CONCEPTEUR, INTEGRATEUR, OPERATEUR DE SYSTEMES CRITIQUES

# DO178/ED12 B & C and Model Based Development and Verification SDL-FORUM 07

Hugues Bonnin

# Agenda



1. **DO178B/ED12B principles overview**
   a. Application context and history
   b. Generic approach
2. **Models places in DO178B/ED12B**
   a. In development process
   b. In verification process
   c. Examples
3. **DO178C/ED12C current work**
   a. Roadmap
   b. Organisation
4. **DO178-C/ED12-C – SG4 current work**

# 1 - DO178B/ED12B principles overview



## 1.a - Application context and history

# « Software Considerations In Airborne Systems And Equipment Certification »

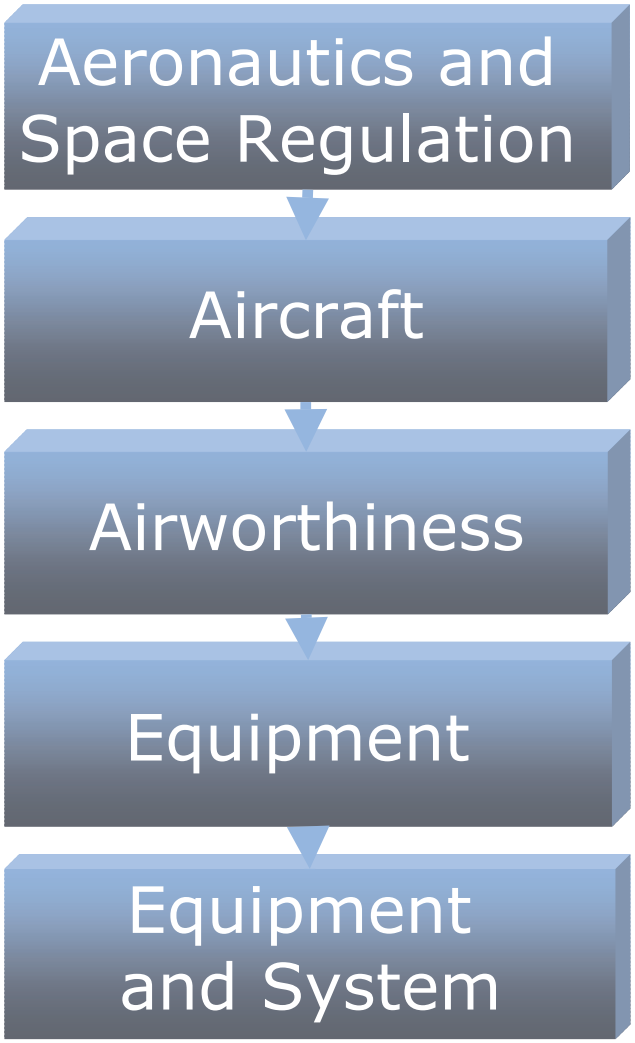


- **DO 178-B/ED 12-B is a standard**
  - established by working groups of **RTCA** and **EUROCAE** organizations in 1992
  - used to establish compliance of **airborne software** to Aeronautics and Space regulations
- **As this standard**
  - has been written by all actors of usage domain (applicants, authorities, scientists) and adopted by consensus in the working group,
  - is verified by independant authorities,
  - is interested only by safety objectives,

**its application is very seriously organized by the applicants, and very well adopted by the users.**

# DO178/ED12 among regulation






For each level, the (common) rules are edicted by :

- FAA (US authorities) : **F**ederal **A**viation **R**egulations
- JAA (~old EU authorities) : **J**oint **A**viation **R**equirements
- EASA (~new EU authorities) : **C**ertification **S**pecifications

