Avionics in the view of reliability

Fly-By-Wire

Vít Hampl

Avionics

The development of avionics software products has to conform to the DO-178B standard. DO-178B does not prescribe a specific development process, it identifies important steps inside a development process and defines objectives for each of these steps. DO-178B distinguishes the development processes from “integral” processes that are meant to ensure correctness control and confidence of the software life cycle processes and their outputs. The verification process is part of the integral processes.

Development Processes

Four processes are identified:

- The software requirements process develops High Level Requirements (HLR) from the outputs of the system process;
- The software design process develops Low Level Requirements (LLR) and Software Architecture from the HLR;
- The software coding process develops source code from the software architecture and the LLR;
- The software integration process loads executable object code into the target hardware for hardware/software integration.

Each of the above mentioned processes is a step towards the actual software product.

Verification Process

The results of all activities1 of the development must be verified. Detailed objectives are defined for each step of the development, typically some objectives are defined on the output of a development process itself and also on the compliance of this output to the input of the process that produced it. For example, Figure 2 presents the objectives related to LLR. Arrows are labeled with verification objectives; the loop arrow on LLR means the objectives only concern LLR while the arrow between LLR and HLR means that objectives address relationships between LLR and HLR. On one hand, LLR shall be accurate and consistent, compatible with the target computer, verifiable, conform to requirements

standards, and they shall ensure algorithm accuracy. On the other hand, LLR shall be compliant and traceable to HLR. Verification means identified by DO-178B are reviews, analyses and test. Reviews provide a qualitative assessment of correctness. Analyses provide repeatable assessment of correctness. Reviews and analyses are used for all the verification objectives regarding HLR, LLR, software architecture and source code. Test is used to verify that the executable object is compliant with LLR and HLR. Test is always based on the requirements (functional test) and shall include normal range and robustness cases. A structural coverage analysis is performed to ensure that the software has been tested enough (different coverage criteria are used depending on the criticality level of the software).

Deductive Methods

The first kind of formal technique is deductive proof based on Hoare logic, and the computation of Dijkstra's weakest precondition predicate transformer. The objective is to prove user defined properties on a given program. Properties must be formally expressed in logic. This technique proceeds in two steps:

- computation of the verification conditions: post-conditions (properties that should hold after the execution of the program) are defined, this first step analyses the program and computes the conditions that must hold for these post-conditions to be verified;
- proof of the verification conditions: a theorem prover is used to prove the conditions computed before.

The first step is completely automated, the second step usually requires interaction with the user, but automation can be improved by the definition of specific heuristics. Several tools exist for different programming languages (mostly C and Java), for example Caveat and Frama-C

Abstract Interpretation Based Static Analysis

The second kind of techniques are techniques based on Abstract Interpretation. The principle of Abstract interpretation is the construction of a sound approximation of the

semantics of programs. A specific approximation is generated for each particular property being analyzed. Abstract interpretation is a completely automated technique. It may produce “false positives” (errors that can occur on the approximation of the program that has been computed, but cannot occur on the real program). The challenge is thus to be able to build a precise enough approximation in order to have as few false positives as possible. This usually implies a specialization of the technique with respect to the analyzed programs.

Usage of formal methods

Unit Proof: Within the development process of the most safety-critical avionics programs, the unit verification technique is used for achieving DO-178B objectives related to the verification of the executable code with respect to the Low Level Requirements, the classical technique being the Unit Tests.

Worst Case Execution: Time analysis: In real-time systems, computing correct values is not enough. Indeed, the program must also compute these values in due time in order to remain synchronized with the physical environment. The scheduling of the most critical avionics real-time programs is an off-line scheduling. This means that the serialization (single processor) of the various program tasks is performed at design time, leading to a fixed interleaving of these tasks. In this context, schedulability analysis boils down to the safe computation of an upper bound of the Worst Case Execution Time of the program tasks, almost exclusively.

