LDRA
Software Technology

Technical Briefing

DO-178C demystified:
Strategies for efficient certification

www.ldra.com

Background

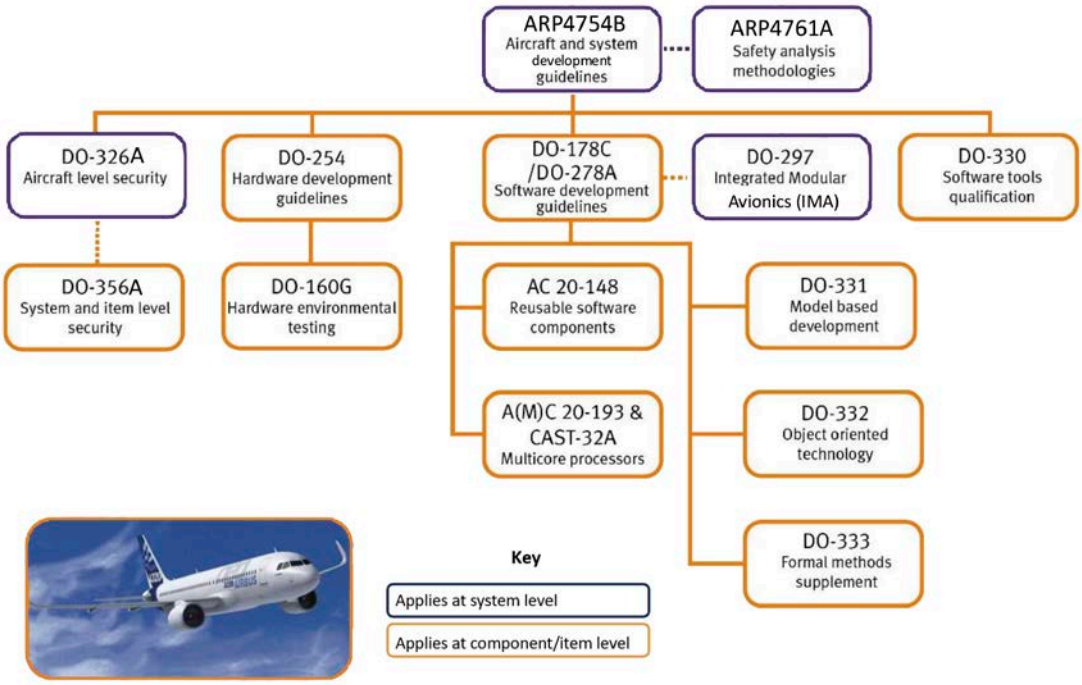
In response to the increased use of software in airborne systems, the Radio Technical Commission for Aeronautics organization [1] (now known as RTCA, Inc.) in collaboration with EUROCAE [2], published the 1982 guidance document *"Software Considerations in Airborne Systems and Equipment Certification"*.

The latest incarnation of the document was released in 2011. DO 178C/ED-12C [3] [4] was joined by a series of supplements and guidance documents (DO-330 [5], DO-331 [6], DO-332 [7], DO-333 [8]) that further extended its scope to cover newly emerged needs and technologies. LDRA has participated extensively on both the DO-178B and DO-178C committees over nearly two decades.

Since 2020, ongoing work around the development of DO-178C has focused on maintaining the safety and reliability of airborne software systems in the wake of further technological advances. Examples of resulting guidance include *"AC 20-193, Multi-core Processors"* [9] and *"AC 20-170A, Integrated Modular Avionics"* [10].

DO-178C in context

The illustration shows DO-178C and sister document DO-278A [11] in the context of the broader avionics development and certification framework.



ARP4754B [12] provides the overarching process for system development and certification. Its sister document, ARP4761A [13], describes methodologies for safety assessment.

These methodologies inform and integrate with the system development processes of ARP4754B, the software development guidelines of DO-178C, and the hardware development guidelines of DO-254/ED-80 [14] [15] & DO-160G/ED-14G [16] [17], whilst DO-297/ED-124 [18] [19] provides complementary development guidance for the implementation of IMA architectures.

As challenges emerge and technologies advance, so standards are introduced or adapted to accommodate them.

DO-178C applications

DO-178C is applied to a wide range of critical software applications on aircraft, ensuring their safety and reliability. These applications include Line-Replaceable Units (LRUs) – that is, modular components that can be quickly swapped out relatively easily. Examples include flight control systems, which manage the aircraft's stability and manoeuvrability; navigation systems, which provide essential data for route planning and positioning; communication systems, which facilitate vital exchanges between the aircraft and ground control; and surveillance systems, which monitor the aircraft's environment and detect potential hazards.

Supplementary guidance is integrated as necessary. For example, developers using Model-Based Development (MBD) to develop an ice protection system would call upon DO-331 to provide the guidance to apply DO-178C to their chosen development toolchain. Similarly, system developers leveraging multicore processors would look to A(M)C 20-193 to understand how their development life cycle should be adapted to suit.

DAL	Failure Condition
A	Catastrophic
B	Hazardous
C	Major
D	Minor
E	No effect

ARP4754B and DALs

ARP4754B requires that prior to system development, functional hazard analyses and system safety assessments are performed to determine the contribution of the system to potential failure conditions. The severity of failure conditions on the aircraft and its occupants are then used to determine a Design Assurance Level (DAL).

The ARP 4754B development process then allocates the associated DALs to the subsystems that implement the system's electronic hardware and software requirements. DO-178C establishes five "software levels" and modulates objectives that must be satisfied. This means that the effort and expense of producing a system is proportionate to the potential consequences of its failure.

