software lifecycle environment, and data control as described in [6, p. 56–61, 74–75] and [8, p. 86–88].

The approval concept starts with enabling basic activities. Therefore, the SCM process will be established at first as it provides them. For this purpose, the Software Life Cycle Environment Configuration Index (SECI) and the Software Configuration Index (SCI) will also be created. Both SECI and SCI [7, pp. 79–80] are for identifying software, where SECI lists hardware and software required for development and SCI lists everything related to the resulting software product. Because basic software, e.g., for word processing, is needed as well, it is useful to define them as early as possible. Further output of this process is SCM Records and Problem Reports.

3.2.2 Software Development Plan

The intent of the SDP is to provide project-specific guidance for developers who will write requirements, design, and code and can be included in the PSAA. At first, the SDP includes project-specific standards for ascertaining software requirements, establishing software design, and writing code. More precisely, that are Software Requirements Standards, Software Design Standards, and Software Code Standards. They will usually be referenced in the SDP and allocated in separate documents. Since they must be project specific, e.g., for considering the assurance level, it is not possible to use an arbitrary industry standard. However, the project-specific standards can be based on them. Second, the designated software life cycle must be explained in the SDP to provide a clear project management structure for developers. The only limitation for selecting a software life cycle model is that the verification must be performed top–down. In other words, the requirements must be verified prior to design verification and the design needs to be verified prior to implementation verification. Moreover, the produced data and criteria for entering and exiting must be defined for each development phase. Finally, a description of the software development environment should also be included. This ensures the reproducibility of the Executable Object Code (EOC) and reduces the risk introduced by the development environment. This includes all tools that are used within the project like hardware platforms, compiler, loader, and programs for requirements engineering, software design, and code development. That description focuses on guidance for using the development environment during the project. The SECI can also be referenced to reduce redundancy. In that case, it should be released along with the plans. However, then the SECI might need to be updated during software development [6, p. 30, 73], [8, p. 81–83, 90–91].

Due to the aforementioned points, the SDP will be included in the PSAA and references the project-specific standards for requirements, design, and code. This leads to a concise SDP chapter which is focused on the software life cycle organization including its environment. Conversely, the standards provide specific guidance to particular life cycle phases. To sharpen the focus further, the SDP describes the development environment in general and references the SECI for details. This avoids inconsistencies between these documents as the PSAA requires otherwise a summary of the SDP. Since not all details of the software development environment will be determined during the planning phase, the SECI will be updated during the development. For instance, this can concern libraries or compiler options and versions.

Because a traditional approach for software approval was chosen, a V-model software life cycle [29, p. 33–34] will be applied. The suggested software life cycle model is shown in Fig. 4 and adjusted to the needs of DO-278A as well as to the approval concept.

Based on [29, p. 33–34], adapted to DO-278A and approval concept

3.2.3 Software Verification Plan

The SVP describes how the software verification will be performed to detect and report errors that were introduced in the planning, development, or verification process. For that purpose, DO-278A recommends describing organizational responsibilities of the software verification process and interfaces to other software life cycles, verification methods like reviews, analysis, and testing methods. Verification environment, including software tools and hardware test equipment, criteria for starting verification process, reverification methods, and assumptions about the correctness of compiler, linker, and loader, should be included as well. Moreover, a description of test methods for partitioning integrity, how verification independence will be achieved, and verification methods for previously developed software and multiple-version dissimilar software should be included if applicable [6, p. 41, 73–74].

The methods for verifying requirements, design, and code are similar, but with different aspects [8, p. 123,153,178]. Hence, the description of the basic verification methods in the SVP will be separated from specific aspects to reduce redundancy. In consequence, the development-related review section will be structured as follows: At first, the SVP supplies organizational information for the software verification process as responsibilities and transition criteria. Then, a basic description of the proposed verification methods like review and analysis will be provided. Subsequently, the methods will be specified in subsections for requirements, design, and code.

