LDRA
Software Technology

Technical White Paper

Safety Assurance Through Independent Verification.
A Focus on RTCA DO-178C & DO-248C

www.ldra.com

Contents

1.0 Introduction

In the realm of safety-critical avionics software, ensuring reliability, accuracy, and compliance with regulatory standards is paramount. One of the fundamental principles that underpin RTCA DO-178C (Software Considerations in Airborne Systems and Equipment Certification) and RTCA DO-248C (Supporting Information for DO-178C and DO-278A) is Independence in Verification. This concept mandates that verification activities—such as reviews, analysis, testing, and structural coverage assessments—must be conducted by individuals who were not involved in the development of the given software life cycle item. By enforcing an independent verification process, the aviation industry mitigates the risk of undetected errors, reduces bias, and enhances the overall safety and integrity of airborne systems.

Independence is particularly critical for higher Development Assurance Levels (DALs), such as DAL A (Catastrophic) and DAL B (Hazardous), where failures could result in loss of life or severe safety implications. This requirement extends not only to manual verification efforts but also to the qualification of software verification tools, as governed by DO-330 (Software Tool Qualification Considerations) and extends to DO-178C technology supplements (DO-331, DO-332 and DO-333). For instance, object code verification and model-based systems engineering pose unique challenges and opportunities around independent verification.

This paper explores the concept of verification independence, extending beyond software issues to encompass system and hardware assurance. It delves into the principles, intent, guidance requirements, including guidance from FAA AC 20- 115d and RTCA DO-248C, challenges and best practices in independent verification in avionics software development. The paper elucidates the mechanisms by which organisations can attain regulatory compliance and ensure robust software assurance through the implementation of structured, independent verification methodologies. It provides a comprehensive analysis of how independence can be effectively achieved through the utilisation of qualified tools.

2.0 Understanding Independent Verification

Independent Verification refers to the process of evaluating a system or component by an entity that is not involved in its development. This separation ensures objectivity and reduces the risk of biases that could *compromise* the integrity of the verification process. A common tendency in software development is for teams to prioritise familiar practices over strict adherence to established guidelines. Independent verification is crucial for identifying potential issues that developers might overlook due to familiarity or confirmation bias. On a high level, this means that verification should be done from requirements through an independent Software verification team rather than by the software developers or system designers.

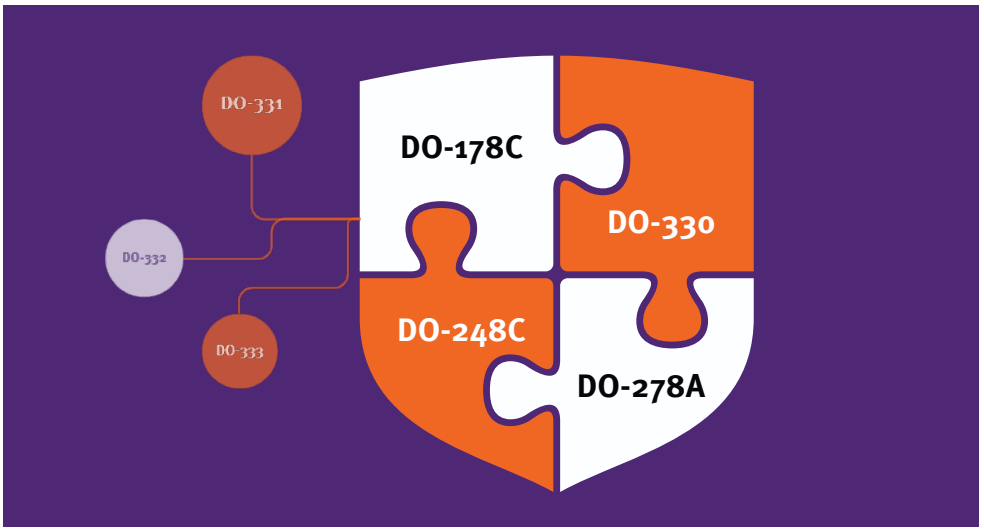
3.0 Guidance and References

Below is the guidance and References from DO-178C, DO-248C, DO-330 and Technology supplements (DO-331, DO-332 & DO-332) on Independence in Verification process activities.

RTCA DO-178C, “Software Considerations in Airborne Systems and Equipment Certification,” is a key guidance document for certifying software in airborne systems. It outlines rigorous processes and objectives for software development and verification to ensure safety and reliability.

The need for independence in safety-critical systems verification arises from the limitations of self-assessment, the need for objective evaluation, and the potential for catastrophic consequences. While DO-178C does not always explicitly *require* full organisational independence, its emphasis on rigorous verification, objective evidence, and building confidence strongly suggests that some degree of independent thought, execution, and evaluation is essential, especially for higher criticality levels. Independent verification enhances defect detection efficiency, increases confidence, and facilitates compliance with regulatory requirements, ultimately contributing to the safety and reliability of these critical systems.

