DO-178B software test optimization

DO-178B is the international safety standard used to certify high-reliability commercial avionics systems software. It methodically outlines the requirement for obtaining FAA software approval. The standard looks into the design, development, verification and validation, and configuration management for software development. Testing critical software for DO-178B's design assurance level (DAL) A, B or C is a fairly intense process. If the approach is not defined upfront with all due consideration, there is a significant impact on the overall effort and cost to qualifying the software.

Defining a well-drafted test approach ensures success in obtaining certification and determines how efficiently the testing will proceed saving time and money. An experienced verification team is required to smartly craft the way through the verification process. A bad approach to qualifying the software often leads to the project spinning out of control. There are numerous cases where verification effort for DO-178B has taken seven to 10 times the original effort estimates. All due to an incorrect test approach.

Testing is simplified when organizations map DO-178B procedures with automatic code generation tools. Model-based development and verification using tools like SCADE significantly bring down overall costs. SCADE helps meet DO-178 objectives by establishing, optimizing, and rolling out a comprehensive and efficient testing strategy. It also provides options for software verification at the requirements level with modified condition/decision coverage (MCDC) report.

For better modeling, SCADE expects requirements to be clearly defined in terms of output affected by inputs in real-time. This helps in verifying requirements functionally instead of just structural coverage. Since 70% to 90% of software can be modeled at a high-level, SCADE uncovers functional errors and gaps at an earlier stage.

The remaining requirements are generally a much smaller percent and can get covered with manual coding. With majority of functional verification happening earlier in the process and at a high-level, the effort involved in software verification is much reduced. This approach reduces the overall risk significantly.

For certification, software requirements should be exhaustively tested. This was traditionally done using low-level testing tools like RTRT, CANTATA++, LDRA or VectorCAST. They provide coverage analysis of test cases including MCDC at the component level. Even though these are flexible at low-level, statistics show most issues are at a higher level, in how the components interact with each other. Mature verification teams focus their testing effort at the high-level to uncover functional errors as much as possible.

A good metric is often 70% to 90% effort in high-level verification and remaining in low-level verification. If complete coverage is attempted at low-level, it can consume up to 50% of time and budget.

Blueprint for action

Avionics companies attempt to achieve 100% coverage through a combination of functional testing and analysis. The following are the best practices one can apply:

There is no one size fits all coverage analysis metric

- Focus on testing at the functional level is based on the requirements collected.
- Execute requirements-based functional testing first, i.e. before any structural coverage analysis metrics are gathered.
- Requirements based test cases do not guarantee testing of the entire code layout. Complete analysis is recommended for additional verification.
- At times requirements based testing may be incomplete because of shortcomings in the requirements, if so identified the requirements based test should be are enhanced to address the omission.
- Promote coverage analysis at the debugging stage so developers uncover the maximum numbers of bugs, allowing testers to focus on identifying the more esoteric defects.
- For code not covered because it is prohibitively expensive to create the system faults and conditions causing the code to be executed, use additional analysis methods like code walkthroughs to verify defects.

The coverage objectives for each system must be chosen based on the system's role. For some full coverage is essential while for other systems that do not damage the environment of the plane, like an in-flight entertainment system, full coverage analysis is not required.