Since almost all verification objectives of DO-278A can be satisfied by review [8, p. 84], a formal peer-review process shall be established as described in [8, p. 125]. With regard to the verification and tracing effort of DO-278A, the inspection review as described in [30] will be applied for reviewing software requirements, design, and code. These reviews are related to the “Review” stages in Fig. 4. To support the review process, it can also be necessary to utilize methods for analysis, e.g., test coverage analysis.

While the development-related verification objectives ensure that the development was performed accurately, there are also testing objectives for demonstrating software compliance to all requirements. The tests also demonstrate that failure conditions identified by SAP are removed. Therefore, compliance and robustness of EOC with all requirements need to be confirmed. In addition, it must be also validated that EOC is compatible with the target computer. For this purpose, DO-278A requires normal range and robustness test cases for requirement-based testing. This method tests requirements against the EOC and ensures the compatibility between software components. Additionally, it shows that the components comply with the software architecture [6, p. 46–51].

After software testing, the verification procedures need to be verified as well to ensure their correctness and suitability. Therefore, reviews and analysis of test cases, procedures, and results including test coverage analysis are required. In addition, bidirectional traceability between software requirements and their test cases is required as well to support the requirements-based test coverage analysis. A bidirectional trace between test cases and their procedures must be established as well for showing that all cases lead to test procedures. Then, to show that all test procedures were executed, a bidirectional trace between test procedures and their results must be created too [6, p. 51–53].

The test methods’ description in the SVP will be organized like the development-related verification methods. While a basic test method description will be added to the other basic methods, its specific characteristic will be attached to the other specific descriptions. Finally, the SVP ends with a section covering a reverification method.

3.2.4 Tool Qualification Planning

Software tools for reducing or automating processes of DO-278A without verifying its output must be qualified according to DO-330. The qualification must be regarded in the PSAA; otherwise, the qualification is not valid for the project. If a tool is qualified once, the qualification is only applicable for that project. Therefore, software tools qualified in another project must be requalified in to use them in a new project [6, p. 88–89].

3.2.5 Software Quality Assurance Plan

The Software Quality Assurance (SQA) process begins in the planning phase and continues during the software life cycle. Methods to be used for satisfying the objectives of this process must be described in the SQA Plan. For this purpose, the SQA Plan should include guidance for its environment, activities, records, transition criteria, and timing. Moreover, an authority statement and a supplier oversight should be provided as well. The latter describes how external developers comply with the project plans and standards. All SQA objectives must be satisfied independently. That is to say, only persons or tools that were not involved in corresponding development activities are allowed to conduct the verification for being independent [6, p. 56–63, 75–76, 133, 143–144].

3.2.6 Plan for Software Aspects of Approval

The PSAA is the first means for the approval authority to determine if the proposed software life cycle is appropriate to the assurance level and has a contract-like status between the applicant and the approval authority. Because the approval authority has the ability to reject the plans, it reduces the project risk, if the PSAA will be submitted early. The plan provides a system and software overview, the software life cycle including a summary of SDP, SVP, SCM Plan, and SQA Plan, as well as a list of data items that will be generated. Moreover, the plan should cover a project schedule and an oversight of external suppliers. Finally, it must also include approval considerations like the assurance level and additional considerations as the usage of COTS software. After finishing the PSAA, it will be submitted to the approval authority. Afterward, the authority can request further data or rework on the plans or agree with them [6, p. 67, 72–73], [8, p. 78–81].

As already mentioned, the planning phase is aligned to finish the PSAA as early as possible. Therefore, all plans and information needed for writing this plan are already available. The SDP with its life cycle model will be included completely, whereas the SVP, SCM Plan, and SQA Plan will be summarized. Moreover, a list of all to be created data items will be added. They will be listed with a short description of their contents. The remaining contents like overviews and approval considerations will be completely covered as well. After a review of the plans for their compliance to DO-278A, they will be submitted to the approval authority.

3.2.7 Additional Plans

Eventually, plans that are not relevant for DO-278A approval can be created. For instance, that are detailed plans for project management or scheduling company resources like flight tests.

3.3 Software Realization Phase

The software realization phase mainly consists of the execution of the plans. For this reason, the following subsections outline which data are required to start the process and which data items will be created. In addition, the organization of them will also be described with regard to the proposed software life cycle.