One of the core principles of DO-178C is the need for independence in verification activities. This guidance document categorizes verification tasks into different levels of independence based on the criticality of the software. For higher Development Assurance Levels (DALs), such as DAL A and B, which are associated with the most critical functions, the standard mandates independence in verification activities.



3.1 Specific Guidance from DO-178C

3.1.1 DO-178C Annex A – Objectives Requiring Independence

- DO-178C explicitly requires independence for certain verification objectives based on software criticality levels (DAL A & B). Key References include.
- Tables A-3, A-4, A-5, A-6, A-7, and A-9 delineate the objectives necessitating independent verification, contingent upon the Design Assurance Level (DAL).

3.1.2 Section 6.0 – Verification Process

- Requires that verification activities are performed independently for DAL A & B software.
- Advocates that independent verification personnel must not be involved in the development of the software life cycle item, as they are verifying.

3.2 Guidance from DO-248C on Independence in Verification

3.2.1 Understanding DO-248C's Role

DO-248C is a companion document to DO-178C, providing clarifications, FAQs, and rationale behind specific objectives in DO-178C. Independence in verification is a key concern in software certification for safety-critical aircraft systems. While DO-178C defines the requirements for software verification, DO-248C explains the intent, rationale, and best practices behind those requirements.

DO-248C expands on this topic by addressing common industry questions or frequently asked questions (FAQs) and explanations, providing additional guidance on how independence should be interpreted and implemented, and clarifying the rationale for its necessity.

3.2.2 Key References from DO-248C

- Discussion Paper- DP #19: Independence in DO-178C/DO-278A
- Figure 4-2 Independence and Verification Objectives Illustrated
- Table 4-5 Independence Interpretation for Verification Objectives

3.2.3 DP #19: Independence in DO-178C/DO-278A

In accordance with DO-178C/DO-278A, certain verification objectives must be achieved with independence, particularly for specific software/assurance levels, to ensure objectivity and mitigate the risk of systematic errors. Independence is maintained by using different personnel or tools for verification than those involved in development, with separate personnel reviewing verification outputs when necessary. This prevents misinterpretations from propagating through both design and verification. However, achieving independence ensures that verification activities are conducted separately from the development process.

Below is the summary of guidance from DO-248C Discussion Paper DP #19.

- **Verification Objectives:** Some verification objectives in DO-178C/DO-278A require independence for certain software/assurance levels.
- **Different Means:** Use different personnel or tools for verification than those used for development to avoid misinterpretations.
- **Independent Review:** Independent personnel may review verification outputs to ensure objectivity.
- **Purpose of Independence:** Prevents a single misinterpretation from affecting both design and verification.
- **No Different Organisation Needed:** A different reporting structure or company is not required.
- **Annex A:** Defines which objectives need independence by software/assurance level.
- **Definition of Independence:** Separation of responsibilities to ensure objective evaluation.
    - **Software Verification:** Performed by someone other than the developer, with tools possibly used for equivalence.
    - **Software Quality Assurance (SQA):** Includes authority to ensure corrective actions.
- **Section 6.2.e:** Verification independence means the activity is performed by someone other than the developer, with tools possibly used for equivalence.
    - **Test Cases and Source Code:** The person creating test cases should not be the same person developing the source code.
- **SQA Functions:** Should be performed by personnel with authority, responsibility, and independence to ensure SQA objectives are met.

Elaborating further from this DP#19, in the form of notes, DO-178C/DO-278A defines independence requirements for verification activities to ensure objective evaluation and avoid errors propagating from development to verification.

- **Objective 9 (Table A-7)** is not discussed here because additional code, typically generated by tools like compilers, is independently verified.
- **Objectives 8 & 9 (Table A-5)** apply when Parameter Data Items (PDI) are used. Independence is ensured by:
    - A different person verifying the **PDI File** than the one who created it.
    - Another independent reviewer ensuring that the verification itself was properly performed.
- **Objectives 3 & 4 (Table A-6)** require that low-level requirements-based tests be developed by someone other than the Source Code developer to maintain independence. However, high-level requirements-based tests may be created by the same person responsible for Software Design or Source Code, as high-level requirements development is generally more collaborative.

In summary, independence in verification, quality assurance, and multi-version development is a fundamental requirement in DO-178C/DO-278A, that helps in reinforcing aviation safety by eliminating systemic vulnerabilities and ensuring compliance with regulatory certification requirements.

Qualified tools can be used to achieve independence in **DO-178C** by providing an objective verification method separate from development. The guidance allows tools to replace human verification if they achieve equivalence in evaluation. Properly qualified tools (per DO-330) ensure errors do not propagate, meeting independence requirements without requiring separate personnel or organisations.

Figure 4-2 in document DO-248C, as presented in DP#19, visually depicts the objectives of independence and verification. Additionally, Table 4-5 offers a detailed interpretation of independence concerning verification objectives, specifically addressing the applicable DO-178C verification objectives outlined in Annex A-3 to A-7.