Maximum stack usage computation: The amount of memory given to a task of an avionics program is determined statically when the program is built. If any task stack of a program actually requires more memory than what has been allocated statically, a stack overflow exception is raised during execution. In order to avoid this serious problem, a safe upper bound of each stack of the program must be computed. With these figures, the computation a safe upper bound of the total amount of memory used for stacks is

performed, by means of an analysis that takes into account some mechanisms such as interrupt tasks or Operating System calls.

Fly-By-Wire

Beginning of the Fly-By-Wire – armed forces

The Digital Fly-By-Wire (DFBW) concept uses an electronic flight-control system coupled with a digital computer to replace conventional mechanical flight controls.

The first test of a DFBW system in an aircraft was in l972 on a modified F-8 Crusader at the Flight Research Center, Edwards, Calif. (now Dryden Flight Research Center). It was the forerunner of the fly-by-wire flight control systems now used on the space shuttles and on today's military and civil aircraft to make them safer, more maneuverable and more efficient. It was safer because of its redundancies and because, for military aircraft, wires were less vulnerable to battle damage than the hydraulic lines they replaced. It was more maneuverable because computers could command more frequent adjustments than a human pilot and designers could do away with features that made the plane more stable and thus harder to maneuver. For airliners, computerized flight control could also ensure a smoother ride than a human pilot alone could provide. Finally, digital fly-by-wire was more efficient because it was lighter and took up less volume than hydraulic controls and thus either reduced the fuel required to fly with the extra weight and/or permitted carrying more passengers or cargo. It also required less maintenance than older systems.

In the first few decades of flight, pilots controlled aircraft through direct force — moving control sticks and rudder pedals linked to cables and pushrods that pivoted control surfaces on the wings and tails.

As engine power and speeds increased, more force was needed and hydraulically boosted controls emerged. Soon, all high performance and large aircraft had hydraulic-mechanical

flight-control systems. These conventional flight-control systems restricted designers in the configuration and design of aircraft because of the need for flight stability.

As the electronic era evolved in the 1960s, so did the idea of aircraft with electronic flight-control systems. Wires replacing cables and pushrods would give designers greater flexibility in configuration and in the size and placement of components such as tail surfaces and wings. A fly-by-wire system also would be smaller, more reliable, and in military aircraft, much less vulnerable to battle damage. A fly-by-wire aircraft would also be much more responsive to pilot control inputs. The result would be more efficient, safer aircraft with improved performance and design.

On May 25, 1972, the highly modified F-8 became the first aircraft to fly completely dependent upon an electronic flight-control system. The pilot was Gary Krier.
Wires from the control stick in the cockpit to the control surfaces on the wings and tail surfaces replaced the entire mechanical flight-control system in the F-8. The heart of the system was an off-the-shelf backup Apollo digital flight-control computer and inertial sensing unit which transmitted pilot inputs to the actuators on the control surfaces.
The first phase of the DFBW program validated the fly-by-wire concept and quickly showed that a refined system — especially in large aircraft — would greatly enhance flying qualities by sensing motion changes and applying pilot inputs instantaneously.
The DFBW program lasted 13 years. The final flight — the 210th of the program — was made April 2, 1985, with Dryden Research Pilot Ed Schneider at the controls.

Fly-By-Wire in civil usage

The first electrical flight control system for a civil aircraft was designed by Aerospatiale and installed on Concorde. This is an analog, full-authority system for all control surfaces and copies the stick commands onto the control surfaces. A mechanical back-up system is provided on the three axes. The first generation of electrical flight control systems with digital technology appeared on several civil aircraft at the start of the 1980's. These systems control the slats, flaps and spoilers. These systems have very stringent safety

requirements. However, loss of a function is permitted as the only consequences are a supportable increase in the crew's workload. In the second generation of EFCS, all control surfaces are controlled electrically by high/level control laws in normal operation and that the system is designed to be available under all circumstances. This system was built to very stringent dependability requirements both in terms of safety and availability. The basic building blocks are the fail-safe control and monitoring computers. These computers have stringent safety requirements and are functionally composted of a control channel and a monitoring channel. The control channel ensures the function allocated to the computer. The monitoring channel ensures that the control channel operates correctly. A high level of redundancy is built into the system. Special attention has been paid to possible external aggressions. The system is built to tolerate both hardware and software design faults. The overall dependability of the aircraft is also reinforced by the stability augmentation and flight envelope protection provided by the system. The aircraft safety is demonstrated using qualitative and quantitative assessments, this approach is consistent with the regulation. Qualitative assessment is used to deal with design faults, interaction faults and external environmental hazard. For physical faults, both qualitative and quantitative assessments are done.