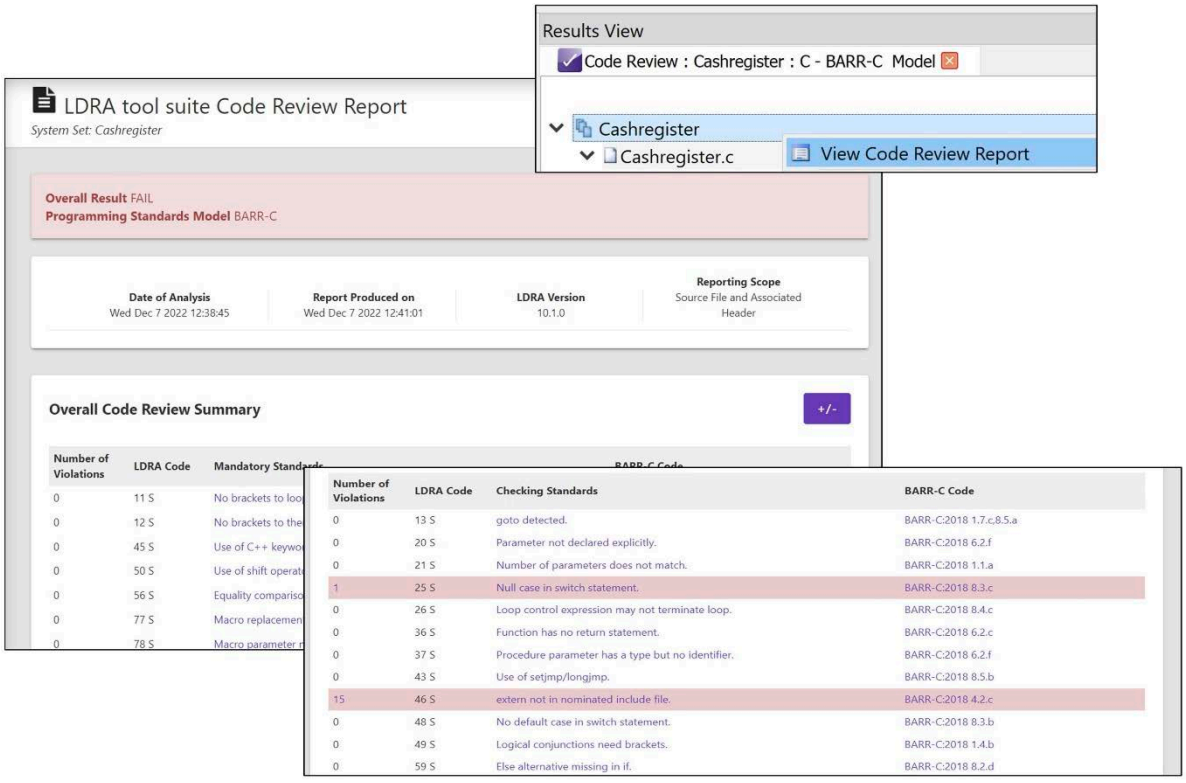
DO-178C recognizes that software safety must be addressed systematically throughout the software life cycle. This involves bi-directional life cycle traceability, software design, coding, validation and verification processes used to ensure correctness, control and confidence in the software. Several mechanisms are defined to help ensure that the processes are adhered to, and to provide evidence of that adherence.

DO-178C Process objectives

Collectively, the **DO-178C planning documents** provide a comprehensive framework for the development and certification of airborne software. They are intended to ensure that all activities are planned, controlled, and verified according to requirements.

There are also four **Stage of Involvement (SOI)** reviews specified by DO-178C. These are designed to ensure that software development processes and outputs adhere to the document's stringent requirements. FAA Designated Engineering Representatives (DERs) and EASA Subject Matter Experts (SMEs) police the DO-178C process particularly through their involvement with SOI audits. The SOI reviews provide structured checkpoints at various phases of the software development life cycle, ensuring a disciplined and methodical approach to achieving compliance with DO-178C.

The **Software Accomplishment Summary (SAS)** is a comprehensive document that encapsulates all the steps taken during the DO 178C software development and verification processes. It includes a recap of the software development activities, a summary of the verification results, and findings from the qualification of verification tools.



Evidential artifacts presented in a comprehensive, clear, and concise manner are a key part of any successful compliant project.

Compliance management

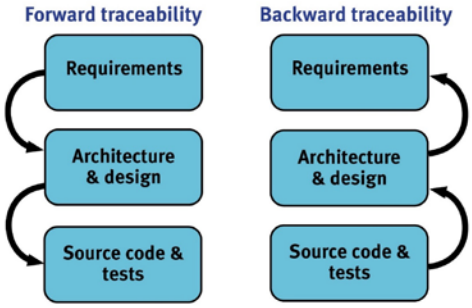
Project management problems are common to the development of many safety-critical software projects. Ensuring that all requirements are accurately captured, implemented, and linked to corresponding code and test cases can be challenging, leading to potential gaps and inconsistencies that complicate compliance efforts. Configuration management poses another challenge, as handling different versions of software artifacts and maintaining the integrity of configurations throughout the project life cycle is complex.

LDRA Certification Services (LCS) [20] offers the LDRA Compliance Management System (LCMS) [21] to address these challenges. LCMS comprehensive compliance documentation, process and document review tools, together with Level A FAA Designated Engineering Representative (DER) and EASA Subject Matter Expert (SME) support. It integrates with the LDRA Tool Suite to provide a comprehensive solution for developing and certifying safety-critical software.

DO-178C Software development and verification processes

The DO-178C software life cycle and the activities it specifies fit within the framework defined by the standard's life cycle processes.

Key elements of the DO-178C software life cycle include the practices analysis. Bi-directional traceability must be established across the life cycle.



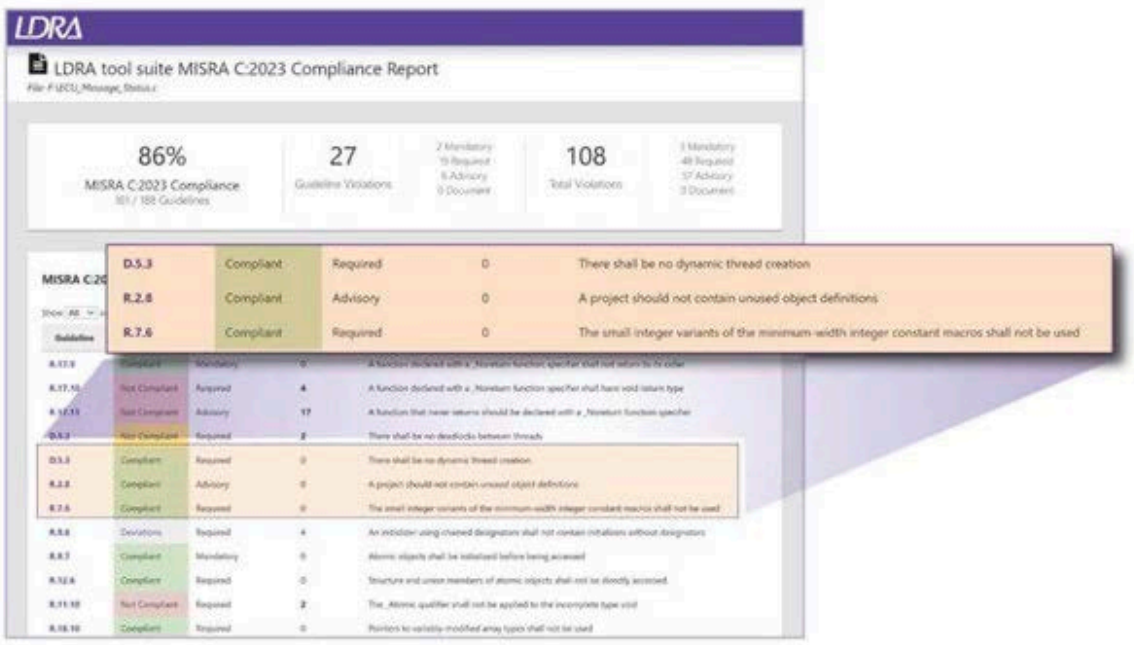
Structural coverage analysis (code coverage, data coupling and control coupling) quantifies the extent to which the source code of a system has been exercised by the testing process. Using these practices, it is possible to ensure that code has been implemented to address every system requirement and that the implemented code has been tested to completeness.