FAR/JAR/CS 25

is one mean of compliance for software to

FAR/JAR/CS 25-1309

DO 178 /ED 12

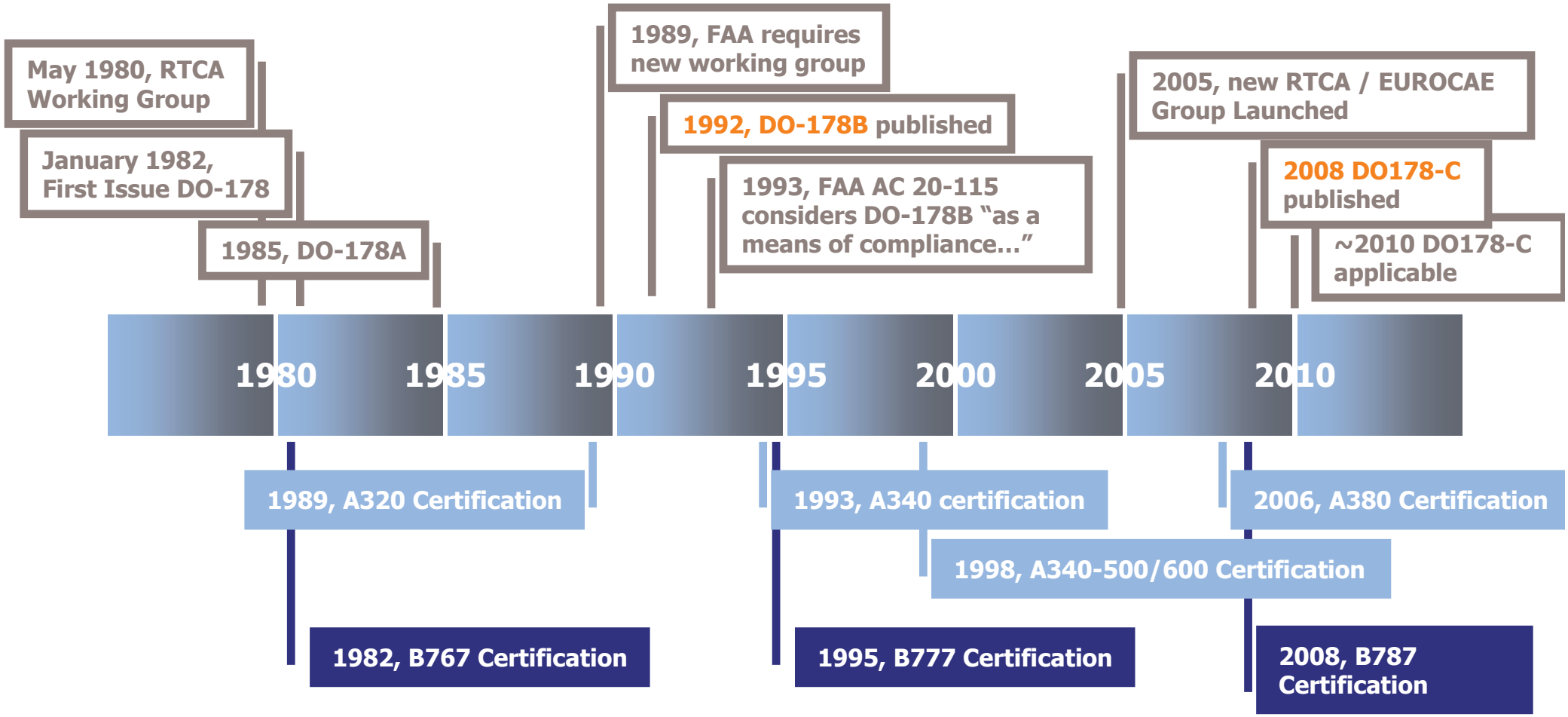
# DO178/ED12 and Certification



- **Only Aircraft and Engines are certified**
  - Airborne software are not certified
  - Airborne software are evaluated in the context of an Aircraft or Engine, and only in that context
- **Authorities evaluate if DO178/ED12 rules are well observed by the applicant, and provide certification credit for the Aircraft or Engine certification.**