On a conventional airplane, the pilot orders are transmitted to the actuators by an arrangement of mechanical components. In addition, computers are modifying pilot feels on the controls. and auto-pilot computers are able to control the actuators. In auto-pilot mode, the flight controls computers take their orders from the auto-pilot computers. The flight controls computers are of a control and monitoring type.

Control and monitoring computers

Functionally, the computers have a control and a monitoring channel. The control channel ensures the function allocated to the computers. The monitoring channel ensures that the control channel operates correctly.

These computers can usually be considered as being two different and independent computers placed side by side. These two computers have different functions and are placed adjacent to each other only to make aircraft maintenance easier. Both command and

monitoring channels of one computer are active simultaneously, or waiting simultaneously so go from stand-by to active state. Usually two computers with one control and one monitoring channel for each of them are used.

Technical specification

The specification of a computer includes on the one hand and equipment and software development technical specification used to design the hardware and, in part the software and on the other hand and equipment functional specification, which accurately specifies the functions implemented by the software. This functional specification is written using Computer-Assisted Specification. One of the benefits of this method is that each symbol used has a formal definition with strict rules governing its interconnections. The specification is under the control of a configuration management tool and its syntax is partially checked automatically.

Each channel includes one or more processors, their associated memories, I/O circuits a power supply unit and specific software. When the results of one of these two channels diverges significantly, the channel or channels which detected this failure cut the links between the computer and the exterior. The system is designed so that the computer outputs are then in a dependable state. Failure detection is mainly achieved by comparing the difference between the control and monitoring commands with a predetermined threshold. This schema therefore allows the consequences of a failure of one of the computer's components to be detected and prevents the resulting error from propagating outside of the computer. This detection method is completed by monitoring for good execution for the program via its sequencing. Flight control computers must be especially robust. They are especially protected against overvoltages and undervoltages, electromagnetic aggressions and indirect effects of lightning. They are cooled by a ventilation system but will operate correctly even if ventilation is lost.

Software development

To make software validation easier, the various tasks are sequenced in a predetermined order with periodic scanning of the inputs. Only the clock can generate interrupts used to

control task sequencing. This sequencing is deterministic. A part of the task sequencer validation consists in methodically evaluating the margin between the maximum execution time for each task and the time allocated to this task.
An important task is to check the conformity of the software with its specification. This is performed by means of tests and inspections. The result of each step in he development process is checked against its specification. For example, a code module is tested from its specification. This test is first of all functional, then structural. Adequate coverage must be obtained for the internal structure and input range, however this does not mean, that the tests are exhaustive. For example, for the structural test of a module, the equivalence classes are defined for each input. The tests must cover the module input range taking these equivalence classes and all module branches as a basis. These equivalence classes and a possible additional test effort have the approval of the various parties involved. The software control channel is different from that of the monitoring channel.

Failure detection and reconfiguration

Latent failure: Certain failures may remain masked a long time after their creation. A typical case is that of a monitoring channel made passive and detected only when the monitored channel itself fails. Tests are conducted periodically so that the probability of the occurrence of an undesirable event remains sufficiently low. Typically, a computer runs its self-tests and tests its peripherals during the power up of the aircraft and therefore at least once a day.
Comparison threshold – robustness: The results are compared in the two channels. The difference between the results of the control and monitoring channels are compared with a threshold. A failure is detected if the difference between the channels is above an allowable threshold. This must be confirmed before the computer is disconnected. The confirmation consists in checking that the detected failure lasts for a sufficiently long period of time. The detection parameters must be sufficiently "wide" to avoid unwanted disconnections and sufficiently "tight" so that undetected failures are tolerated by the computer's environment. More precisely, all system tolerances are taken into account to prevent

undue failure detection, and errors which are not detectable are assessed with respect of their handling quality, and structural loads effect.