The use of software tools offers particularly significant benefits during software development and software verification as discussed in §5.0 and §6.0 of the standard, respectively.

DO-178C Section 5.0: SOFTWARE DEVELOPMENT PROCESSES

Five high-level processes are identified in the DO-178C §5.0 SOFTWARE DEVELOPMENT PROCESSES: Software Requirements Process (§5.1), Software Design Process (§5.2), Software Coding Process (§5.3), Integration Process (§5.4), and Software Development Process Traceability (§5.5).

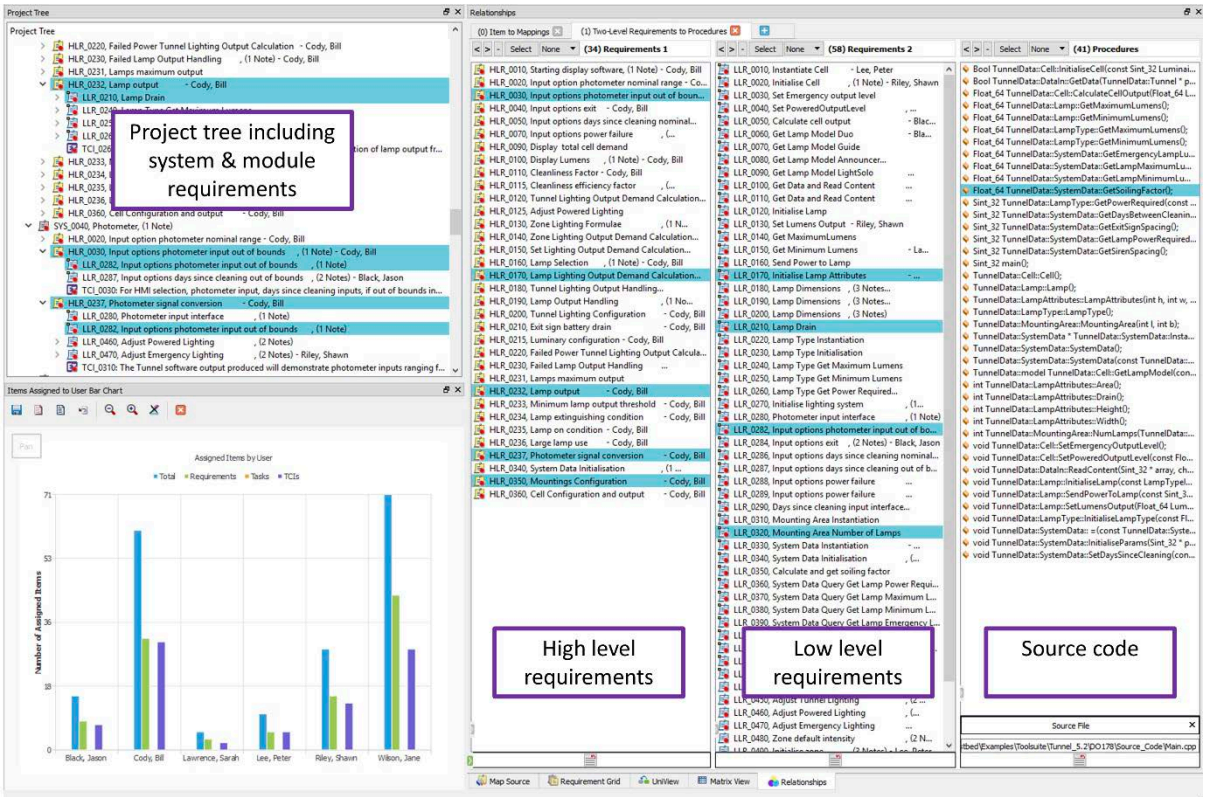
DO-178C §5.3 specifies software coding process objectives. For example, developers of source code should implement low-level requirements and conform to a set of software coding standards. LDRA static analysis tools provide code scanning – an automated "inspection" of the source code.



They make compliance checking easier, less error prone, and more cost effective by comparing the code under review with the rules dictated by the chosen a software coding standard.

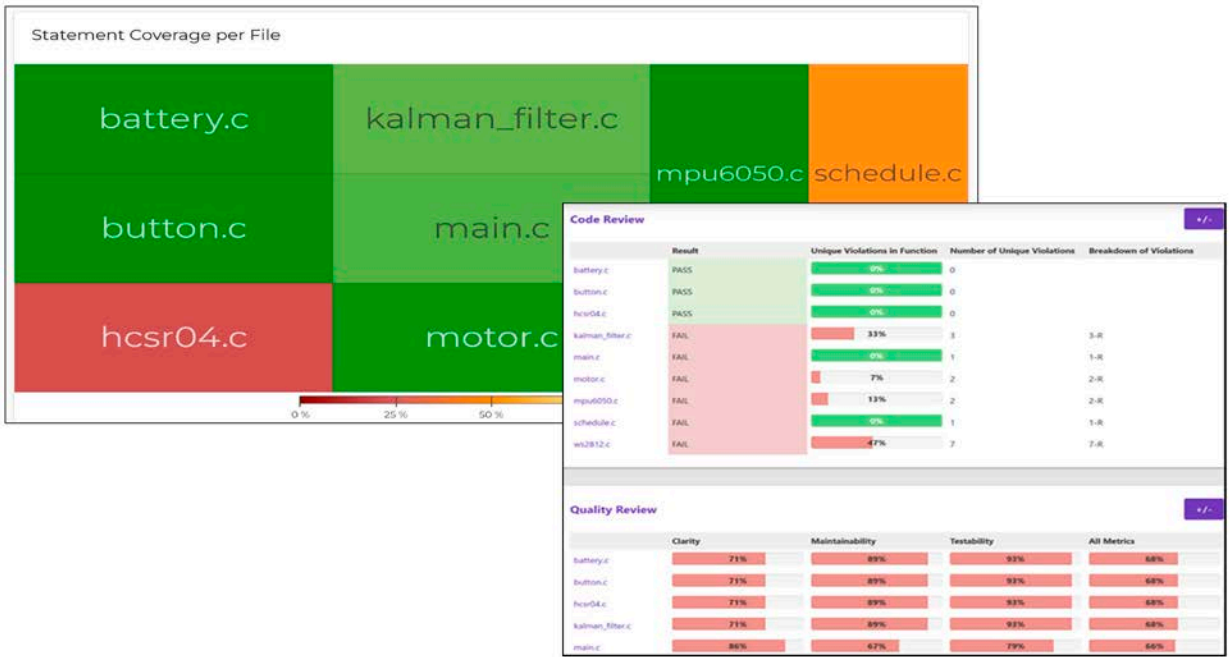
If everything follows the development life cycle in textbook fashion, that is perhaps a one-off, trivial task. The requirements will never change, and tests will never throw up a problem. But unhappily, that is rarely the case.

The TBmanager component of the LDRA tool suite is a desktop traceability application, and is integrated with code review, data and control coupling analysis, low-level testing, and code coverage tools.