# DO178/ED12 History





# 1 - DO178B/ED12B principles overview



## 1.b - A Generic approach

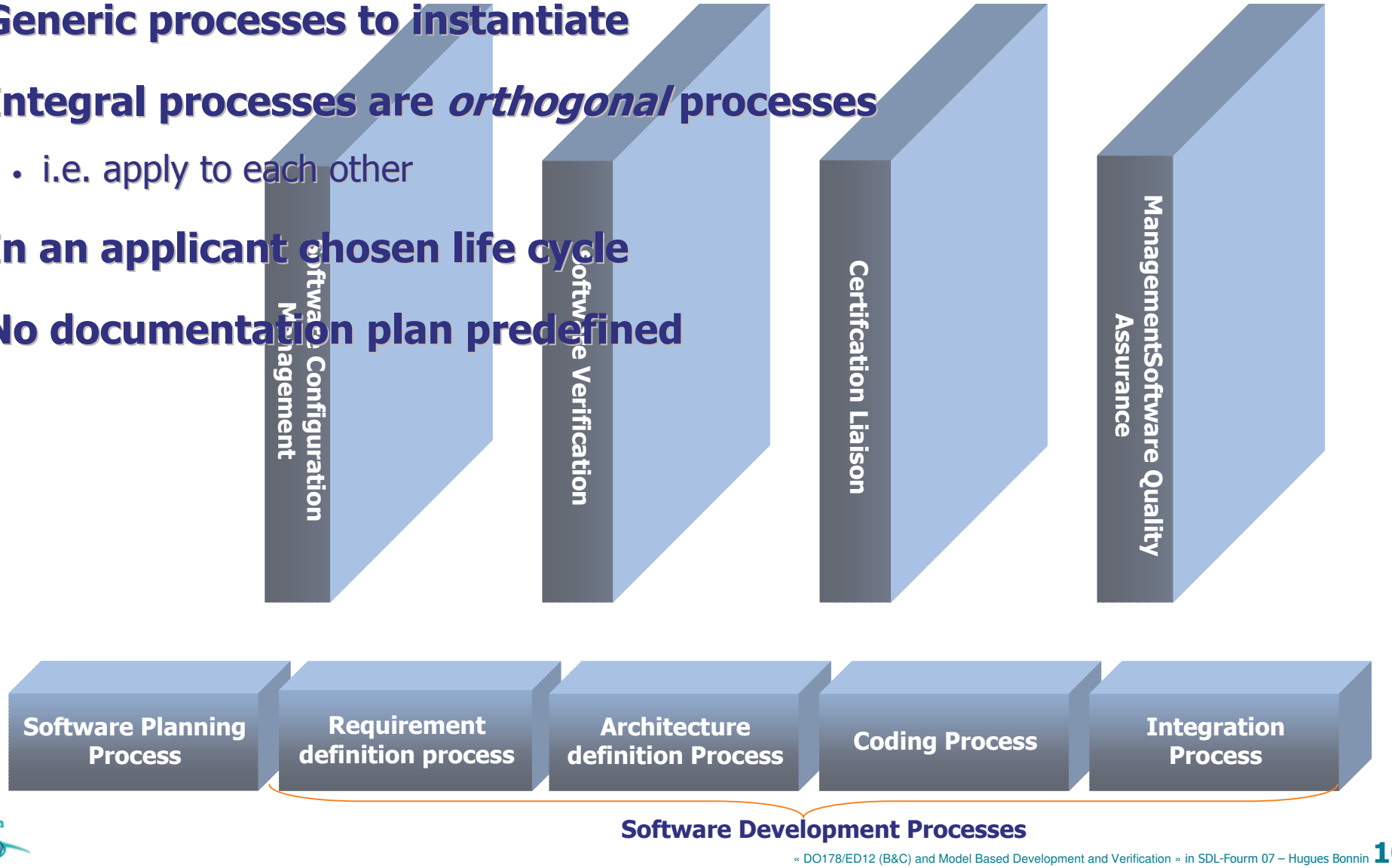
# DO178B/ED12B anatomy



# DO178B/ED12B Generic approaoch



- **Generic processes to instantiate**
- **Integral processes are *orthogonal* processes**
  - i.e. apply to each other
- **In an applicant chosen life cycle**
- **No documentation plan predefined**



