Certification in Aerospace

Some Issues and Aspects

IFIP 10.4 Meeting
July 1, 2007

Dr. Stefan Poledna
CEO, TTTech Computertechnik AG

Background

Cert Experience with Time-Triggered Communication

- DO-178B Level A software development (OS, driver, loading, COM …)
- DO-178B validation tool development
- DO-254 Level A reverse engineered communication controller (TTP)
- Formal methods for core algorithms

Honeywell Jet Engine Control
Dual use

Hamilton Sundstrand's Common Electronic Architecture (CEA)

Cabin Pressure Control System for Airbus A380

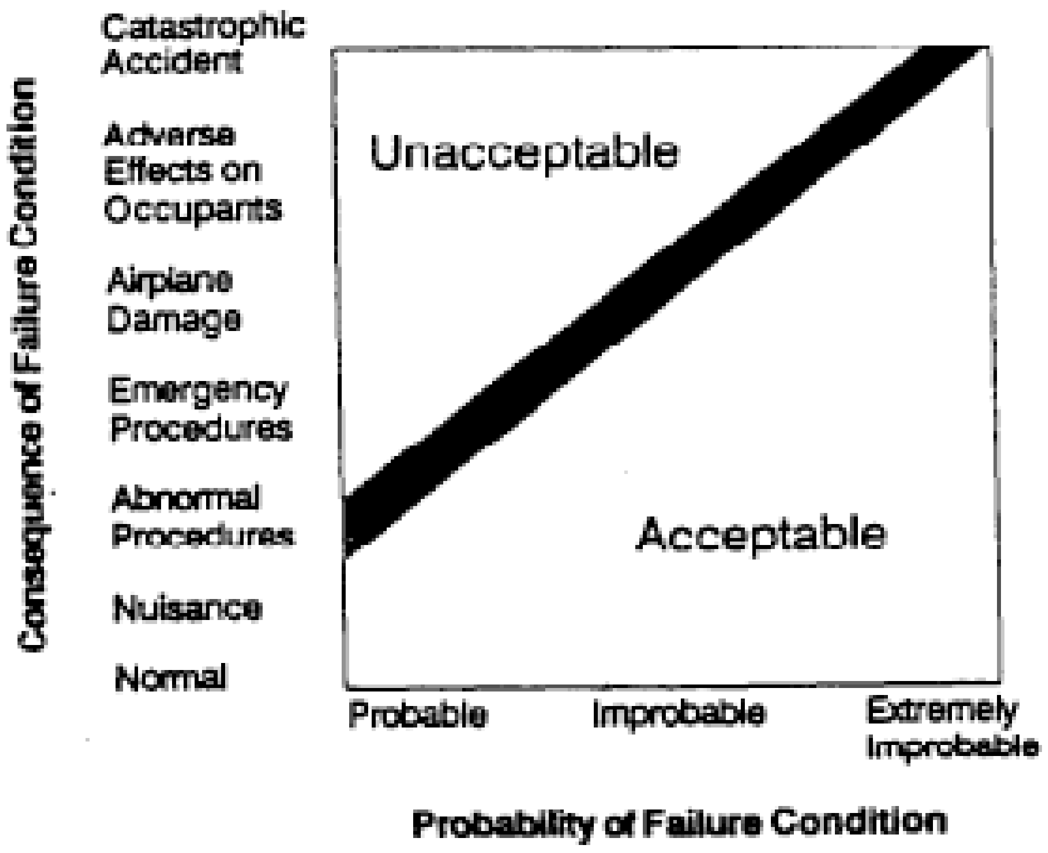
Boing 787
Electrical Power System
Cabine Pressure Control

Time-Triggered Architecture

Probability vs. Consequences of Failure

TTTech

Figure 1: Probability vs. Consequence Graph



FAA AC 25.1309-1A

The primary safety objective is to achieve an inverse relation of severity to probability.