Redundancy: The redundancy aspect is handled at system level. The functions of the system are divided out between all the computers so that each one is permanently active at least on one subassembly of its functions. For any given function, one computer is active the others are in standby. As soon as the active computer interrupts its operation, one of the standby computers almost instantly changes to active mode without a jerk or with a limited jerk on the control surfaces. Typically, duplex computers are designed so that they permanently transmit healthy signals and so that the signals are interrupted at the same time as the "functional" outputs following the detection of a failure.

System Validation

The system validation proceeds through several different steps:

- peer review of the specifications, and their justification
- analysis, most notably the System Safety Assessment which, for a given failure condition, check that the monitoring and reconfiguration logic allow to fulfill the quantitative and qualitative objectives, but also analysis of system performances, and integration with the structure
- tests with a simulated system, taking credit to the automatic programming of the functional specification, with a coupling with a rigid aircraft model
- test of an equipment on a partial test-bench, with input simulation and observation of internal variables
- tests on iron bird and flight simulator. The iron bird is a test bench with all the system equipment, installed and powered as on aircraft. The flight simulator is another test bench with an aircraft cockpit, flight controls computers, and coupled with a rigid aircraft model. The iron bird and the flight simulator are coupled for some tests.
- flight tests, on up to four aircraft, fitted with an "heavy" flight test instrumentation. More than 10000 flight controls parameters are permanently monitored and recorded.

The working method for these tests is twofold:

- a deterministic way, based on a test program, with a test report answering

- a way which takes credit of the daily use of these test facilities for work on other systems, for demonstration, or test engineer and pilot activity. If the behavior of the system is not found satisfactory, a Problem Report is risen, registered and investigated.

Fly-by-optics

Fly-by-optics is sometimes used instead of fly-by-wire because it can transfer data at higher speeds, and it is immune to electromagnetic interference. In most cases, the cables are just changed from electrical to fiber optic cables. Sometimes it is referred to as "Fly-by-light" due to its use of Fiber Optics. The data generated by the software and interpreted by the controller remain the same.

Power-by-wire

Having eliminated the mechanical circuits in fly-by-wire flight control systems, the next step is to eliminate the bulky and heavy hydraulic circuits. The hydraulic circuit is replaced by an electrical power circuit. The power circuits power electrical or self-contained electro-hydraulic actuators that are controlled by the digital flight control computers. All benefits of digital fly-by-wire are retained.
The biggest benefits are weight savings, the possibility of redundant power circuits and tighter integration between the aircraft flight control systems and its avionics systems. The absence of hydraulics greatly reduces maintenance costs. This system is used in the Lockheed Martin F-35 and in Airbus A380 backup flight controls. The Boeing 787 will also incorporate some electrically operated flight controls (spoilers and horizontal stabilizer), which will remain operational with either a total hydraulics failure and/or flight control computer failure.

Intelligent Flight Control System

A newer flight control system, called Intelligent Flight Control System (IFCS), is an extension of modern digital fly-by-wire flight control systems. The aim is to intelligently compensate for aircraft damage and failure during flight, such as automatically using

engine thrust and other avionics to compensate for severe failures such as loss of hydraulics, loss of rudder, loss of ailerons, loss of an engine, etc. Several demonstrations were made on a flight simulator where a Cessna-trained small-aircraft pilot successfully landed a heavily-damaged full-size concept jet, without prior experience with large-body jet aircraft. This development is being spearheaded by NASA Dryden Flight Research Center. It is reported that enhancements are mostly software upgrades to existing fully computerized digital fly-by-wire flight control systems.

Sources

http://personales.upv.es/juaruiga/teaching/TFC/Material/Trabajos/AIRBUS.PDF
http://www.nasa.gov/centers/dryden/news/FactSheets/FS-024-DFRC.html
http://www.laas.fr/IFIPWG/Top3/07-Traverse.pdf
http://www.aviationexplorer.com/Fly_By_Wire_Aircraft.html
http://www.springerlink.com.nukweb.nuk.uni-lj.si/content/0265621453714708/fulltext.pdf