3.3 Guidance from DO-178C Technology Supplements (DO-331, DO-332, DO-333) on Independence in Verification

The principle of independence in verification is fundamentally a core requirement within DO-178C, driven by the Design Assurance Level (DAL). It is not the primary focus of these specific technology supplements. The objectives detailed in Tables A-3 through A-9 of Annex MB.A (DO-331,Model-Based Development and Verification Supplement to DO-178C and DO-278A), Annex OO.A (DO-332, Object-Oriented Technology and Related Techniques Supplement to DO-178C and DO-278A), and Annex FM.A (DO-333, Formal Methods Supplement to DO-178C and DO-278A) provides thorough guidance that reinforces the principles of independence in verification. These objectives serve to augment and refine the independence criteria outlined within the DO-178C verification activities, specifically tailored to these technologies. Model-based development poses unique challenges, as often verification can be done at the model scope and by the modelling tools. This can lead to cases where the model-generated code is aligned to the model-generated world, but may have discrepancies or polarity issues with the real world. In this case, performing additional verification at the requirement scope is especially important.

3.4. Regulatory Expectations on Independence in Verification

Regulatory compliance ensures that aviation certification authorities (e.g., FAA, EASA) accept the software verification process and its independence requirements as per DO-178C Annex A. This is critical for obtaining software approval for airborne systems. Applicants must coordinate with certification authorities to ensure project-specific compliance.

3.4.1 Key Aspects of Regulatory Compliance

3.4.1.1 Compliance with Certification Authorities

- **FAA (Federal Aviation Administration)**: FAA Advisory Circular AC 20-115D, endorses DO-178C as the accepted means of showing compliance for airborne software.
- **AC 20-115D** reinforces the need for independent verification and validation (IV&V) to ensure software correctness and compliance with safety standards, and the role of DO-330 (Software Tool Qualification Considerations) for verifying software tools used in development and testing.
- **EASA (European Union Aviation Safety Agency)**: Uses AMC 20-115D, which is aligned with FAA AC 20-115D and emphasizes independence in software verification for higher DALs (A & B).
- **Other Global Civil Aviation Regulators**: Generally, follow FAA and EASA guidance.

Key Requirement: Applicant must demonstrate that verification activities, particularly for DAL A & B, are conducted independently from the developers of the selected artifacts per DO 178C Anex-A Objectives.

3.4.1.2 Certification Process and Independence Verification

To comply with DO-178C, applicants must provide documentation that demonstrates the following:

- Verification Independence
  - Provide verification evidence that the review/test was conducted by a person not involved in development.
  - Show separation of responsibilities between development and verification in the Software Development Plan (SDP).
- Verification Tools & Traceability
  - Use independent tools (e.g., static analysers, Dynamic test tools and requirement traceability tools) to support verification.
  - Maintain a traceability matrix to ensure each requirement has been verified independently.

- Process Assurance (DO-178C Section 9.4.3)
  - Ensure independent audits are conducted for Process Assurance (Annex A, Table A-9 Software Quality Assurance Process).
  - Process Assurance must confirm that verification objectives are met independently for DAL A & B software.
- Certification Liaison with Authorities
  - Engage with FAA/EASA/TCCA Designated Engineering Representatives (DERs) early to confirm verification independence expectations.
  - Conduct SOI (Stage of Involvement) audits with authorities to ensure compliance during software development.

4.0 Challenges and best practices in implementing Independent Verification

Implementing independent verification in safety-critical systems, such as those governed by RTCA DO-178C, presents several challenges. Addressing these challenges requires careful planning, robust processes, and a commitment to maintaining high standards throughout the software development lifecycle.
Here are some key challenges and the proposed industry best practices/solutions to address them. By implementing these solutions, organisations can overcome the challenges of independent verification considerably and ensure the safety and reliability of their safety-critical systems and satisfy the regulatory requirements.

4.1 Resource Management

- **Strategic Planning**: Allocate resources strategically by prioritising critical verification activities. Use a phased approach to distribute workload and manage resources effectively
  - Implement a phased approach where verification starts early in development (e.g., continuous integration & incremental verification).
  - Allocate resources based on expertise (e.g., separate teams for requirements validation, test case development, and review processes).
  - Utilise Early Defect Detection methods such as static analysis, model checking, and simulations before full-fledged testing.
  - Enable concurrent engineering by having independent verification teams work in parallel with development teams.
- **Outsourcing**: Consider outsourcing verification tasks to specialised third-party organisations with expertise in independent verification, which can also enhance efficiency and leverage specialised expertise. While outsourcing can introduce managerial and technical challenges, such as overseeing the suppliers' processes and ensuring the quality of their output, as applicant is responsible for providing appropriate oversight on the supplier.
  - Making the supplier follow the applicants' process can be a good approach to ensuring there is no conflict between multiple processes. It also makes it easy to have good control over the process followed by the supplier, making it easier to audit and show compliance.
  - Importantly, while outsourcing allows for greater independence, compliance must still be considered at the item scope. Audit and compliance burden cannot entirely be placed on the supplier organisation.
- **Outsourcing - Alternate Approach:** Organizations can consider establishing separate internal teams with expertise in independent verification as an alternative approach which can bring in the following benefits.
  - **Enhanced Control:** Organisations maintain direct oversight of verification processes, ensuring adherence to internal standards and protocols.
  - **Improved Communication**: Internal teams can collaborate more effectively with other departments, leading to quicker resolution of issues and better alignment with organizational goals.
  - **Confidentiality:** Sensitive information remains within the organisation, reducing the risk of data breaches or leaks and protecting the confidentiality of the information.
  - **Consistency:** Internal teams can develop a deep understanding of the organisation's systems and processes, leading to more consistent and reliable verification outcomes.
  - **Flexibility:** Internal teams can quickly adapt to changes in project requirements or organisational priorities without renegotiating with external parties. Helps in the ease of resource allocation and management.