3.3.1 Software Requirements Process

The software requirements process can be started if the respective transition criteria as provided by the software life cycle are satisfied. Moreover, system requirements and architecture, hardware interfaces, and the SDP including the Software Requirements Standards must also be defined as they are the input of this process. This process ensures that the High-Level Requirements (HLRs) are developed and the derived ones are defined and provided to system development and SAP. Derived requirements are not directly traceable to higher-level ones or define behavior beyond them. The Software Requirements Data are the process output, which contains the HLRs. Furthermore, Trace Data show the association between system requirements and HLRs will be allocated as well. It provides the verifiability that the requirements are implemented completely and visibility to derived HLRs [6, p. 34, 39, 142].

Once the requirements are developed, they will be reviewed according to the SVP. Likewise, the segmentation of development and verification plan, the resulting data of this process will be allocated. The requirements development-related Software Requirements Data and Trace Data will be gathered in one standalone document, whereas the Software Verification Results (SVR) will be gathered in a separate document. To ensure bidirectional traceability, the system requirements will also be supplemented with a reference to the relevant HLRs.

3.3.2 Software Design Process

After the planned transition criteria are satisfied, the software design process can be started. That process can be started if the planned transition criteria are satisfied. Further, the Software Requirements Data and SDP including Software Design Standards need to be defined as they are the input of this process. This process must ensure that the Low-Level Requirements (LLRs) and the software architecture are developed from the HLRs. Moreover, it ensures that derived LLRs are provided to the system development and the SAP too. LLRs are a breakdown of the HLRs of those no further information is required for code development. The Design Description is the process output, which contains the LLRs and software architecture. Additionally, Trace Data that show the association between HLRs and LLRs will be allocated as well. Trace Data provide verifiability that the requirements are implemented completely and visibility to derived LLRs [6, p. 35, 39,144].

If the software design is created, it will be reviewed according to the SVP. The resulting data of this process will be segmented analog to the resulting data of the requirements process. For this reason, the Design Description will be combined with the corresponding Trace Data in one document and the corresponding verification will be added to the SVR document created in the requirements process. The HLRs will be supplemented with a reference to the relevant LLRs as well as to archive bidirectional traceability.

3.3.3 Software Implementation Process

The software implementation process consists of coding and integration and can be started if the respective transition criteria as provided by the software life cycle are satisfied. Furthermore, the software architecture, LLRs, SDP, and Software Code Standards must also be available as they are its input. The coding process uses software architecture and LLRs for writing Source Code. Then, the integration process compiles, links, and loads the Source Code into the target computer. As a result of this process, the EOC and Adaption Data Item File will be output as well as data for compiling, linking, and loading. Moreover, Trace Data showing the association between LLRs and Source Code will be allocated as well. It supplies verifiability that the LLRs are fully implemented as well as there is no undocumented function [6, p. 37–39].

If the software is implemented, it will be reviewed according to the SVP. The resulting data of this process will be segmented as mentioned before. Therefore, Trace Data references will be integrated into the Source Code, whereas the corresponding review results will be added to the SVR document. Additionally, LLRs will be supplemented with a reference to the relevant Source Code as well.

3.4 Software Verification Phase

The reviews for the development processes are already covered in the respective requirements, design, and implementation process. This phase focuses on software testing and the verification of testing.

3.4.1 Software Verification Process

System and software requirements, software architecture, Trace Data, Source Code and EOC, as well as the SVP are input into the software verification process. The Software Verification Cases and Procedures, SVR, and Trace Data are its output. After satisfying the relevant transition criteria, this process can be started. Errors that arose in the development will be detected and reported in this process. The corresponding development processes will remove them. For this purpose, Software Verification Cases and Procedures will be used for testing and developed as specified in the SVP. They define their scope and extent, identify required equipment, and describe the expected test layout. Moreover, inputs, conditions, expected results, coverage criteria, and pass or fail criteria should be defined for each test case as well [6, p. 41–42, 79], [8, p. 206–207].

