WInnF'12

The Software Communications Architecture (SCA) and DO-178B Avionics Certification

RTCA DO-178B, Software Considerations in Airborne Systems and Equipment Certification

Dr. Rainer Storn

2013 January 10

ROHDE&SCHWARZ

Content

- Airborne SDRs and Objectives of the DO-178B Standard
- Objectives of the SCA
- Conflict Between DO-178B and SCA
- Measures Necessary for Mitigation

Content

- Airborne SDRs and Objectives of the DO-178B Standard
- Objectives of the SCA
- Conflict Between DO-178B and SCA
- Measures Necessary for Mitigation

Airplane accidents due to SW

The Chinook Mark 2 helicopter crashed on June 2nd 1994, BBC News.

Possible cause: **engine control software**

Flight-control software has been blamed for the crash of the prototype Swedish JAS39 Gripen fighter February 1989

Quantas Flight 72 Oct. 7 2008, with Airbus A330 made emergency landing after uncommanded pitch down manoeuvres due to fault in **inertial reference system software**

After the $45 million Global Hawk spy plane had finished a successful flight test on Dec. 6, 1999 and was at a full stop, a **mission planning software** problem caused the aircraft to accelerate to about 175 mph until it veered off the base's main runway

Demand for Functional Assurance is on the Rise

Avionics

1992
DO-178B/C
(ED-12B/C),
1999

RTCA

EUROCAE

DO-254
2000

RTCA

Air Traffic Control (ATC)

ED-153
2009

ED-109A
2012

EUROCAE

Road Vehicles

ISO 26262
2011

ISO

Typical Application Areas for DO-178B Assurance Levels

Level	Corresponding Failure Description	Function
A (cata-strophic)	Failure may cause a crash	Fly by wire controls [1)] Jet Engine control [1)] Auto pilot [1)]
B (hazar-dous)	Failure has a large negative impact on safety or performance **...**	IFF (friend or foe) [1)] Missile launch [1)]
C (major)	Failure is significant, but has a lesser impact than hazardous failure	Data mining [1)] Communication [2)]
D (minor)	Failure is noticeable, but has a lesser impact than a major failure	Passenger reading lights
E (no effect)	Failure has no impact on safety, aircraft operation, or crew workload	Entertainment System

[1)] Akos Horvath, Standards in Avionics System Development, Budapest University of Technology and Economics, oct. 2008.

[2)] SAE / ARP 5150, Safety Assessment of Transport Airplanes in Commercial Service, nov. 2003 .

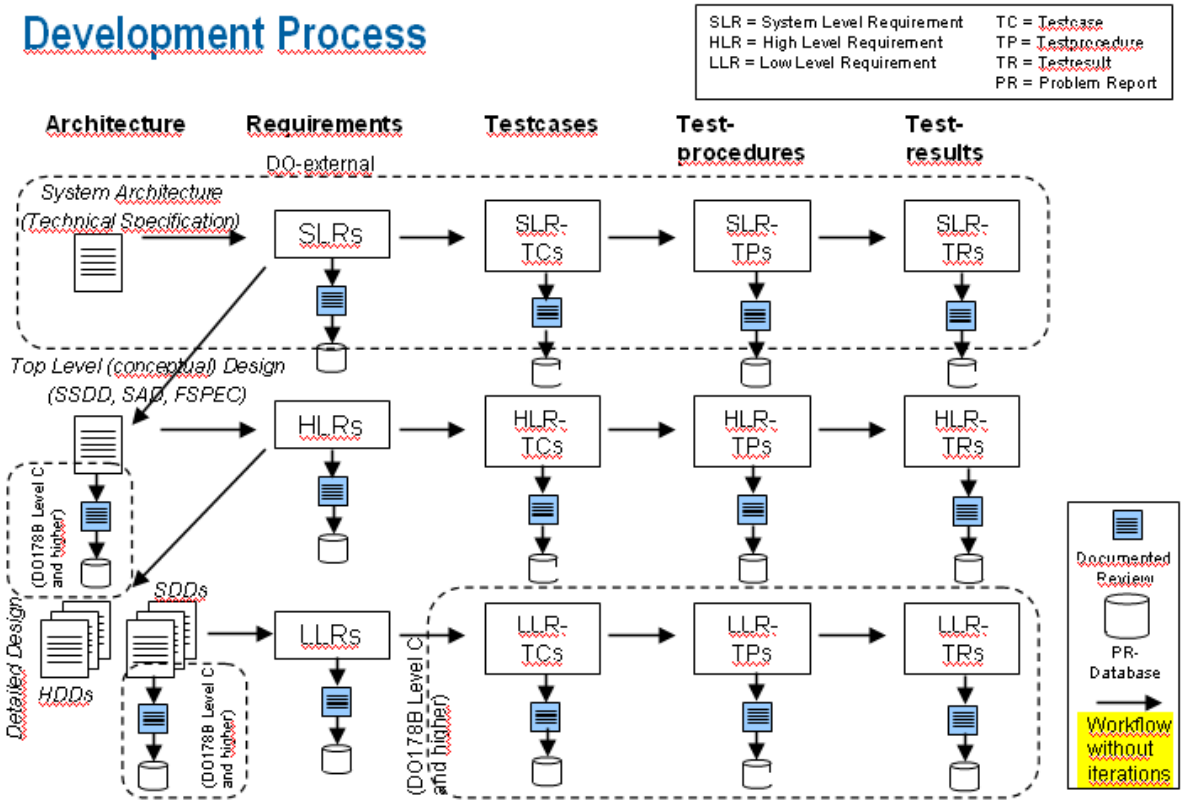
Objectives of DO-178B (the easy part: Processes)