# Objectives, not means



- **No life cycle scheme imposed**
  - V cycle, but iterative life cycle, or incremental life cycle
- **No method imposed**
  - Textual definition of requirements, semi-formal or formal language, partial or complete modelisation, functional or object approach…
- **No coding language imposed**
  - C, Ada, assembler, are all equally considered
- **For verification, one mandatory mean**
  - Tests is mandatory
  - But analysis and review complete the verification means
- **For all the processes (direct and integral), objectives are given, depending on *software level* (related to safety impacts of a software failure).**

# 2 - Models places in DO178B/ED12B



## 2.a – Models in development processes

# Requirements and Model (1)



- **Requirements and traceability are the central point of DO178-B/ED12-B**
  - From system level to code executing on target, DO178-B/ED12-B goal is to ensure that in the software emebedded in the Aircraft
    - Each need is correctly implemented
    - Nothing other that the needs is implemented
    - Behaviour is always safe
  - So at every refinement stage, each requirement is traced in the next stage outputs, and is traced by tests.
- **Modelisation may be used to define Requirements**
  - But, in the practise of the DO178-B/ED12-B, applicants and authorities has difficulties to accept that the model represent the requirements *alone*, i.e. *without a textual* expression of a requirement.
  - Models are more often used as a complementary mean to describe the needs and the requirements.
    - For example, a functional approach like SA-RT could be used to introduce the relationship and the cartography of the different function of a software, but the traced artefact will be a short paragraph of text.
  - This is true for each level of requirement : High Level Requirement (HLR), or Low Level Requirement (LLR)

# Requirements and Model (2)



- **To support requirement definition, some models are used**
  - Use case diagrams in UML
  - Formalized language like Lustre/SCADE, SDL

# Architecture and models (1)



- **Modelisation may be used to define Architecture**
  - In DO178-B/ED12-B context, it is common to use models to define and represent architecture. It is well accepted by authorities.
  - Methods may be « home made » or standardised
  - Specialized tools (CASE) may be used or not
- **DO178-B/ED12-B rules**
  - In chapter 5 of DO178-B/ED12-B, the characteristics of software architecture are :
    - Traceable to High Level Requirements
    - Verifiable and consistent
    - Compliant with the Software Design Standards
    - Compatible with the target computer
    - Control flow and data flow
  - Architecture is the breakdown of the software into components, related to each other with Control flow and data flow

# Architecture and models (2)



- **Static architecture**

  To conform to rules, all types of static architecture modelisation is correct

  - Structure Diagrams of UML : class diagram, package diagram, …
  - Systems, Blocks and Channels in SDL
  - System, Package, Subprogram in AADL
  - …

- **Dynamic architecture**

  To describe dynamic architecture, following concepts are usable

  - Behavioral Diagrams of UML
  - Process, signals, MSCs of SDL
  - Thread, process, event of AADL

# Coding from models (1)



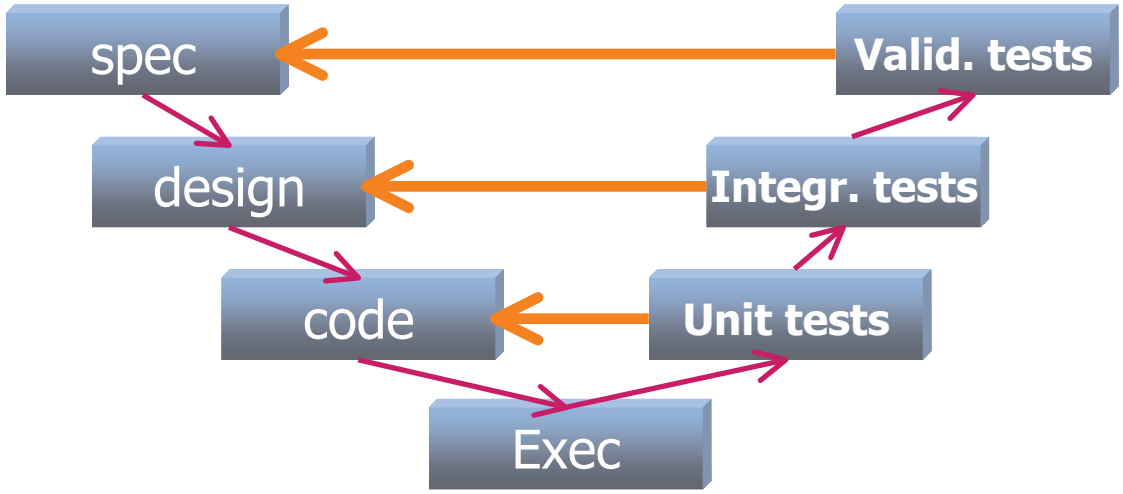
- **DO178-B/ED12-B coding phase**
  - After specification and design, the code has to be written, based on *low level requirements* (which are defined as the last requirements refinement)
- **Dichotomy of manual/automatic coding**
  - DO178-B/ED12-B considers that,
    - Either the code generator is qualified
    - Or the code is considered to be manually coded, that is, to be verified strictly in conformance to the DO178-B/ED12-B rules
- **Qualification of the code generator**
  - To qualify a generator, the applicant has to develop and verify it whith the same rules that the ones used to develop the embedded software, at the same level
    - It means that, for example, to qualify a generator of level A code, this generator must have been developed with all requirements of level A : strict configuration and change management , independance between development and tests, MC/DC tests, object code consideration…