4.2 Complexity of Systems

- **Modular Verification**: Break down complex systems into smaller, manageable modules for verification. This approach simplifies the process and allows for focused verification of each component.

4.3 Maintaining Objectivity

- **Clear Separation of Roles**: Establish clear separation of roles between development and verification teams/functions to maintain objectivity. Implement policies to ensure independence in verification activities.
- **Regular Audits**: Conduct regular audits to ensure that verification activities remain unbiased and independent.

4.4 Automated Verification Tools and Tool Qualification

- **Advanced Tools:** Utilize advanced and automated verification tools and techniques, such as formal methods and model checking and AI/ML based tools, to handle the complexity of safety-critical systems. RTCA DO-330 provides detailed guidance on Tool Qualification.
- Qualified tools can greatly enhance the quality and efficiency of the verification process, help in showing compliance to independence requirement of DO-178C and provide required compliance documentation supporting faster compliance finding and approvals.
- **Tool Qualification Plans**: Develop comprehensive tool qualification plans that outline the criteria and processes for qualifying verification tools aligned with RTCA DO-330. Have a foresight to see how common tools can be used across multiple product lines and accommodate for the same in Tool Qualification Plans.
- **Continuous Monitoring**: Continuously monitor and update tools to ensure they meet the required standards and produce reliable results for a given program and Hardware platform. This flows into project-specific issues and mitigations as is appropriate.

4.5 Keeping Up with Technological Advances

- **Continuous Training**: Invest in continuous training and professional development for verification teams to keep them updated with the latest technologies and methodologies.
- **Collaboration with Experts**: Collaborate with industry experts and research institutions to stay informed about emerging trends and best practices.

4.6 Integration with Development Processes

- **Agile Verification**: Integrate verification activities into agile development processes to ensure continuous verification throughout the development lifecycle. This can speed final verification for the record by performing verification tasks during the agile sprint process. In addition, this helps keep the Verification team engaged during development and update processes.
- **Automated Testing**: Implement automated testing frameworks to streamline verification activities and reduce manual effort.

4.7 Regulatory Compliance

- **Compliance Management Systems**: Implement compliance management systems to track and manage regulatory requirements. Use these systems to ensure all verification activities align with relevant standards.
- **Regular Updates**: Stay updated with changes in regulatory guidance and adjust verification processes accordingly.

4.8 Communication and Coordination

- **Effective Communication Channels**: Ensure effective communication channels between development and verification teams to ensure clear and timely information exchange.
- **Collaborative Tools**: Use collaborative tools and platforms to facilitate coordination and documentation of verification activities.

5.0. Role of Qualified Tools

Qualified tools play a *very* significant role in strengthening the independence and effectiveness of verification activities for safety-critical systems. Here's how they contribute:

5.1 Reducing Bias and Increasing Objectivity

- **Independent Tool Chains:** When the verification team uses a different set of qualified tools than the development team, it reduces the risk of shared blind spots. Developers might be accustomed to the quirks and limitations of their tools, potentially overlooking errors that another tool might catch. Independent tools offer a fresh perspective.
- **Automated Analysis:** Qualified tools often provide automated analysis capabilities (e.g., static analysis, model checking). Automation reduces the potential for human bias in the verification process, ensuring a more objective evaluation of the system.

5.2 Enhancing Thoroughness and Coverage

- **Comprehensive Testing:** Qualified testing tools can help generate more comprehensive test suites, ensuring better coverage of the system's functionality and edge cases. This is particularly important for achieving the required levels of code coverage (e.g., MC/DC) for safety-critical systems.
- **Formal Analysis:** Qualified formal methods tools allow for rigorous mathematical analysis of the system's design and code. This can uncover subtle flaws that might be missed by traditional testing methods.
- **Requirements Traceability:** Qualified requirements management tools help maintain precise traceability between requirements, design elements, code, and test cases. This makes it easier for independent verifiers to ensure that all requirements are properly addressed and verified.