- The Main goal of the DO-178B is to improve Design Assurance
- The existence of several Processes is stipulated
  - Planning process
  - Development process
  - Configuration Mgmt process
  - *Verfication* process (Review, Analysis, Test)
  - Quality Assurance process

Planning / Development / Correctness

"Guilty until proven innocent"

Evidences



Objectives of DO-178B (the hard part: Verification!)

- Evidence for the functionality of the design → Accuracy & Consistency [1]:
  - Determinism:
    - Prevention of stack overflow (no recursive functions, no large structures unless non-hazardousness is proven)
    - Prevention of memory leaks (no dynamic memory allocation → OOP?, SCA?)
    - Evidence of „worst case execution time“ → deterministic time management
    - Prevention of data overflow, uninitialized variables, task- and interrupt conflicts / deadlocks
- No dead code allowed
- Testing effort because of coverage demands [3] → code coverage, requirements coverage, interface coverage (data & control coupling)
- Demands on COTS-SW
  - Must be developed according to DO-178B [2] → Manufacturer must provide „DO-package“ or evidences must be provided by reengineering.

[1] RTCA / DO178B, Software Considerations in Airborne Systems and Equipment Certification, december 1, 1992, ch. 6.3.4 f & Table A-5 → **Level C** and higher.

[2] RTCA / DO178B, Software Considerations in Airborne Systems and Equipment Certification, december 1, 1992, ch. 2.4 f : „COTS software included in airborne systems or equipment should satisfy the objectives of this document.“

[3] E.g.. for Level C: Statement Coverage, Data & Control Coupling. All Levels: 100% Requirements Coverage for Testcases.

R&S®MR6000A – DO178B/DO-254 compliant SDR

A400M

AW101

- **Multitude of waveforms:**
  - STANAG 4204, 4205
  - HQ I/II & SATURN
  - Link 11
- **Optional NATO embedded COMSEC**
- **Efficient SW**
  - 15s for cold start to operational state of any waveform
  - 1s for waveform change
  - < 5MB footprint
- **Dimensions: ARINC 600 housing**
- **Excellent RF-parameters**
  - Rx sensitivity: -107dBm
  - Intermodulation suppression: 85dB
  - Crossmodulation immunity according to U.S. standard ARINC 716
  - Spurious response suppression: 100dB
  - High transmit power under real-world conditions: 20W RF power at VSWR = 3
  - Low transmitter noise: noise power density -174dBc at a carrier offset of 10MHz