It eases the project management challenges associated with such complexity by constantly maintaining a requirements traceability matrix down to the source code and test cases - even through disparate repositories.

LDRAvault is a web-based, enterprise level application that aggregates and manages certification artifacts across projects and programs providing transparency into the development and verification process.



Imported reports and results across multiple users and projects are collated as snapshots which form part of visualizations such as heat maps and trend graphs.

DO-178C Section 6.0: SOFTWARE VERIFICATION PROCESSES

Dynamic analysis involves using *"test cases and procedures"* (DO-178C §6.0) to exercise Executable Object Code (EOC), on a target representative of that to be deployed in the field. It provides evidence of both correct functionality and of the parts of the code exercised to achieve it ("structural coverage"). The *"test cases and procedures"* can include any combination of low-level tests (sometimes referred to as unit tests), integration tests, and system tests.

Low-level tests verify the complete and exclusive implementation of the low-level requirements specified in the Software Verification Plan (SVP), whereas software integration testing verifies the relationships between software components with reference to the requirements and the software architecture. In practice, the mechanisms used for low-level testing often lend themselves to integration testing and hence verify behaviour in the context of a call tree.

As previously mentioned, keeping track of a project in flux can be challenging. Again, the TBmanager component of the LDRA tool suite automates the maintenance of the bi-directional relationship between the products of the different development phases while saving a great deal of time and helping eliminate errors – not just as far the development of the requirements and source code, but through to requirements-based testing and test coverage for both high- and low-level requirements.

Traceability Matrix

Child \ Parent	HLR_0090	HLR_0231	HLR_0125	HLR_0340	HLR_0110	HLR_0120	HLR_0220	HLR_0030	HLR_0020	HLR_0130	HLR_0180	HLR_0350	HLR_0233	HLR_0100	HLR_0140	HLR_0190
TCI_0050																
TCI_0020									X							
TCI_0250		X														
TCI_0260																
TCI_0270													X			
TCI_0280																
TCI_0290																
TCI_0300																
TCI_0310																
TCI_0320				X												
TCI_0340												X				
TCI_0330																
TCI_0345																

Whenever changes are made, regression tests can be re-applied as development progresses to ensure that existing functionality is not compromised by new development.

At the enterprise level, LDRAvault is as applicable to verification as it is to development. Once again, visualizations like heat maps and trend graphs are leveraged to illustrate the progress of the verification tasks associated with the project.

Structural coverage analysis objectives

Structural coverage identifies which code structures and component interfaces have been exercised during the execution of requirements-based test procedures. Structural coverage analysis determines if there are any parts of the code which have not been sufficiently exercised, and if not, why.

LDRA tool suite Test Case Coverage Overview Report

System Set: Cpp_tunnel_lighting_system

Sequence Cell
Overall Result PASS

Date of Analysis	Report Produced on	LDRA Version
Mon Jan 20 2020 10:39:03	Mon Jan 20 2020 10:41:09	10.0.0

Test Case Regression Summary Table

Test Case	Procedure	File	Inputs	Outputs	Status	Statement (%)	Branch/Decision (%)	MC/DC (%)
1	TunnelData::Cell::Cell	Cell.cpp	0	44	PASS	100%	100%	na
2	TunnelData::Cell::InitialiseCell	Cell.cpp	58	21	PASS	72%	68%	na
3	TunnelData::Cell::SetEmergencyOutputLevel	Cell.cpp	1	0	PASS	72%	67%	0%
4	TunnelData::Cell::SetPoweredOutputLevel	Cell.cpp	9	1	PASS	80%	69%	0%
5	TunnelData::Cell::CalculateCellOutput	Cell.cpp	2	1	PASS	100%	na	na
6	TunnelData::Cell::GetLampModel	Cell.cpp	1	0	PASS	68%	36%	0%

DO-178C §6.4.4 details requirements for the achievement of 100% MC/DC, decision, and statement coverage, depending on DAL. Collation of structural coverage metrics is typically achieved by "instrumenting" a copy of the source code (that is, superimposing function calls to collate coverage data) and executing that instrumented code using requirement-based test cases.

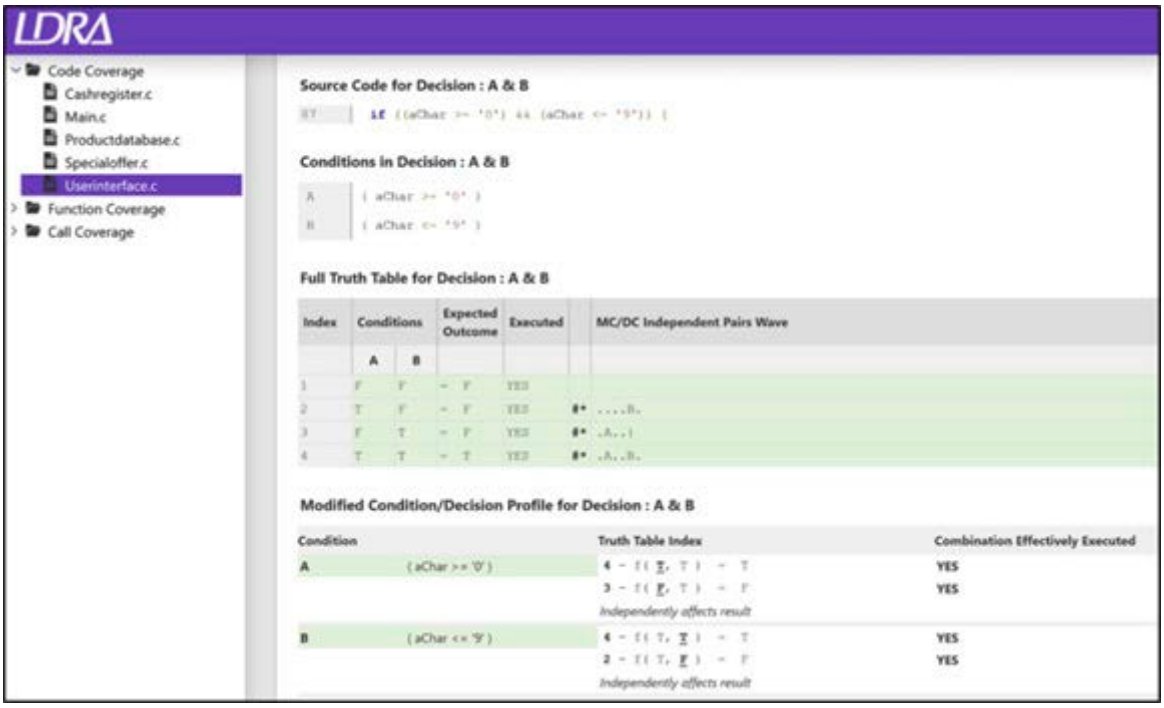
SCA is then applied to assess the effectiveness of this testing by measuring how much of the code has been exercised. An iterative "review, analyse, verify" cycle is typically needed to ensure that the software coverage objectives are fulfilled. Graphical representation can be a great help.