5.3 Improving Efficiency and Repeatability

- **Automation:** Qualified tools can automate many verification tasks, such as test case generation, test input population, test execution, results analysis, and report generation. This saves time and resources, allowing for more thorough verification within the project's constraints.
- **Repeatability:** Automated verification processes using qualified tools ensure consistent and repeatable results. This is crucial for demonstrating compliance to certification authorities and ensuring that verification activities can be easily reproduced.

5.4 Facilitating Compliance with Standards

- **DO-178C Compliance:** For instance, DO-178C (and similar standards) often have specific requirements for tool qualification. Using qualified tools demonstrates adherence to these standards, making it easier for the verification team to demonstrate compliance during audits.
- **Artifact Generation:** Qualified tools can often generate required documentation and reports, such as test summaries, coverage reports, and traceability matrices compliant with DO-178C requirements. This simplifies the verification process and reduces the documentation burden.

5.5 Building Confidence and Trust

- **Credibility:** Using qualified tools adds credibility to the verification process. It shows that the verification team is using industry-standard tools and best practices, which increases confidence in the results.
- **Auditability:** Qualified tools often provide audit trails and logs, making it easier to demonstrate the rigour and thoroughness of the verification process to certification authorities.

5.6 Challenges and Considerations of Using Qualified Tools

- **Tool Qualification Effort:** Qualifying tools can be a significant undertaking, requiring substantial effort and documentation. This needs to be factored into the project plan.
- **Cost:** Qualified tools can be expensive. However, the cost should be weighed against the benefits they provide in terms of ease of showing compliance, improved safety and reduced risk.
- **Training:** Verifiers need to be properly trained on how to use the qualified tools effectively.

6.0 Comprehensive Tool Suite from LDRA

LDRA tool suite® can be an asset in achieving independent verification for safety-critical systems. Here's how they can help:

i. **Comprehensive Tool Suite:**
LDRA provides a comprehensive suite of tools that support multiple stages of the software development lifecycle—including requirements traceability, requirement-based testing, static analysis, dynamic testing, and code coverage analysis. This integrated approach helps streamline verification processes and enhance overall development efficiency.

ii. **Support for Safety-Critical Standards:**
LDRA tools are designed to support compliance with safety-critical standards like DO-178C (for aerospace), ISO 26262 (for automotive), and IEC 61508 (for industrial applications). They can help you meet the stringent requirements of these standards and demonstrate compliance to certification authorities.

iii. **Requirements Traceability:**
LDRA tools facilitate requirements traceability by linking requirements to design elements, code, and test cases. This makes it easier for independent verifiers to ensure that all requirements are properly addressed and verified.

iv. **Static and Dynamic Analysis:**
The LDRA tool suite® provide both static and dynamic analysis capabilities. Static analysis can help identify potential coding errors, vulnerabilities, and deviations from coding standards early in the development process. Dynamic analysis, on the other hand, can help verify the runtime behaviour of the software and identify issues that might not be caught by static analysis alone.

v. **Code Coverage Analysis:**
The LDRA tool suite® can perform code coverage analysis to demonstrate that the test cases exercise all parts of the code. Achieving the required levels of code coverage (e.g., MC/DC) is essential for safety-critical systems, and LDRA tools can help you achieve and demonstrate this.

vi. **Test Automation:**
The LDRA tool suite® can automate many testing tasks, such as test case generation, test execution, and results analysis. This can save considerable time and resources, allowing for more thorough testing within the project's constraints.

vii. **Tool Qualification Support:**
LDRA provides support for tool qualification, which is often a requirement for safety-critical systems. LDRA offers qualification support packages and documentation to help demonstrate that the tools are reliable and produce consistent results.

viii. **Independent Verification and Validation:**
By providing a comprehensive set of tools and supporting various aspects of the verification process, the LDRA tool suite® can empower independent verification teams to perform their tasks more effectively and efficiently. They can help ensure that the verification process is thorough, objective, and compliant with relevant standards.

ix. **Worst-Case Execution Time (WCET) and execution time analysis in critical embedded systems**
Independent verification demands a rigorous understanding of software timing behaviour, making Worst-Case Execution Time (WCET) analysis a critical element. Ensuring this analysis is performed independently helps minimise bias and maximises the integrity of safety-critical embedded systems.