System Design and Analysis Process

- **Functional Hazard Assessment (FHA)**
  - Identifcation of aircraft-level hazardous failure conditions
  - Classification of failure conditions according to the severity of their consequences
- **System Design & Analysis**
  - Required level of rigor increases with severity of failure consequences
  - Minor/Major/Catastrophic Failure Conditions map to requirements for qualitative and/or quantitative assessments according to flow chart
  - Credit for product service experience may be claimed for simple devices.
  - Also considers flight crew work-load and required ground crew actions

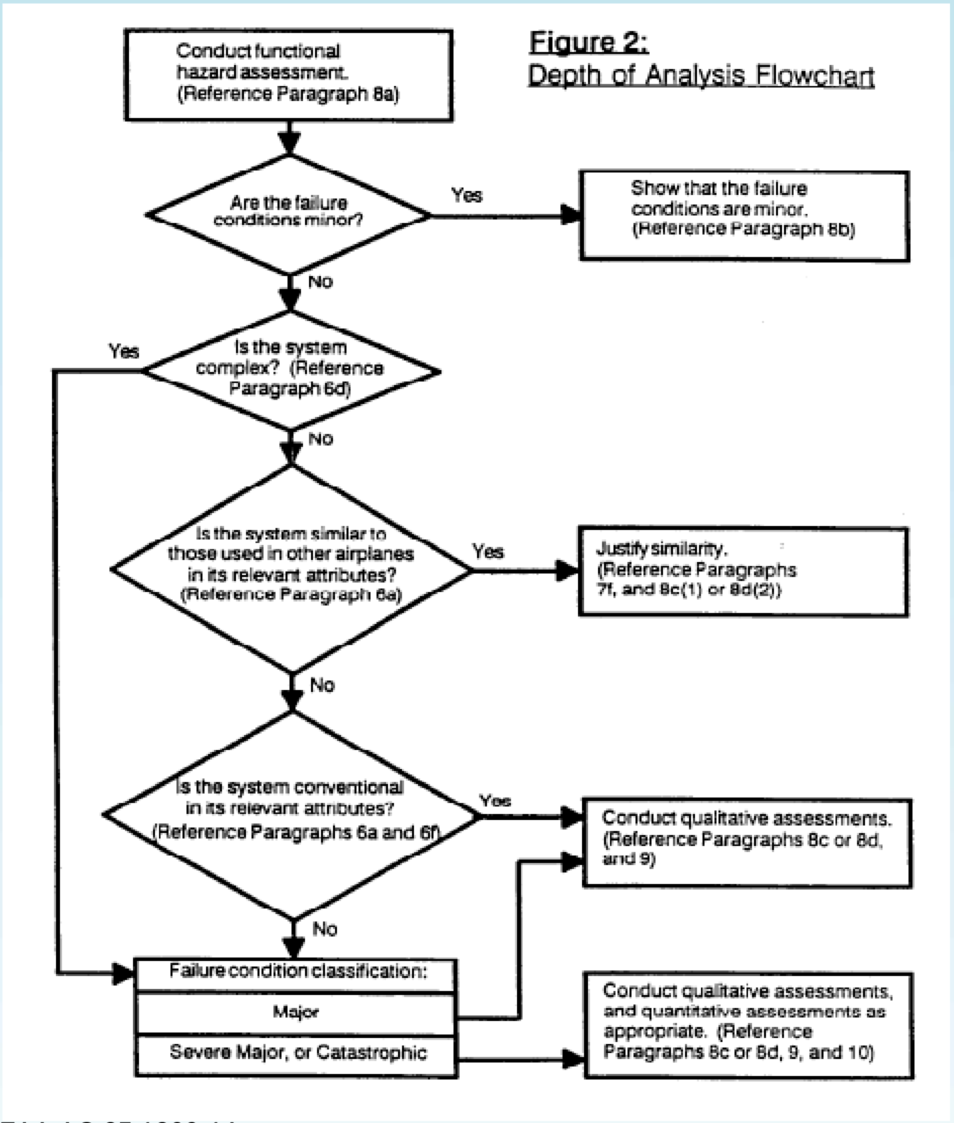


Figure 2:
Depth of Analysis Flowchart
Conduct functional hazard assessment. (Reference Paragraph 8a)
Are the failure conditions minor?
Yes
Show that the failure conditions are minor. (Reference Paragraph 8b)
No
Is the system complex? (Reference Paragraph 6d)
Yes
No
Is the system similar to those used in other airplanes in its relevant attributes? (Reference Paragraph 6a)
Yes
Justify similarity. (Reference Paragraphs 7f, and 8c(1) or 8d(2))
No
Is the system conventional in its relevant attributes? (Reference Paragraphs 6a and 6f)
Yes
Conduct qualitative assessments. (Reference Paragraphs 8c or 8d, and 9)
No
Failure condition classification:
Major
Severe Major, or Catastrophic
Conduct qualitative assessments, and quantitative assessments as appropriate. (Reference Paragraphs 8c or 8d, 9, and 10)