Content

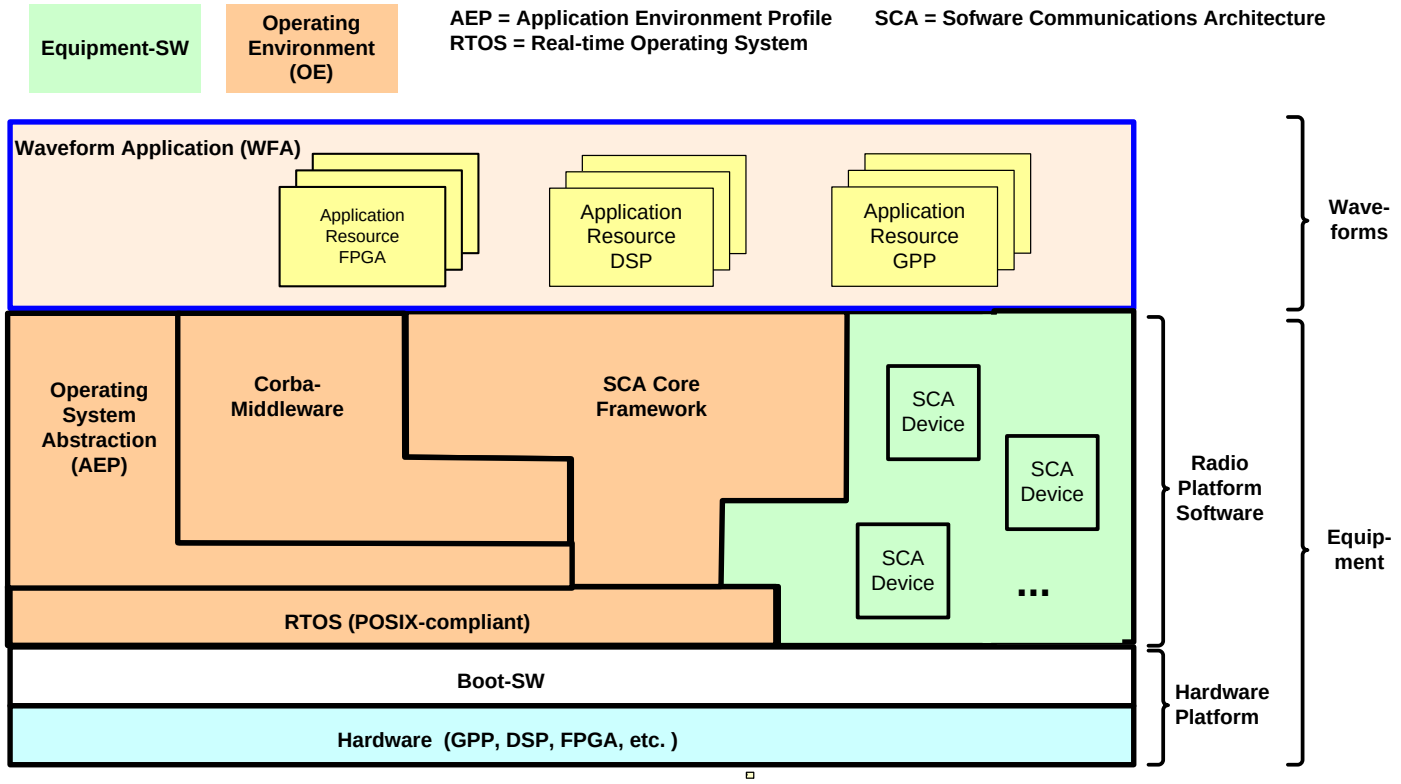
- Airborne SDRs and Objectives of the DO-178B Standard
- Objectives of the SCA
- Conflict Between DO-178B and SCA
- Measures Necessary for Mitigation

Objectives of SCA [1)]

- Support a family of radios
  - Interoperable
  - Programmable
  - Scaleable
  - Affordable
- Maximizes independence of SW from HW
  - Application and device portability & reuse → Abstraction
  - Rapid technology insertion over time
- Extensible to new waveforms and/or HW components → Flexibility
- Incorporates embedded, programmable INFOSEC
- Supports JTRS ORD
  - Operator reconfigurable
  - Multiple waveforms (legacy and new)
  - Simultaneous multichannel operation.

[1)] Raytheon Company, SCA Training for Developers and Testers, sept. 16 – 20, 2002.

SCA – Implementation Layer



- Distributed Components, Object Orientation, Configurability (via XML-files)
- Separation of Platform (Operating Environment) and Waveform (application)
  - Platform contains POSIX RTOS, CORBA ORB, Core Framework
  - Common Services and APIs to support device and application portability.

Content

- Airborne SDRs and Objectives of the DO-178B Standard
- Objectives of the SCA
- Conflict Between DO-178B and SCA
- Measures Necessary for Mitigation

Object Oriented Programming (OOP) Key Concepts

"C makes it easy to shoot yourself in the foot; C++ makes it harder, but when you do it blows your whole leg off."
Bjarne Stroustrup

Concept (abstraction)	Selection of Problems for DO178B-based Verification
inheritance	I dead or deactivated code when superclass methods are replaced by sub-class methods or superclass is not instantiated I Unexpected behaviour for multiple inheritance, overriding of attributes or methods I Complicates traceability and verification I difficulty in meeting the DO-178B/ED-12B objective of data and control coupling (Structural Analysis)
encapsulation	I Programmers may exercise unintended functionality since the structure of the objects is hidden (especially true for libraries)
Subtype polymorphism	I Potential for ambiguity and late binding make the system more complicated to verify I Polymorphic function calls may have difficulties to ensure deterministic realtime behaviour