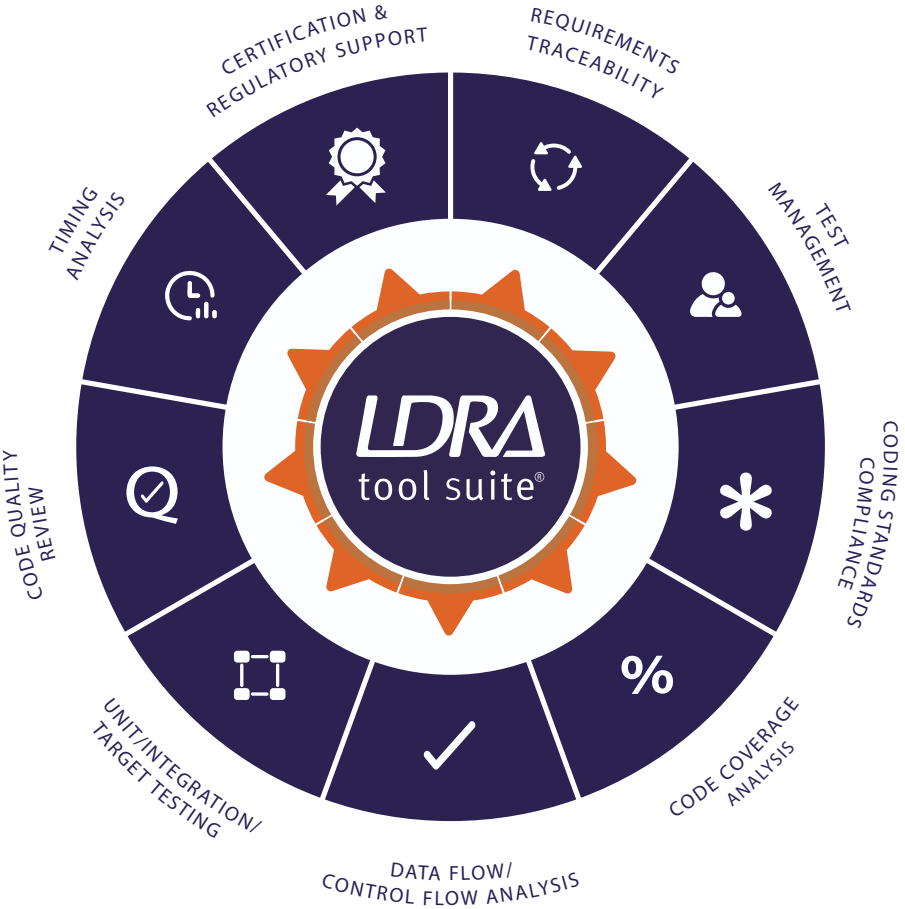
In complex, multi-core situations, WCET measurement and analysis are particularly important as execution time becomes more probabilistic rather than deterministic in these systems.

The LDRA tool, TBwcet, automates the measurement of Worst-Case Execution Times (WCET) on the target, thus addressing this critical need by providing robust static and dynamic analysis capabilities specifically designed for WCET and execution time analysis. Its independent approach utilises control and data flow analysis to meticulously identify all possible execution paths, leading to more reliable WCET estimations. Complementing this, the tool can be easily integrated with various environments, including hardware and simulators, to measure actual execution times, providing an objective assessment against real-time constraints.
Ultimately, LDRA's comprehensive solution, with its traceability, automated reporting, and certification support, empowers independent verification efforts, resulting in more accurate, efficient, and regulatorily compliant Worst Case Execution Time (WCET) analysis for safety-critical applications.

x. Integration with Other Tools:

The LDRA tool suite® can be integrated with other tools with ease in your development environment, such as requirements management tools, modelling tools, and configuration management tools, compilers and IDEs. This can further streamline and enhance efficiency in the development and verification process.

While integration and independence may seem at odds, a well-designed integration strategy can, in fact, enhance the independence of verification activities. A seamless flow throughout the digital process lifecycle facilitates the efficient incorporation of necessary modifications. For example, as requirements or measurements evolve, corresponding verification activities must be promptly updated. Requirement traceability and comprehensive impact analysis are essential technological enablers for managing these updates in a timely and cost-effective manner, thereby strengthening the overall verification process.