FAA AC 25.1309-1A

Considerations for Complex Systems

- **Simple/conventional/non-complex systems**
  - Can be shown to be implemented correctly by exhaustive testing (def. DO-254) and fail *statistically* with the combined probabilities of their physical constituent parts.
  - Random component failure can be mitigated via redundancy strategies and can be analyzed by methods like FTAs and FMEAs.
- **Complex systems**
  - Systems too large for exhaustive testing (includes almost all SW-based systems)
  - May fail *systematically* due to the increased chance of residual *design errors,* which lead to certain failure under specific (but a priori unknown) operating conditions.
  - No probability numbers can be assigned to „the likelihood of design error“ in a product
  - Lacking accepted methods to demonstrate the absence of design error in a product, current certification regulations require rigorous development process assurance to qualitatively minimize the likelihood of design errors instead.
  - Consequently, increasing development assurance burden (especially w.r.t. **testing**) is assigned with relation to the safety effect of the implemented function in the form of ***Development Assurance Levels*** defined for complex HW and SW subsystems.
  - Process objectives to meet these levels are defined in DO-178B and DO-254

Development Assurance Level Assignment

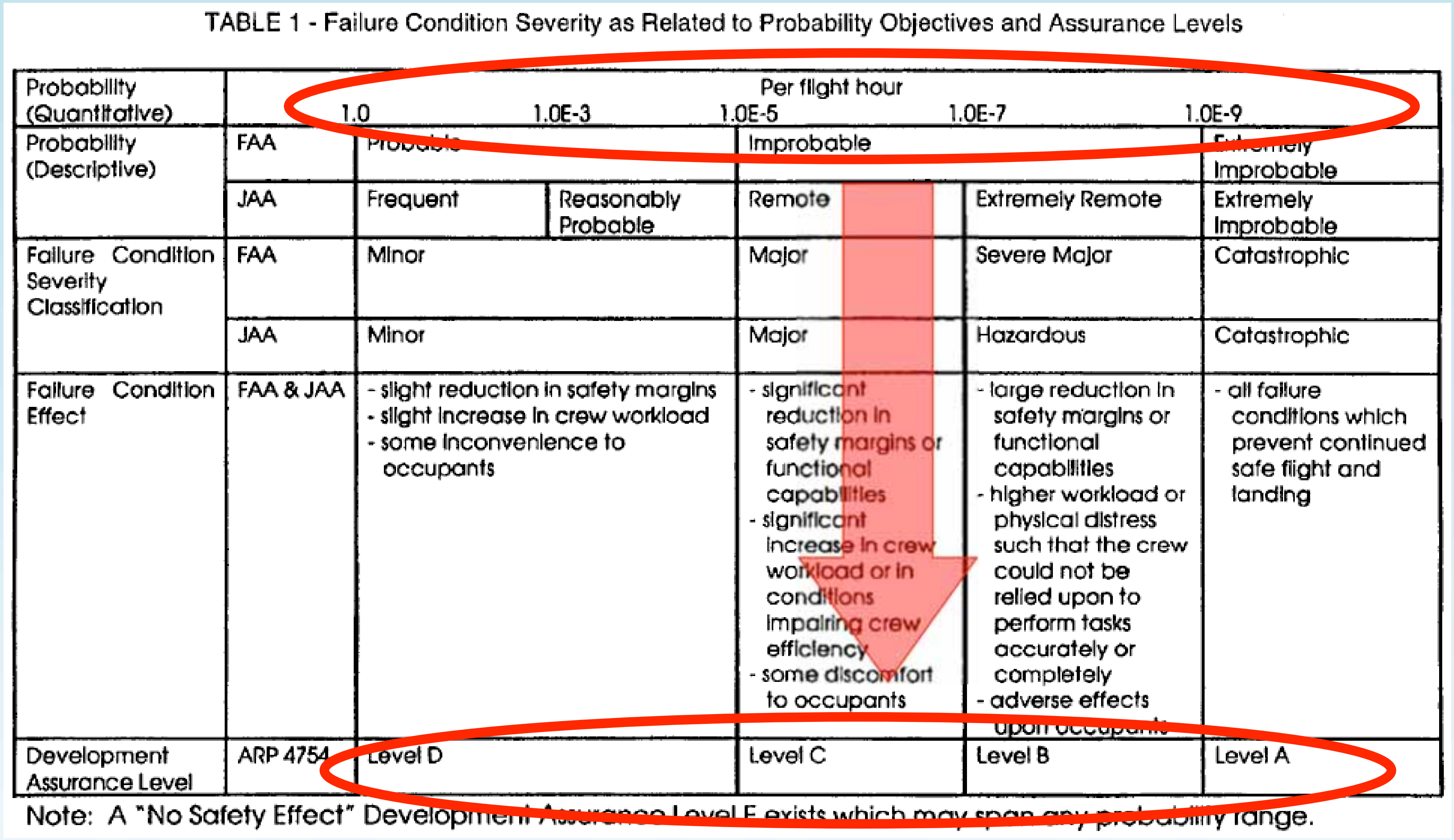


TABLE 1 - Failure Condition Severity as Related to Probability Objectives and Assurance Levels

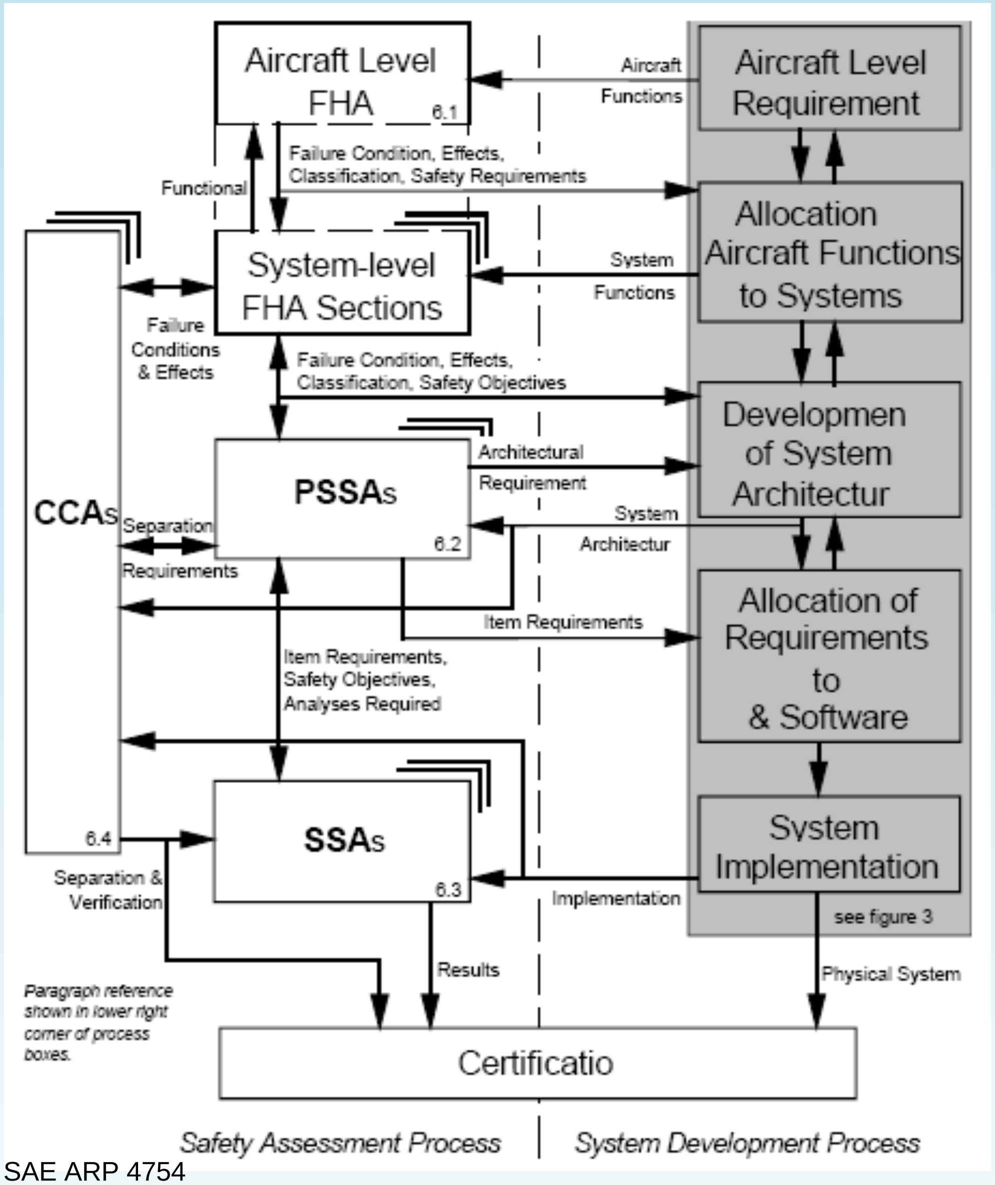
Probability (Quantitative)	Per flight hour 1.0	1.0E-3		1.0E-5	1.0E-7	1.0E-9
Probability (Descriptive)	FAA	Probable		Improbable		Extremely Improbable
	JAA	Frequent	Reasonably Probable	Remote	Extremely Remote	Extremely Improbable
Failure Condition Severity Classification	FAA	Minor		Major	Severe Major	Catastrophic
	JAA	Minor		Major	Hazardous	Catastrophic
Failure Condition Effect	FAA & JAA	- slight reduction in safety margins - slight increase in crew workload - some inconvenience to occupants		- significant reduction in safety margins or functional capabilities - significant increase in crew workload or in conditions impairing crew efficiency - some discomfort to occupants	- large reduction in safety margins or functional capabilities - higher workload or physical distress such that the crew could not be relied upon to perform tasks accurately or completely - adverse effects upon occupants	- all failure conditions which prevent continued safe flight and landing
Development Assurance Level	ARP 4754	Level D		Level C	Level B	Level A