  => This rules are very strict and hard to realise

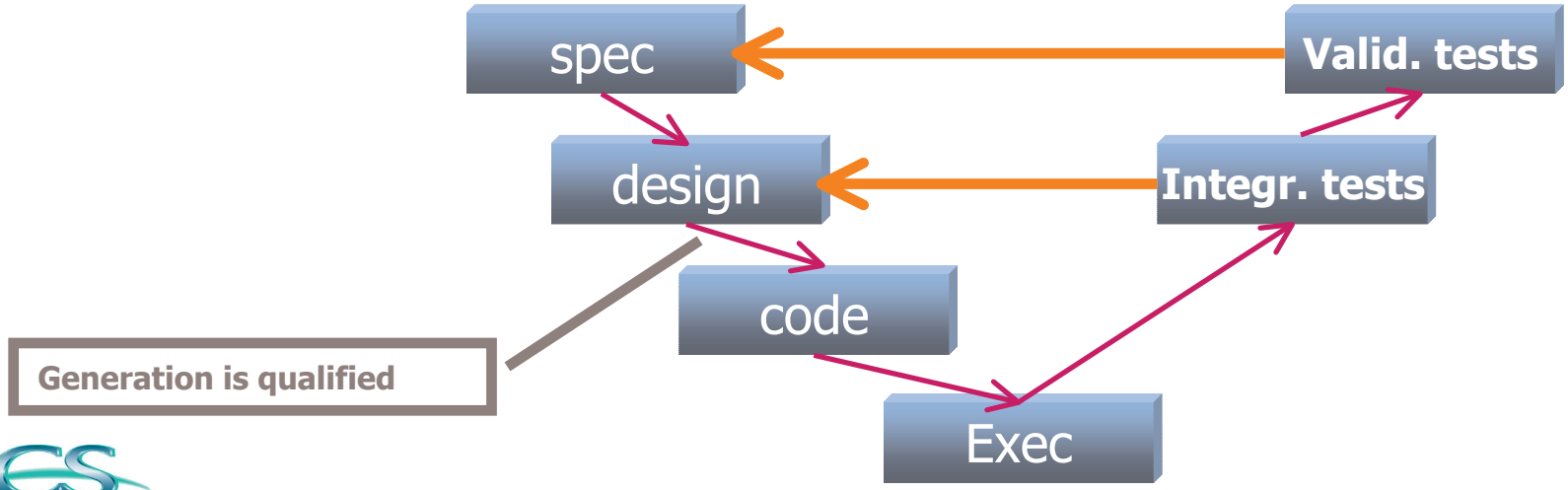
# Coding from models (2)