OOP Key Concepts

Concept (abstraction)	Selection of Problems for DO178B-based Verification
overloading	I Can lead to unintended subprogram selection I Polymorphic and overloaded functions may make tracing and verifying the code difficult I additional demands on development tools such as compilers, linkers, and debuggers
Built-in-functions and libraries	I Libraries may change from compiler to compiler. I Complicates verification
Dynamic Object Creation	I non-deterministic behavior I Potential memory leakage I Reference ambiguity, fragmentation starvation, deallocation starvation, heap exhaustion, premature deallocation, lost update and stale reference, time bound allocation and deallocation

OOP Key Concepts

Concept (misc)	Selection of Problems for DO178B-based Verification
Exceptions	▪ Verification: Inconsistent state as a result of an exception not handled correctly ▪ Verification: It is difficult to estimate the time between when an exception has occurred and control has passed to a corresponding exception handler ▪ Verification: It is difficult to estimate memory consumption for exception handling
Constructors & Destructors	▪ C++ allows insertion of constructors and destructors by the compiler outside the control of the programmer

Challenges with today's CORBA solutions [1), 2)]

- First of all: CORBA is object oriented
- Failure to conform to FAA/NASA OO guidelines for DO-178B certification
- Failure to conform to language specific guidelines for safety critical systems
- Unrestricted use of dynamic allocation after initialization → may lead to memory leaks
- Use of unbounded data structures
- Use of algorithms with unpredictable or unbounded execution times
- Use of recursion
- Dynamic loading of classes
- Unrestricted use of exceptions
- Use of aliasing, involving pointers or arguments
- Choice of model for threads, synchronization and thread communication
- Oneway calls are „best effort" calls → non-determinism
- Potential for deadlock or starvation of threads within the ORB
- ORB size and complexity
- Lack of precise requirement specifications and design documentation
- Lack of reliability

[1)] Gary Daugherty, Rockwell-Collins, http://www.nitrd.gov/subcommittee/sdp/vanderbilt/motivating_examples/rockwell-corba-aircraft-exa.pdf, 2001.

[2)] John Bard, Vincent J. Kovarik.Jr., Software Defined Radio – The Software Communications Architecture, Wiley, 2007

Content

- Airborne SDRs and Objectives of the DO-178B Standard
- Objectives of the SCA
- Conflict Between DO-178B and SCA
- Measures Necessary for Mitigation

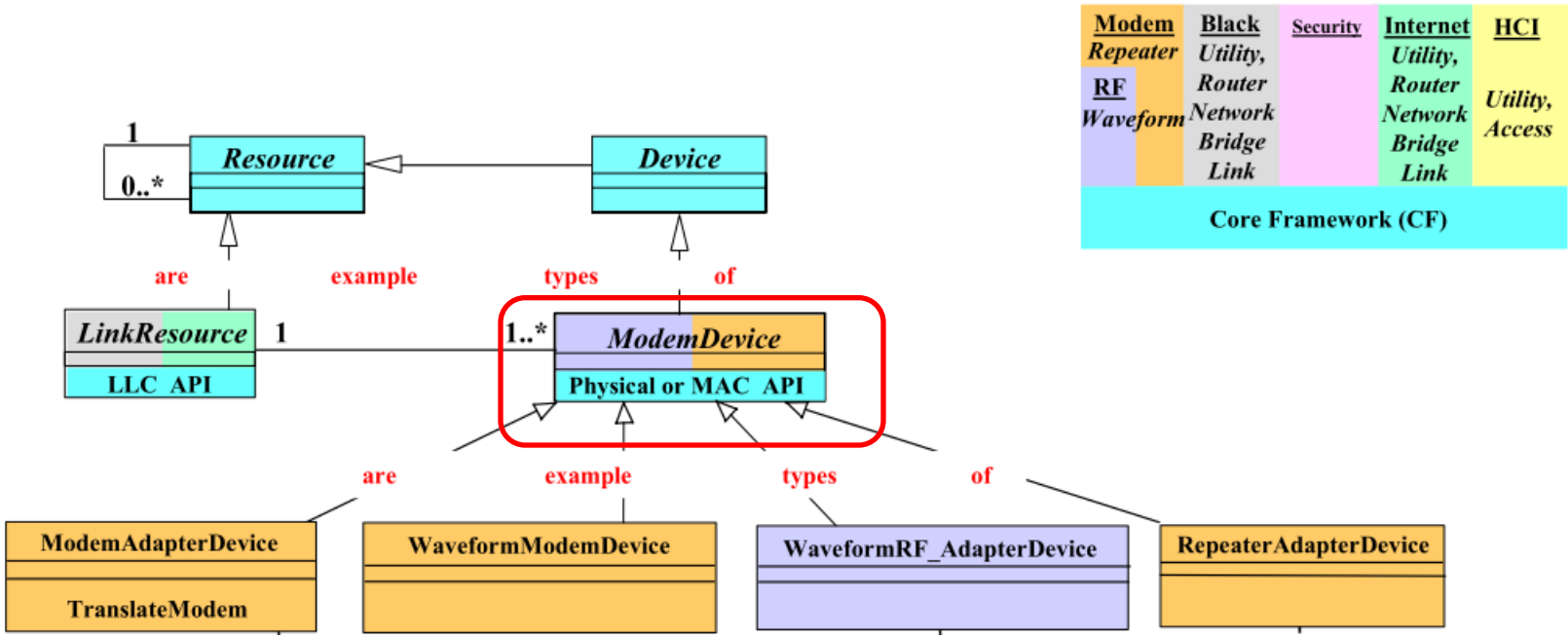
How to Use C++ for Safety Critical Applications