If the Software Verification Cases and Procedures are defined, they will be reviewed according to the SVP. Therefore, the Software Verification Cases and Procedures including its Trace Data will be combined in one document. The bidirectional Trace Data references will be demonstrated with a traceability matrix. The review results will be added to the SVR document like before. Moreover, the results of the tests will be added to the SVR document as well.

3.4.2 Software Accomplishment Summary

The Software Accomplishment Summary shows that the developed software is compliant to the PSAA. For this purpose, the system and software overview, approval considerations, software life cycle, additional considerations, and the suppliers' oversight will be included like in the PSAA. However, the Software Accomplishment Summary emphasizes on their changes. In addition, while the PSAA mentioned to be created software life cycle data, the Software Accomplishment Summary includes that data. Moreover, the software configuration with part numbers and version as well as its characteristics like EOC size and timing margins need to be included. Then, a change history focusing on changes caused by safety defects and life cycle improvements must be added if applicable. Additionally, if there are unresolved problem reports, a summary of them with potential safety effects, functional and operational restrictions, etc. also needs to be included. Finally, the Software Accomplishment Summary must include a compliance statement that provides how the compliance was demonstrated and list additional rules from the approval authority and deviations from the plans if they are not already addressed [6, p. 81–82].

After the Software Accomplishment Summary is finished, it will be submitted to the approval authority. If they approve it, the software will be taken into operation.

4 Conclusion

This paper aimed to show a concept for approving the software of GLASS. For this purpose, standards and further considerations that have an immediate influence on the approval according to DO-278A were introduced at first. Then, DO-278A was analyzed according to assurance level 3 for objectives that need to be satisfied. Based on them and the concomitant activities, an approval concept was created. The concept follows a traditional approach, which includes an adapted V life cycle model for software development. Initially, the concept schedules the creation of the required plans, outlines its contents, and presents our implementation approach. That planning phase is aligned to finish the Plan for Software Aspects of Approval as early as possible, so that the approval authority examines it early in the project. Accordingly, the plans for configuration management, software development and verification, quality assurance, and tool qualification must be finished first. After the approval authority has confirmed the plan, the software will be implemented and verified according to the respective plans. Ultimately, the software accomplishment summary will be written and submitted as well. If it will be approved, the software can be taken into operation.

Acknowledgements The authors would like to thank the German Ministry for Economic Affairs and Energy for funding the GLASS project under Grant No. 20Q1933C.

Funding Open Access funding enabled and organized by Projekt DEAL.

Declarations

Conflict of Interest Thomas Dautermann reports financial support was provided by German Ministry for Economic Affairs and Energy.