System requirements can be shown to have been correctly decomposed, implemented, and verified by combining a complete trace from requirements through to code and test cases, with the achievement of comprehensive functional test coverage and structural coverage objectives.

Modified Condition / Decision Coverage (MC/DC)

In addition to statement and branch coverage, for level A software MC/DC is obligatory. MC/DC requires that *"Each condition in a decision has been shown to independently affect that decision's outcome"* (DO-178C Glossary).

MC/DC is a coverage metric for multiple condition decisions. It does not require every possible combination to be executed but instead requires only one more test than the number of conditions involved. For example, with 6 conditions there are 64 possible combinations – and yet only 7 tests are needed to achieve MC/DC coverage.



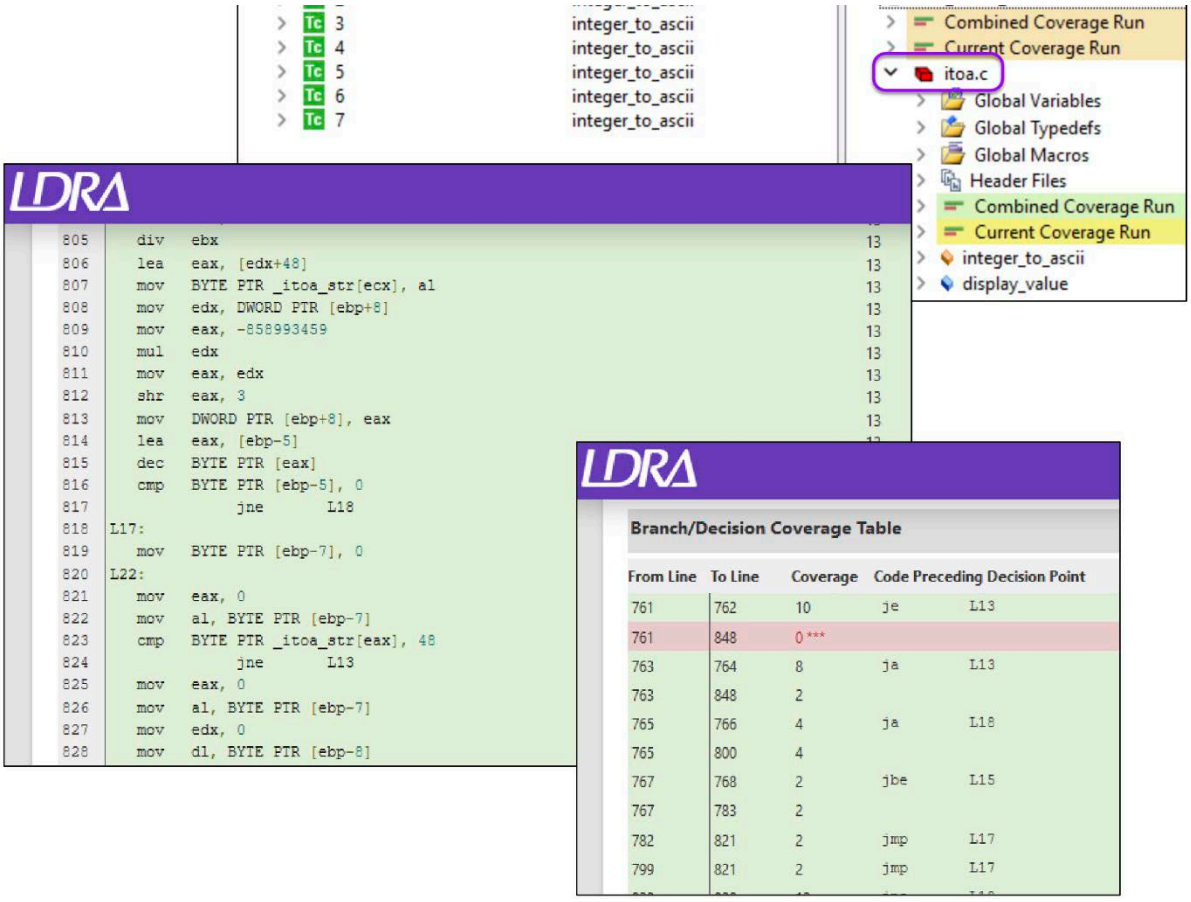
However, those tests must show that each of the conditions independently affect the result, and it is not always obvious which input values might achieve that. The use of an MC/DC planner makes the selection of appropriate values much easier.

Source to object code traceability

For Level A systems, structural coverage at the source level isn't enough. Compilers often add additional code or alter control flow, and often their behaviour is not deterministic. To ensure that functionality is not compromised, DO-178C §6.4.4.2.b states that:

"...if the software level is A and a compiler, linker, or other means generates additional code that is not directly traceable to Source Code statements, then additional verification should be performed to establish the correctness of such generated code sequences."

Because there is a direct one-to-one relationship between object code and assembly code, one way for a tool to provide evidence of that verification is to display both a graphical representation of the source code and an equivalent representation of the assembly code.



Object Code Verification (OCV) measures code coverage at both the source and the assembly level by instrumenting each in turn.

Data Coupling and Control Coupling

DO-178C §6.4.4.2.c requires *"analysis to confirm that the requirements-based testing has exercised the data and control coupling between code components."*

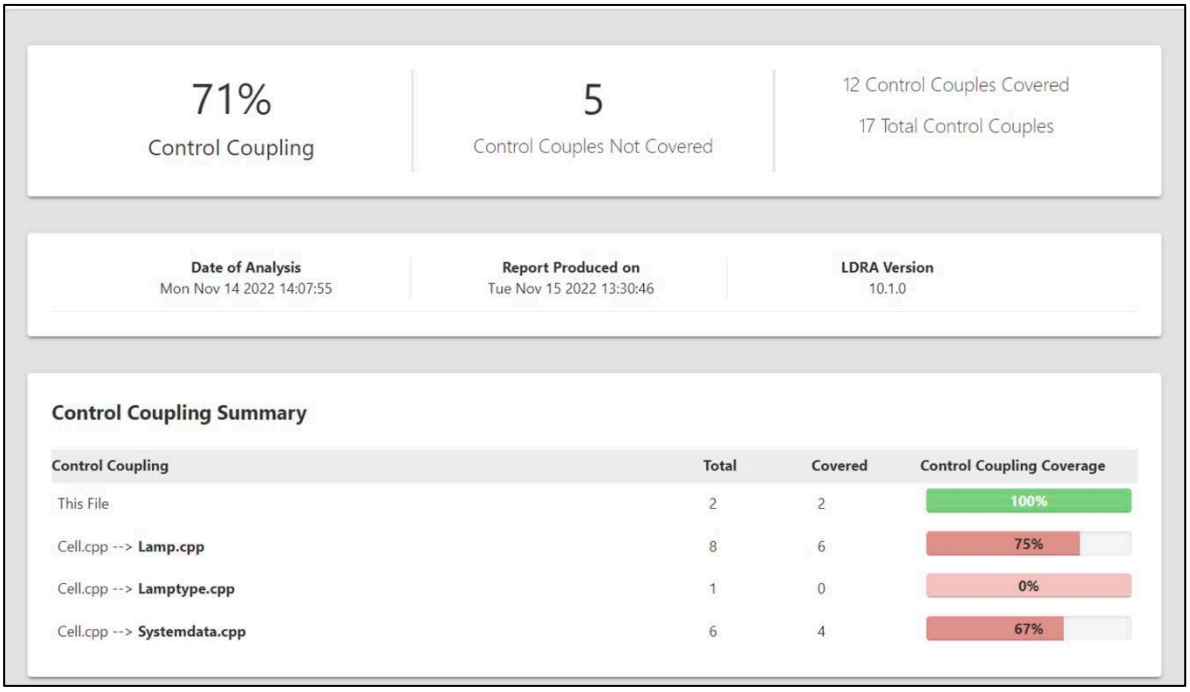
Data coupling and control coupling analysis therefore need to be performed post execution, and the generated artifacts reviewed against the system requirements and architecture.