- **Typical cycle with manual coding**




- **Typical cycle with automatic coding**



# 2 - Models places in DO178B/ED12B



## 2.a – Models in verification process

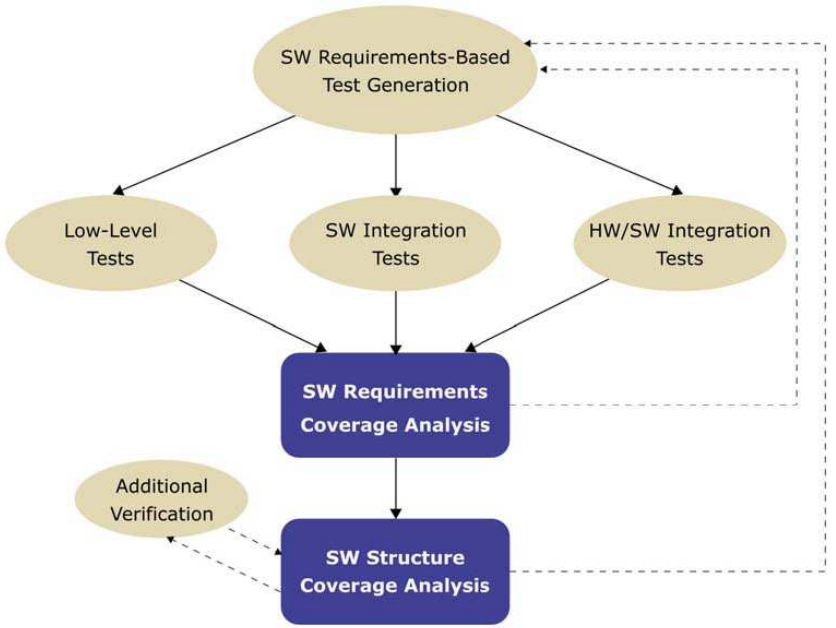
# Verification and models (1)



- **Tests in DO178-B/ED12-B**
  - The chapter 6 explains that *analysis*, *review* and *tests* may be used to verify the software. But the part on tests, and the objectives associated, is much more longer than the others :

    => The verification by tests is a mandatory mean in DO178-B/ED12-B

- **Coverage**
  - Associated to tests, coverage objectives are given :
    - **Requirements coverage** : this is the first criteria to be fulfilled ; all the tests has to be traced to requirements, all the requirements must be traced by at least one test.
    - **Code coverage** : this criteria comes after the requirement criteria.
  - Code coverage level depends on software level : level D=no coverage, level C=intructions, level B=decisions, level A=MC/DC



# Verification and models (2)



- **Execution on target**
  - The only valuable airborne software representation for DO178-B/ED12-B is *object code*
    - All the tests have to be run on target
    - At level A, some demonstration of traceability between source code and source object has to be done.
- **Validation/verification**
  - Reminder (definition) :
    - Validation : The process of determining that the requirements are the correct requirements and that they are complete. The system life cycle process may use software requirements and derived requirements in system validation.
    - Verification : The evaluation of the results of a process to ensure correctness and consistency with respect to the inputs and standards provided to that process.
  - DO178-B/ED12-B defines only activity and objectives of verification and no one of validation
- **Simulation**
  - All that constraints (tests, coverage, target, no validation) imply that in a Model Based approach, efficient means like simulation, or model-checking (formal method) are not usable with a certification credit :