References

1. Sanders L, Fritch V (1973) Instrument landing systems. IEEE Trans Commun 21:435–454. https://doi.org/10.1109/TCOM.1973.1091710
2. Dautermann T, Felux M, Grosch A (2012) Approach service type D evaluation of the DLR GBAS testbed. GPS Solut 16:375–387. https://doi.org/10.1007/s10291-011-0239-3
3. Dautermann T (2014) Civil air navigation using GNSS enhanced by wide area satellite based augmentation systems. Prog Aerosp Sci 67:51–62. https://doi.org/10.1016/j.paerosci.2014.01.003
4. RTCA, Inc. (2011) DO-178C—software considerations in airborne systems and equipment certification. Dec 2011
5. RTCA, Inc. (2021) DO-178C: software considerations in airborne systems and equipment certification—errata 1. Feb 2021
6. RTCA, Inc. (2011) DO-278A—software integrity assurance considerations for communication, navigation, surveillance and air traffic management (CNS/ATM) systems. Dec 2011
7. RTCA, Inc. (2021) DO-278A: software integrity assurance considerations for communication, navigation, surveillance and air traffic management (CNS/ATM) systems—errata 1. Feb 2021
8. Rierson L (2013) Developing safety-critical software: a practical guide for aviation software and DO-178c compliance. CRC Press/Taylor & Francis Group, Boca Raton
9. Dautermann T, Ludwig T, Geister R, Ehmke L (2020) Extending access to localizer performance with vertical guidance approaches by means of an SBAS to GBAS converter. GPS Solut 24(2):37. https://doi.org/10.1007/s10291-019-0947-7
10. Lipp A (2023) ECAC GBAS implementation. International GBAS Working Group, San Francisco
11. SAE International (2010) ARP4754A—guidelines for development of civil aircraft and systems. Dec 2010
12. International Civil Aviation Organization (2018) Annex 10 to the convention on civil aviation—aeronautical telecommunication. July 2018
13. RTCA, Inc. (2000) DO-254—design assurance guidance for airborne electronic hardware. Apr 2000
14. RTCA, Inc. (2011) DO-330—software tool qualification considerations. Dec 2011
15. SAE International (1996) ARP4761—guidelines and methods for conducting the safety assessment process on civil airborne systems and equipment. Dec 1996
16. Verordnung über Art, Umfang, Beschaffenheit, Zulassung, Kennzeichnung und Betrieb von Anlagen und Geräten für die Flugsicherung (Flugsicherungs-Anlagen- und Geräte-Musterzulassungs-Verordnung - FSMusterzulV). Dec 2001. [Online]. https://www.baf.bund.de/SharedDocs/Downloads/DE/Publikationen_BAFReferate/ST/ST_FSMusterzulV.pdf?__blob=publicationFile&v=2
17. DFS Deutsche Flugsicherung GmbH (2008) NfL II 51/08 - Bekanntmachung über die Anforderungen zur Musterzulassung von GBAS-Bodenanlagen als Flugnavigationsfunkstelle. Sep. 2008. [Online]. https://www.baf.bund.de/SharedDocs/Downloads/DE/Publikationen_BAFReferate/ST/ST_MusterZ-NfL-II-51-08-GBAS.pdf?__blob=publicationFile&v=2
18. RTCA, Inc. (2011) DO-248C—supporting information for DO-178C and DO-278A. Dec 2011
19. European Organization for Civil Aviation Equipment (EUROCAE) (2013) ED-114A—minimum operational performance specification for global navigation satellite ground based augmentation system ground equipment to support category I operations. Mar 2013
20. European Union Aviation Safety Agency (2020) Certification specifications and acceptable means of compliance for large aeroplanes CS-25. Dec 2020. [Online]. https://www.easa.europa.eu/sites/default/files/dfu/cs-25_amendment_26_0.pdf
21. RTCA, Inc. (2021) DO-248C: supporting information for RTCA/DO-178C and RTCA/DO-278A—errata 1. Feb 2021
22. RTCA, Inc. (2021) DO-330: software tool qualification considerations—errata 1. Feb 2021
23. RTCA, Inc. (2011) DO-331—model-based development and verification supplement to DO-178C and DO-278A. Dec 2011
24. RTCA, Inc. (2021) DO-331: model-based development and verification supplement to RTCA/DO-178C and RTCA/DO-278A—errata 1. Feb 2021
25. RTCA, Inc. (2011) DO-332—object-oriented technology and related techniques supplement to DO-178C and DO-278A. Dec 2011
26. RTCA, Inc. (2011) DO-333—formal methods supplement to DO-178C and DO-278A. Dec 2011
27. PATMOS Engineering Services, Inc. (2021) DO-178C templates for evaluation. [Online]. https://www.avionics-certification-academy.com/courses/129421/lectures/1894924. Accessed 26 Mar 2021
28. European Organization for Civil Aviation Equipment (EUROCAE) (2019) ED-114B—minimum operational performance specification for global navigation satellite ground based augmentation system ground equipment to support category I operations. Sept 2019
29. Kleuker S (2018) Grundkurs Software-Engineering mit UML: Der pragmatische Weg zu erfolgreichen Softwareprojekten. Springer Vieweg, Wiesbaden. [Online]. https://doi.org/10.1007/978-3-658-19969-2
30. Stellman A, Greene J (2006) Applied software project management. Online, 2006
31. Hilderman V, Baghi T (2007) Avionics certification: a complete guide to DO-178 (software), DO-254 (hardware). Avionics Communications, Leesburg