The analysis of control and data coupling by traditional means can be tedious and challenging. LDRA's patented approach leverages static analysis and dynamic analysis in tandem to make that process more efficient and less error prone.

Control Coupling

Control Coupling is defined in the glossary of DO-178C as *"The manner or degree by which one software component influences the execution of another software component."*

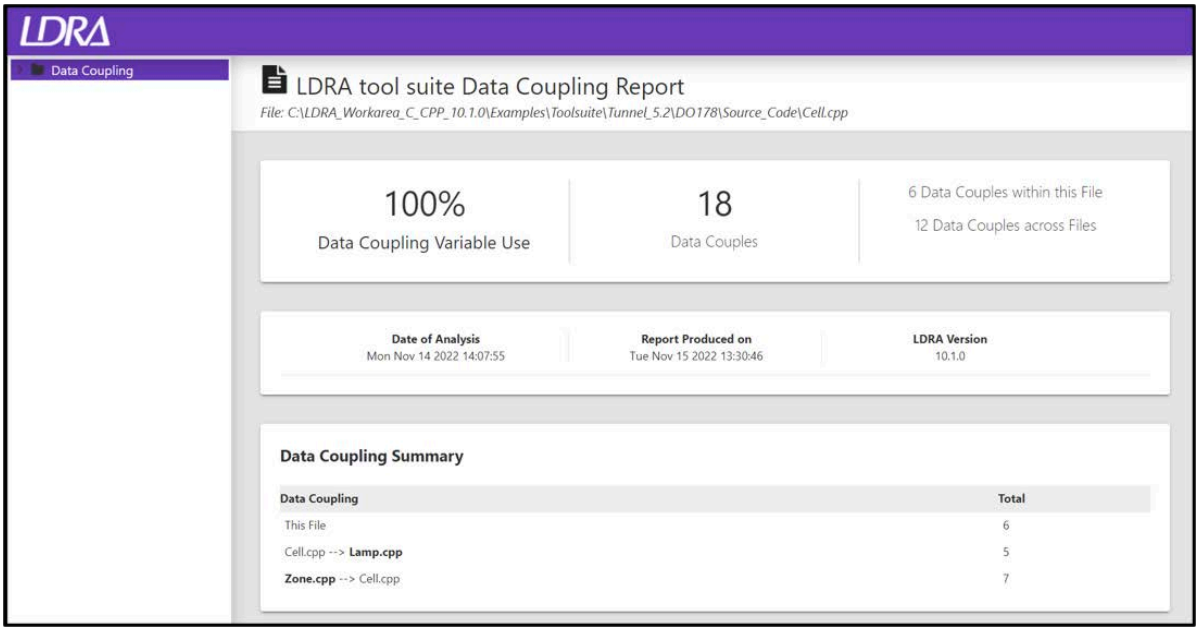
The reports generated by the LDRA tool suite identify the number and location of intra-file, file-to-file, and module-to-module calls.



They identify any gaps and guide targeted verification activities, and ultimately provide evidence that all couples have been validated.

Data Coupling

Data coupling is defined in the glossary of DO-178C to be *"The dependence of a software component on data not exclusively under the control of that software component"*. DO-178C §6.4.4.2.c requires *"Analysis to confirm that the requirements-based testing has exercised the data and control coupling between components"*.



Data coupling analysis is focused on the observation and analysis of data elements as they are set and used ("set/use pairs") across software component boundaries.

Execution time

DO-178C requires that software execution times be predictable and consistent with the system's requirements, referencing a need for execution time analysis throughout §6.4.

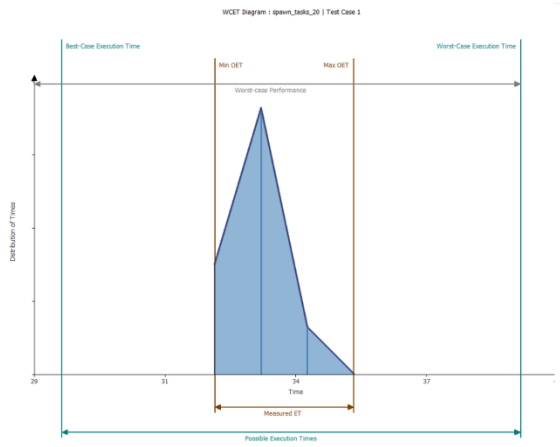
The objective is to establish an upper execution time bound for each task (called Worst Case Execution Time, or WCET). The WCET will depend on both the target hardware and the software.

It is increasingly common practice to deploy multicore processors (MCPs) in avionics applications. These MCP devices almost always share hardware resources outside the processor cores, and time-related delays occur as users wait for access to these Hardware Shared Resources (HSRs).

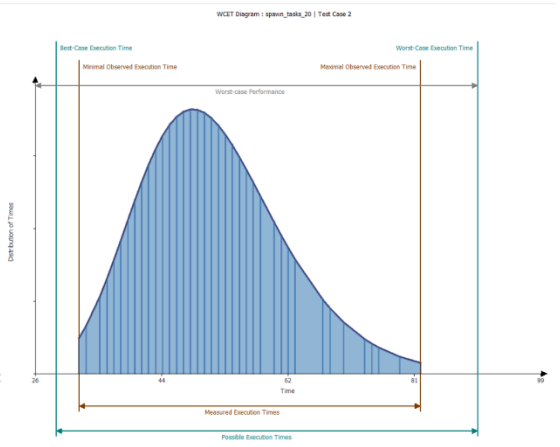
AC 20-193 and AMC 20-193

FAA's AC 20-193 document and EASA's AMC 20-193 equivalent were written *"to identify topics that could impact the safety, performance and integrity of a software airborne system executing on Multi-Core Processors (MCP)"*. They are known collectively as A(M)C 20-193.

The LDRA tool suite facilitates a practical, A(M)C 20-193 compliant approach to the optimization of system configuration using interference research. It leverages execution time measurement using the TBrun component of the LDRA tool suite supported by the optional TBwcet module [22].