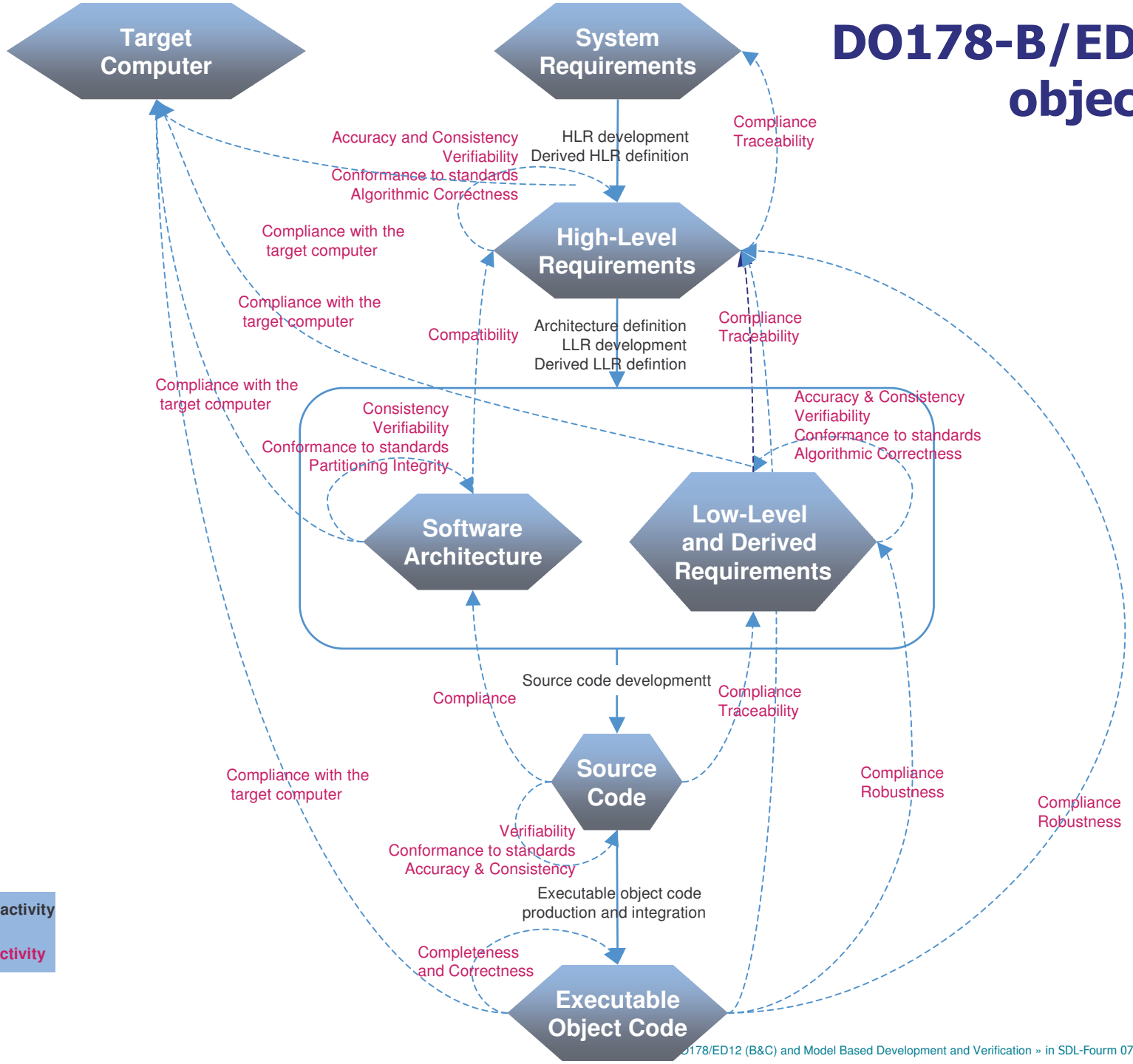
    **=> simulation is only used for industrial objectives but not for DO178-B/ED12-B certification**

