Chapter 14
Component-Based Development of Aeronautical Software

Ítalo Romani de Oliveira
Atech Tecnologias Críticas, Brazil

Ricardo Alexandre Veiga Gimenes
Safety Analysis Group (GAS), University of São Paulo (Poli-USP), Brazil

Jorge Rady de Almeida Jr.
Safety Analysis Group (GAS), University of São Paulo (Poli-USP), Brazil

ABSTRACT
Modern aircraft heavily relies on software to fly and operate, which lessens pilot workload, increases flight stability and fuel efficiency, and provides several other benefits. However, the more automated an aircraft is, the more prone to complexity its software modules are, raising special safety issues to be considered in the project. This chapter presents an overview of the Verification and Validation requirements for safety-critical software in aeronautics and, given the high costs to meet them, explains in detail a component-based methodology which can contribute to reduce the overall costs of software development and, at the same time, provide enhanced safety.

INTRODUCTION
Modern Air Transportation demands a great amount of software, with a wide variety of purposes, from airline ticket reservation to aircraft engine control. Higher levels of system integration on transport aircraft systems and equipment have created new safety and certification challenges, making the aviation industry and the regulatory authorities elaborate technical guidance, gathering their mutual objectives and responsibilities in the system safety area. A great part of the software systems in aviation is considered safety-critical, as is the case of many aircraft control systems, in which failures can lead to catastrophic events, as happened to the Airbus A320 in Habsheim, France, 1988, just to mention a classical example. This accident was attributed to a software design error related to the complexity introduced by the once innovative computerized flight control. The failure showed the crew that the
plane was flying at 100 ft above ground. When the captain increased throttle to level off at 100 ft, the engines and the completely computerized throttle control did not respond. The gear was only 30 ft above ground when the aircraft was passing over the runway (Walters, 2000), and a few seconds later the aircraft crashed into the bush. Due to the high social impact related to flight safety, there is a paramount need of requiring strict verification processes and tests on critical aeronautical software, defined and supervised by governmental authorities.

Software Verification and Validation (V&V)

The way to improve the verification and validation of software is still a theme of several scientific studies. However, there are recommendations and methodologies in which many researches converge, helping to improve the software reliability.

The objectives of the software verification activities are to demonstrate the following properties (Howden, 1986; Powell, 1986):

a. Correctness: the extent to which the product is fault free;

b. Consistency: the extent to which the product is consistent within itself and with other products;

c. Necessity: the extent to which everything in the product is necessary;

d. Sufficiency: the extent to which the product is complete;

e. Performance: the extent to which the product satisfies its performance requirements.

Verification activities come together with validation activities, which aim at ensuring that the project deliverables are in accordance with the expectations of the project stakeholders. Since verification and validation activities complement each other and are often overlapped, in software engineering they are treated in the same bucket called V&V.

Types of Software Verification

Software verification can be performed by testing and by formal techniques. Testing consists in defining a set of conditions for the software inputs, and requiring the outputs to provide the expected results in each case. It cannot demonstrate absolute absence of errors, except for the cases tested. Frequently, however, the algorithmic complexity and the dependency on hardware events turn exhaustive testing very hard, and formal techniques must be applied. These formal techniques consist in automatically exploring all the software execution paths, in which case it is called model-checking, or using rules of propositional calculus to automatically prove that some required property is valid, given the system axioms, in which case the technique is called theorem proving.

The possibility of performing automatic verification sounds very interesting; however, in practice, model-checking techniques suffer from combinatorial explosion, and automatic theorem proving requires extremely disciplined and time-consuming logical creativity; besides, both formal techniques are incompatible with the continuum hypothesis (i.e., to work with continuous number intervals in the real domain), which is an essential support for elaborating closed-loop control systems. Nevertheless, the perception in the Computer Science community is that automatic formal verification can be very helpful if carefully used. The use of formal verification techniques can help to increase the verification effectiveness and, sometimes, dispense testing.

Verification and Validation of Aeronautical Software

Critical software for civil aviation can be categorized in two major groups: Avionics and Air Traffic Management (ATM). The former one is embedded
Related Content

Applying the Certification’s Standards to the Simulation Study Steps
[www.igi-global.com/chapter/applying-certification-standards-simulation-study/55982?camid=4v1a](www.igi-global.com/chapter/applying-certification-standards-simulation-study/55982?camid=4v1a)

Unmanned Aerial Vehicle Applications for Military GIS Task Solutions
[www.igi-global.com/chapter/unmanned-aerial-vehicle-applications-for-military-gis-task-solutions/223732?camid=4v1a](www.igi-global.com/chapter/unmanned-aerial-vehicle-applications-for-military-gis-task-solutions/223732?camid=4v1a)

Commercial Aircraft: A Holistic and Integrated Model of the Flux of Information Regarding the Operational Support
José Lourenço da Saúde and Jorge Miguel Reis Silva (2010). *Computational Models, Software Engineering, and Advanced Technologies in Air Transportation: Next Generation Applications* (pp. 146-175).
[www.igi-global.com/chapter/commercial-aircraft-holistic-integrated-model/38107?camid=4v1a](www.igi-global.com/chapter/commercial-aircraft-holistic-integrated-model/38107?camid=4v1a)

Managing Complex Technology Innovation: The Need to Move Beyond Stages and Gates
[www.igi-global.com/article/managing-complex-technology-innovation/69383?camid=4v1a](www.igi-global.com/article/managing-complex-technology-innovation/69383?camid=4v1a)