TBwcet timing diagram
Without interference



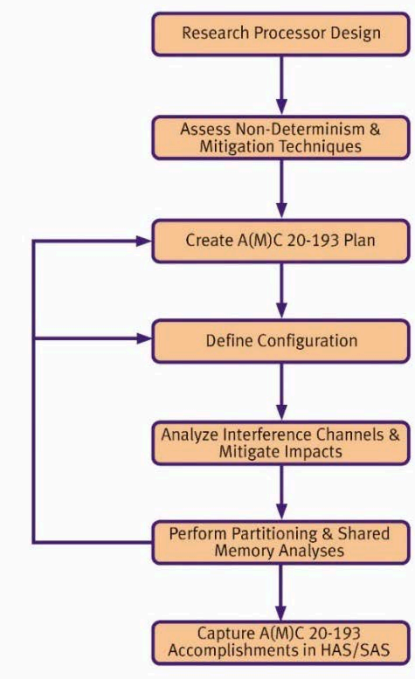
TBwcet timing diagram
With interference

There are many highly effective robust partitioning mechanisms available. These mitigate for many of the most significant interference channels – but not all. The onus remains on the developer to demonstrate that interference mitigation is effective which can only be achieved through measurement.

LDRA's solution adopts a "wrapper" principle which affords the flexibility analysis from complete system behaviour, through a thread or process, right down to class/function/procedure level. This approach provides the ability to both "drill in" to problem areas, and to analyse the complete system.

Stressors consisting of specially designed software are essential in MCP applications for use in the derivation of WCET by simulating worst-case interference scenarios. LDRA leaves the choice of HSR stressor mechanism to the user. Open source solutions such as Stress ng [23] provide custom stressors for HSRs.

The resulting iterative interference research process [24] makes flexible and responsive project management tools an imperative The TBmanager component of the LDRA tool suite would be an appropriate choice.



Supplements to DO-178C

When DO-178C superseded DO-178B, four supplements were created to address the need for clarity and guidance on modern software development practices within the context of aviation software certification. They add, delete, or modify objectives, activities, and life cycle data in DO-178C to cater to the specific needs of these technologies.

RTCA identifier	EUROCAE identifier	Title
DO-330	ED-215	Software Tool Qualification Considerations
DO-331	ED-216	Model-Based Development and Verification Supplement to DO 178 and DO-278A
DO-332	ED-217	Object-Oriented Technology and Related Techniques Supplement to DO-178 and DO-278A
DO-333	ED-218	Formal Methods Supplement to DO-178 and DO-278A

DO-330 Software Tool Qualification Considerations

It is essential to ensure that any tools deployed can be relied upon within the project specific operational context. The LDRA tool suite is designed to be used for verification purposes, and so it output is not used as part of the airborne software. Such a tool is always assigned Tool Qualification Level 5.

Certification authorities undertake tool qualification on a project by project basis, so the responsibility for showing the suitability of any tools falls on to the organisation developing the application. However, they can leverage Tool Qualification Support Packages (TQSP). LDRA TQSPs contain a series of documents, including tool operation requirements that identify the development process needs satisfied by the tool, and test cases to demonstrate that the tool is operating to specification in the verification environment.

DO-331 Model-Based Development and Verification Supplement

Both Model-Based Systems Engineering (MBSE) and Model-Based Development (MBD) are addressed by DO-331. MBSE guidance emphasizes the overall system's requirements and interactions, while MBD recommendations concentrate on the software's design and testing processes. DO-331 takes the approach that specification models or design models take the place of high-level and low-level requirements respectively.

DO-331 §MB.6.0 (Software verification process) expands on how best practice applies to MBD, with DO-331 §MB.6.8.2 (Model Simulation for Verification of Executable Object Code) expanding upon which verification objectives can be partially satisfied at the model level, and which must be performed at the target level.

Partial credit in the model

DO-331 §MB.6.8.2 describes objectives applicable to the compliance of EOC with high-level and low-level requirements, test coverage of software structure, and data coupling and control coupling. Additional verification activities must be performed on the target hardware to fully satisfy these objectives. When certification credit is sought from model simulation to partially satisfy software testing objectives and test coverage regarding high-level requirements, the same design model is used for code generation and to produce the EOC.

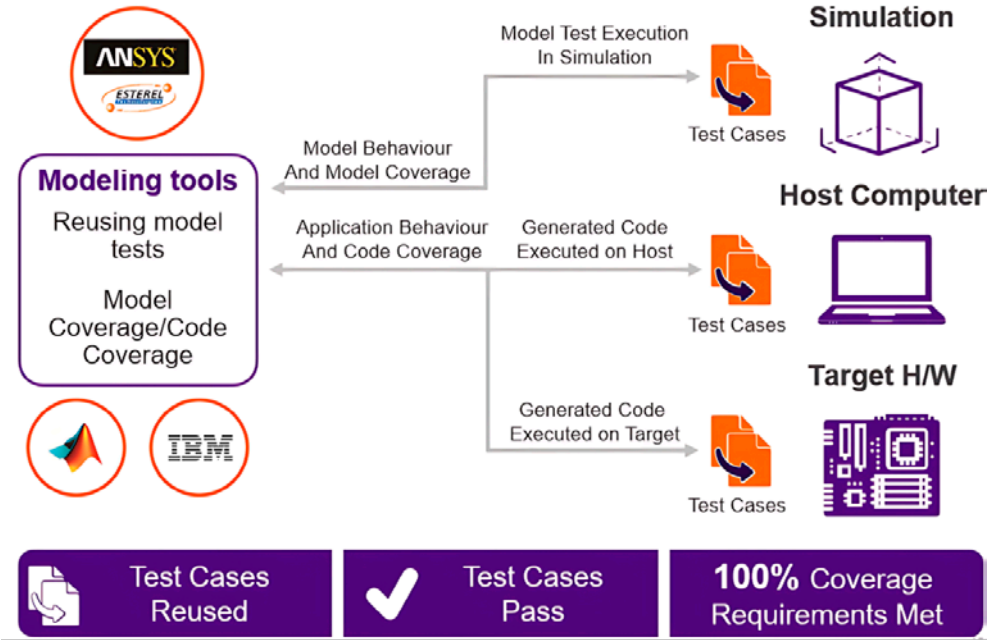
The document also discusses the need for plans to identify test and test coverage activities that will be satisfied at the model level, and those are to be exercised on the target.

Verification on target

DO-331 §MB.6.8.2 goes on to suggest that several *"specific tests should still be performed in the target environment"*. It identifies the various forms of verification objectives that can only be met on the target. It also lists various types of errors that can and cannot be revealed at the simulation level and can only be detected on the target hardware.

Finally, DO-331 §MB.B.11 (FAQ #11) addresses model coverage activity, suggesting that model coverage analysis cannot usually take the place of structural coverage analysis as per DO-178C §6.4.4.2.

The integration of test and modelling tools help to achieve that seamlessly, including the static analysis of generated code, the collection of code coverage from model execution, and the migration of model tests into an appropriate form for execution on the target hardware.