# 2 - Models places in DO178B/ED12B



## 2.c – Examples

# DO178-B/ED12-B objectives



# Industrial examples



**(industrial examples will be developped during SdlForum07)**

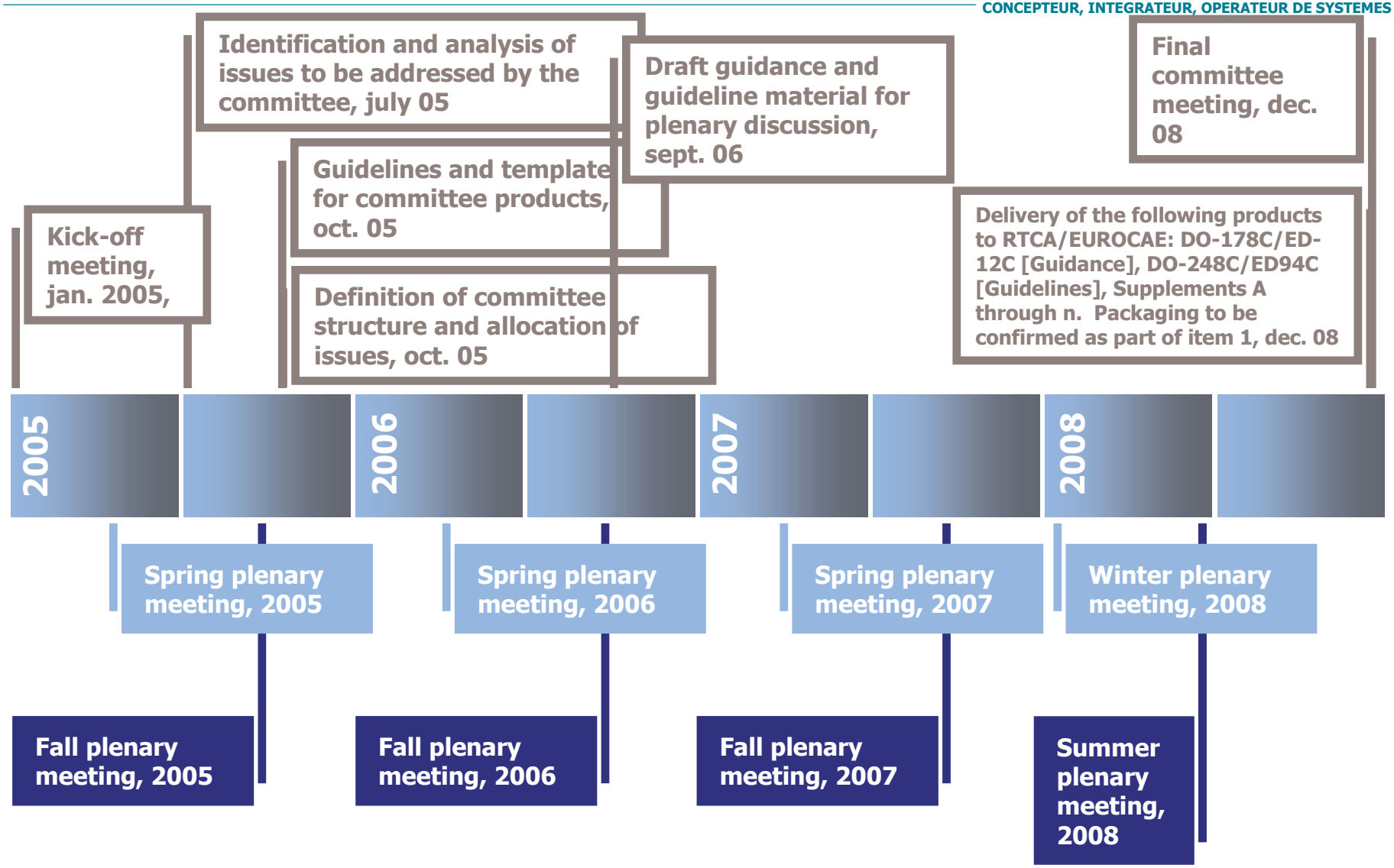
# 3 - DO178-C/ED12-C current work



## 3.a - Roadmap

# DO178-C/ED12-C Roadmap





# 3 - DO178-C/ED12-C current work



## 3.a - Organisation

# DO178-C/ED12-C organisation



- **Joint working group**
  - RTCA SC-205
  - EUROCAE WG-71
- **Working in subgroups**
  - The working group (~ 150 p.) is subdivided in 7 subgroups + executive committee :
    - SG1: SCWG Document Integration Sub-group
    - SG2: Issues & Rationale Sub-group
    - SG3: Tool Qualification Sub-group
    - **SG4: Model Based Design & Verification Sub-group**
    - SG5: Object Oriented Technology Sub-group
    - SG6: Formal Methods Sub-group
    - SG7: Safety Related Considerations Sub-group

Technical subgroups : define supplements

# 4 - DO178-C/ED12-C – SG4 current work

# DO178-C/ED12-C – SG4 current work



- System/software exchanges
- Models and Requirements
- Models and Simulation
- Models and Standards (or Rules)

**(to be developped according to last plenary meeting 10-14 september, Vienna)**