6.1 The LDRA tool suite® Mapping to DO-178C Life Cycle Activities

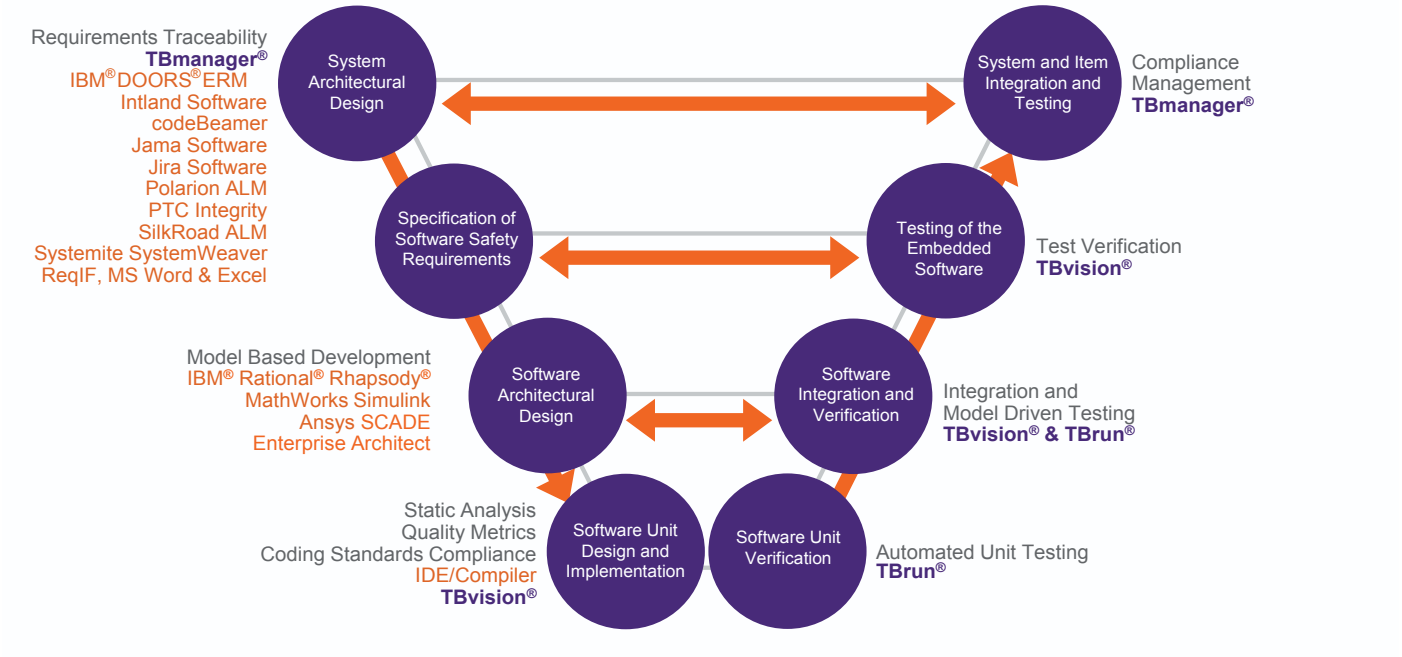
The table below offers a comprehensive overview of each component within the LDRA tool suite®. It also includes references to relevant sections of DO-178C, aiding applicants in understanding the capabilities of each component in the LDRA tool suite®

LDRA Tool	Purpose	DO-178C Life Cycle Activities Supported
TBmanager®	End-to-end requirements traceability, verification management	- Software Requirements Process (5.1) - Software Design Process (5.2) - Software Coding Process (5.3) - Test Planning (6.0) - Test Case Development (6.4.2) - Test Execution (6.4.3)
TBvision®	Visualisation of coding standard violations, Code Quality, and System tests	- Software Requirements Process (5.1) - Software Design Process (5.2) - Structural Coverage Analysis (6.4.4.2)
TBrun®	Automated unit testing and dynamic analysis	- Test Case Development (6.4.2) - Test Execution (6.4.3)
DDFC (Dynamic Data Flow Coverage)	Automatic control and data flow analysis	- Software Design Process (5.2) - Structural Coverage Analysis (6.4.4.2)
TBsafe®	Modified Condition Decision Coverage (MC/DC) analysis. Adds Information Flow Analysis when used in conjunction with the LDRA tool suite®.	- Test coverage of software structure (modified condition/decision coverage) is achieved. (6.4.4.c)
TBvision®	Code coverage analysis for statement, branch, and MC/DC coverage	- Structural Coverage Analysis (6.4.4.2)
TBvision®	Automated code review for coding standard compliance (MISRA, CERT, etc.)	- Software Coding Process (5.3)
TBwcet®	TBwcet® automates the measurement of Worst-Case Execution Times (WCET) on the target	- Review and analysis of source code (6.3.4) – f. Accuracy and consistency
TBobjectbox®	TBobjectbox® provides a complete Object Code Verification (OCV) solution, including source code to object code traceability and object code coverage analysis.	- 6.4.4.2.b Structural Coverage Analysis need to be done on the source code, object code and executable object code.

6.2 V Model – LDRA tool suite® for DO-178C Life Cycle Activities

The figure below illustrates the DO-178C lifecycle activities supported by individual tools within the LDRA tool suite®. TBmanager® is depicted as offering plug-ins for a seamless interface with other industry requirements management tools, facilitating efficient management of Requirements Traceability. Similarly, the LDRA tool suite® provides seamless integration with Model-Based development tools from various industry vendors.

For more details on the LDRA tool suite® for DO-178C Life Cycle Activities, visit *https://ldra.com/products/ldra-tool-suite/.*



7.0 Conclusion

As systems become increasingly complex, the need for independent verification will only continue to grow, making it a critical area of focus for the future of safety-critical systems engineering. Independent verification is an indispensable element in the development of safety-critical systems.

Despite the inherent complexities, the advantages realized through **increased safety, heightened reliability, and bolstered confidence** demonstrably surpass the investment required. By embracing the principles of independence and implementing the strategies outlined in this paper, organisations can significantly enhance the safety posture of their critical systems and minimise the potential for catastrophic failures.

Qualified tools are essential for effective and independent verification of safety-critical systems. They enhance objectivity, improve thoroughness, increase efficiency, facilitate compliance, and build confidence in the verification process. While tool qualification requires effort and investment, it is a crucial step in ensuring the safety and reliability of these critical systems.