DO-332 Object-Oriented Technology and Related Techniques Supplement

DO-332 describes concepts and key features of object-oriented technologies and related techniques, discusses their impact on the planning, development, and verification processes, and enumerates their vulnerabilities.

OO Objectives

Two pertinent objectives are included in the DO-332 supplement:

- **§OO.6.7.1 Verify local type consistency.** Ensuring that that "each class passes all the tests of all its parent types which the class can replace" by means of Liskov Substitution Principle tests is usually the most practical approach here.
- **§OO.6.8.1 Verify the use of dynamic memory management is robust:** A range of static and dynamic analysis techniques can be deployed to fulfil DO-332 A-7 OO.6.8.1 and the related vulnerabilities outlined in Annex OO.D.1.6.1.

Tracking memory allocation and deallocation helps to ensure the proper freeing of memory, as do associated checks prior to dereferencing. Low-level testing provides a mechanism to explore various allocation/deallocation scenarios to help ensure that vulnerabilities are addressed. Timing hooks within low-level tests help characterize allocation/deallocation timing, and dynamic data flow analysis monitors data elements in runtime to detect lost updates and stale references.

DO-333 Formal Methods Supplement to DO-178 and DO-278A

DO-333 provides guidance on the use of formal methods. Formal methods are mathematically based techniques for the specification, development, and verification of software and hardware systems. DO-333 integrates these methods into the existing framework of DO-178C to provide an alternative to the more established methods of providing a rigorous and reliable certification process.

The document outlines various formal methods tools, such as theorem provers, model checkers, and abstract interpretation tools, and provides criteria for their qualification. It details how these formal methods can be leveraged to verify complex systems, complementing the traditional approaches established by DO-178C. Where formal methods are leveraged as part of a safety case, DO-333 helps ensure that safety-critical systems meet the usual stringent reliability and performance standards.

Conclusions

The use of traceability, test management and static/dynamic analysis tools for an airborne software project that meets the DO-178C certification requirements offers significant productivity and cost benefits. Tools generally make compliance checking easier, less error prone and more cost effective. In addition, they make the creation, management, maintenance and documentation of requirements traceability straightforward and cost effective.

In particular, the provision of an automated, comprehensive software tool chain offering complete 'end-to-end' traceability across the development life cycle, encompassing requirements, code, on-target tests, artifacts, and objectives with both static and dynamic analysis capabilities is invaluable to developers and project managers alike.

Development in compliance with DO-178C will never be an easy task. However, the right tools can be very helpful in making such work as easy as it can possibly be.

Works cited

[1] RTCA, "RTCA," [Online]. Available: http://www.rtca.org/ . [Accessed 14th June 2024].

[2] EUROCAE, "EUROCAE," Userfull, 2024. [Online]. Available: https://eurocae.net/. [Accessed 14th June 2024].

[3] RTCA, Inc., RTCA DO-178C - Software Considerations in Airborne Systems and Equipment Certification, RTCA, Inc., 2011.

[4] European Organization for Civil Aviation Equipment (EUROCAE), ED-12C - Software Considerations in

Airborne Systems and Equipment Certification, Paris, France: EUROCAE, 2011.
[5] RTCA, Inc., RTCA DO-330 - Software Tool Qualification Considerations, Washington DC: RTCA, Inc., 2011.
[6] RTCA, Inc., RTCA DO-331 - Model-Based Development and Verification Supplement to DO-178C and DO-278A, Washington DC: RTCA, 2011.
[7] RTCA, Inc., RTCA DO-332 - Object-Oriented Technology and Related Techniques Supplement to DO-178C and DO-278A, Washington DC: RTCA, 2011.
[8] RTCA, Inc., RTCA DO-333 - Formal Methods Supplement to DO-178C and DO-278A, Washington DC: RTCA, Inc., 2011.
[9] Federal Aviation Administration (FAA), Advisory Circular 20-193: Use of Multi-Core Processors., Washington, DC.: FAA, 2024.
[10] Federal Aviation Administration (FAA), "AC 20-193 - Use of Multi-Core Processors," U.S. Derpartment of Transportation, 8th January 2024. [Online]. Available: https://www.faa.gov/regulations_policies/advisory_circulars/index.cfm/go/document.information/documentID/1036408. [Accessed 5th March 2024].
[11] RTCA, Inc., RTCA DO-278A - Software Integrity Assurance Considerations for Communication, Navigation, Surveillance and Air Traffic Management (CNS/ATM) Systems, Washington DC: RTCA, Inc., 2011.
[12] SAE International, "ARP4754B - Guidelines for Development of Civil Aircraft and Systems," 20th December 2023. [Online]. Available: https://www.sae.org/standards/content/arp4754b/. [Accessed 14th June 2024].
[13] SAE International, "ARP4761A - Guidelines for Conducting the Safety Assessment Process on Civil Aircraft, Systems, and Equipment," 20th December 2023. [Online]. Available: https://www.sae.org/standards/content/arp4761a/. [Accessed 14th June 2024].
[14] RTCA, Inc., DO-254 - Design Assurance Guidance for Airborne Electronic Hardware, Washington, DC: RTCA, Inc., 2008.
[15] European Organization for Civil Aviation Equipment (EUROCAE), ED-80 - Design Assurance Guidance for Airborne Electronic Hardware, Paris, France: EUROCAE, 2000.
[16] RTCA, Inc., DO-160G - Environmental conditions and test procedures for airborne equipment, Washington, DC: RTCA, Inc., 2010.
[17] European Organization for Civil Aviation Equipment (EUROCAE), ED-14G - Environmental Conditions and Test Procedures for Airborne Equipment, Paris, France: EUROCAE, 2010.
[18] RTCA, Inc., "DO-297 - Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations," 8 November 2005. [Online]. Available: https://my.rtca.org/NC__Product?id=a1B36000001IchGEAS. [Accessed 6 January 2022].
[19] European Organization for Civil Aviation Equipment (EUROCAE), ED-124 - Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations, Paris, France: EUROCAE, 2010.
[20] LDRA, "LDRA Certification Services (LCS)," [Online]. Available: https://ldra.com/certification-services/. [Accessed 25th June 2024].
[21] LDRA, "Data Sheet - LDRA Compliance Management System (LCMS)," May 2024. [Online]. Available: https://ldra.com/documentviewer.php?file=LCMS-Data-Sheet-v2.1.pdf. [Accessed 25th June 2024].
[22] LDRA, "TBwcet®," [Online]. Available: https://ldra.com/products/tbwcet/. [Accessed 3rd February 2023].
[23] C. King, "Ubuntu wiki: stress-ng,," 7th October 3030. [Online]. Available: https://wiki.ubuntu.com/Kernel/Reference/stress-ng. [Accessed 3rd February 2023].
[24] D. Iorga, T. Sorensen, J. Wickerson and A. F. Donaldson, "Slow and Steady: Measuring and Tuning Multicore Interference," in IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS), Syndey, Australia, 2020.

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