- For example MISRA C++ [1] or JSF C++ [2]
- Such standards reduce problematic OOP constructs (verifiable by tools):
  - multiple inheritance
  - recursions,
  - dead code
  - exceptions
  - dynamic memory allocation after initialization
  - …

[1] MISRA C++:2008, Guidelines for the use of the C++ language in critical systems, june 2008.

[2] Lockheed Martin, Joint Strike Figher Air Vehicle C++ Coding Standards for the System Development and Demonstration Program, dec. 2005.

SCA-Modem Device may be constructed fully DO-178B-compliant



- The functions, performed by the *ModemDevices* ... are not dictated by the CF [1)]
- Modem Device can be fully developed according to DO-178B

[1)] Joint Program Executive Office JTRS, *Software Communications Architecture Specification v2.2.2,* 2006

SCA 4.0 – Simplifications Make Verification More Manageable

- Existence of a lightweight profile
- Possible replacement of CORBA with more efficient and deterministic communication mechanisms
- Definition of lightweight AEP
- Omission of event channel, log capability, and application installation possible
- Reduction of interface complexity by introduction of optional inheritance

Employ DO-178B-Compliant COTS SW

- Examples: RTOS, IP-Stack, Voice-Codec
- DO-178B-compliant SW comes with a „DO-package“
  - Development Plans
  - Requirements, Testcases, Testprocedures, Testresults
  - Review Evidences
  - QA Evidences
  - Evidences for Requirements Coverage, Code Coverage, WCETA, Stack Analysis etc.

Keep Architecture as Simple as Possible

- Reduces number of testcases
- Simplifies the verification part demanded by DO-178B

```
#include <stdio.h>
void main(void)

{
  char *message[] = {"Hello ", "World"};
  int i;

  for (i = 0; i < 2; ++i)
    printf("%s", message[i]);
  printf("\n");
}
```

Metaphor:

„Hello World“ simplified

```
#include <iostream.h>
#include <string.h>

class string
{
private:
  int size;
  char *ptr;

public:
  string() : size(0), ptr(new char('\0')) {}

  string(const string &s) : size(s.size)
  {
    ptr = new char[size + 1];
    strcpy(ptr, s.ptr);
  }

  ~string()
  {
    delete [] ptr;
  }
  friend ostream &operator <<(ostream &, const string &);
  string &operator=(const char *);
};

ostream &operator<<(ostream &stream, const string &s)
{
  return(stream << s.ptr);
}

string &string::operator=(const char *chrs)
{
  if (this != &chrs)
  {
    delete [] ptr;
    size = strlen(chrs);
    ptr = new char[size + 1];
    strcpy(ptr, chrs);
  }
  return(*this);
}

int main()
{
  string str;
  str = "Hello World";
  cout << str << endl;
  return(0);
}
```

- **In addition(!):** keep memory footprint low (DO-254 → SEU)

Conclusion

Civil software safety standards like DO-178B will become a necessity also for military airborne SDRs

We want to have the flexibility of the SCA together with DO-178B compliance

- These two standards present opposing objectives
- It‘s a challenge, but feasibility is emerging

Milestones on the road are among others

- Keep the architecture simple
- Implementation of the SCA 4.0 („SCA Next“)
- Appropriate coding standards
- Buy DO-178B-compliant COTS components

Economical feasibility of the harmonization depends on

- Architectural details
- Maturity of the SW development process

Questions ?

Thank you