LDRA tool suite® provides a robust and comprehensive solution for independent verification activities in safety-critical systems. LDRA tool suite® supports compliance with relevant industry standards and target required safety assurance levels. LDRA's capabilities encompass a wide range of essential verification functions, including static and dynamic source code analysis, comprehensive code coverage analysis, requirements traceability, streamlined test case design and execution, efficient test management and reporting, automated coding standards compliance checking, and accompanying tool qualification kits to facilitate easy tool qualification.

Utilising LDRA's industry-leading qualified verification tools—backed by over 50 years of expertise in developing solutions for safety-critical industries—enhances the objectivity, rigour, and efficiency of your verification process. This not only strengthens the safety and reliability of your critical systems but also ensures compliance with the required standards for independence in software verification activities.

While this paper primarily focuses on aerospace software systems requiring compliance with DO-178C, the principles and concepts discussed are broadly applicable to hardware design assurance for airborne electronic systems, particularly those containing FPGAs, PLDs, and ASICs governed by DO-254 and other safety-critical domains as well. Industries such as automotive, nuclear energy, and nuclear medicine also require stringent verification processes to ensure system reliability and compliance with regulatory standards. The need for independence in verification and validation, as emphasised in this discussion, is a fundamental aspect of safety assurance across these sectors. By adapting these methodologies, organisations can enhance the integrity, robustness, and regulatory compliance of safety-critical systems beyond aerospace applications.

About The Author

Shashi Kumar P, an Indian Air Force veteran, is a highly experienced aerospace professional with over 43 years of expertise in avionics systems development, verification, quality assurance, certification, and technical leadership. He led Product Assurance and Certification teams for high-visibility programs in Honeywell. A former Honeywell Senior Engineering Manager and their first Indian Software Certification Specialist (equivalent to an FAA company DER). He holds advanced degrees in Software Systems, Business Management, Public Administration and Human Resource Management. Additionally, he is an INCOSE ESEP and CSQP certified professional. He remains active in advancing aerospace safety and certification through his active participation in INCOSE and SAE-India chapters.

References

1. **RTCA-DO-178C** - *DO-178C is the primary guidance document for software development and certification in airborne systems*. It defines the guidelines for software safety, reliability, and compliance to ensure that avionics software meets the necessary safety requirements.
2. **RTCA DO-248C** - *Supporting Information for DO-178C and DO-278A*. DO-248C is a companion document to DO-178C (airborne software certification) and DO-278A (ground-based aviation software certification). It provides clarifications, FAQs, and rationale for the objectives and guidelines outlined in these standards.
3. **RTCA DO-330 -** *Software Tool Qualification Considerations*. RTCA DO-330 provides guidelines for the qualification of software tools used in the development, verification, and certification of airborne systems. It establishes a structured process for determining whether a tool can be trusted to produce correct, compliant, and reliable outputs without requiring manual verification.
4. **RTCA DO-331** – *Model-Based Development and Verification Supplement to DO-178C and DO-278A*. DO-331 is a technology supplement to DO-178C (airborne-software certification) and DO-278A (ground-based aviation software certification) that provides guidelines for using Model-Based Development (MBD) and Verification in safety-critical avionics system.
5. **RTCA DO-332** - *Object-Oriented Technology and Related Techniques Supplement to DO-178C and DO-278A*. DO-332 is a supplement to DO-178C (airborne software certification) and DO-278A (ground-based aviation software certification) that provides guidance on using Object-Oriented Technology (OOT) and Related Techniques in safety-critical avionics software.
6. **RTCA DO-333** - *Formal Methods Supplement to DO-178C and DO-278A*.
   DO-333 is a supplement to DO-178C (airborne software certification) and DO-278A (ground-based aviation software certification) that provides guidance on using Formal Methods for the development and verification of safety-critical avionics software.
7. **FAA Advisory Circular, AC 20-115D** - *Airborne Software Development Assurance*. AC 20-115D provides guidance on the use of RTCA DO-178C for the certification of airborne software. It establishes DO-178C as the primary means of compliance for software approval in aircraft systems.
8. **RTCA DO-254 -** Design Assurance Guidance for Airborne Electronic Hardware, providing crucial guidance for the development and certification of complex electronic hardware in airborne systems. This includes components like Field Programmable Gate Arrays (FPGAs), Programmable Logic Devices (PLDs), and Application-Specific Integrated Circuits (ASICs). It outlines a structured and rigorous process for hardware design, verification, and validation to ensure that these critical electronic components meet airworthiness requirements.

www.ldra.com

LDRA UK & Worldwide
Portside, Monks Ferry,
Wirral, CH41 5LH
Tel: +44 (0)151 649 9300
e-mail: info@ldra.com

LDRA Technology Inc.
2540 King Arthur Blvd, 3rd Floor, 12th Main Lewisville Texas 75056
Tel: +1 (855) 855 5372
e-mail: info@ldra.com

LDRA Technology Pvt. Ltd.
Unit B-3, Third floor Tower B, Golden Enclave
HAL Airport Road Bengaluru 560017
Tel: +91 80 4080 8707
e-mail: india@ldra.com