Note: A "No Safety Effect" Development Assurance Level E exists which may span any probability range.

SAE ARP 4761

Development Process of Complex Systems

- Iterative analysis with **Functional Hazard Assessment (FHA)**
  - Determines severity of failures
- Development of **System Architecture**
  - Allocation, Redundancy, Partitioning, ...
- **Common Cause Analysis (CCA)**
- **Preliminary System Safety Assessment (PSSA)** of design, iteratively (top-down)
  - Determines Safety Requirements and
  - Development Assurance Levels
- Allocation of requirements to **hardware** and **software items**
- HW/SW item development according to DO-254 and DO-178B, respectively
- ***System Safety Assesments* (SSAs)** analyze implementation (bottom-up)


Aircraft Level FHA
6.1
Aircraft Functions
Aircraft Level Requirement
Functional
Failure Condition, Effects, Classification, Safety Requirements
Allocation Aircraft Functions to Systems
System-level FHA Sections
System Functions
Failure Conditions & Effects
Failure Condition, Effects, Classification, Safety Objectives
Developmen of System Architectur
CCAs
PSSAs
6.2
Architectural Requirement
System Architectur
Separation Requirements
Item Requirements
Allocation of Requirements to & Software
Item Requirements, Safety Objectives, Analyses Required
SSAs
6.3
6.4
Separation & Verification
Implementation
System Implementation
see figure 3
Results
Physical System
Paragraph reference shown in lower right corner of process boxes.
Certificatio
Safety Assessment Process
System Development Process

SAE ARP 4754

RTCA DO-178B

- Industry consensus document for software certification in 3rd edition (1992), recognized by the international certification authorities as *applicable means of compliance*.
- Suggests software life cycle processes, but does *not* prescribe a preferred software life cycle. Rather a set of seperate processes which may be implemented in most life cycles is presented.
- Defines explicit process objectives to be met for each life cycle process.
- DO-178B compliant life cycles generally create evidence, that the objectives of each software process have been met, are verifiable and can be reproduced.
- DO-178B recognizes *Development Assurance Levels* labeled from A (highest) to D (lowest), where certain process objectives are waived with decreasing level.
- Processes may be automated, but tools used may require tool qualification, too. (distinction between Development Tools and Verification Tools)

DO-178C Topics

- Tool Qualification
- Model Based Design and Verification
- Object-Oriented Technology
- Formal Methods
- Safety and CNS Related Considerations

RTCA DO-254

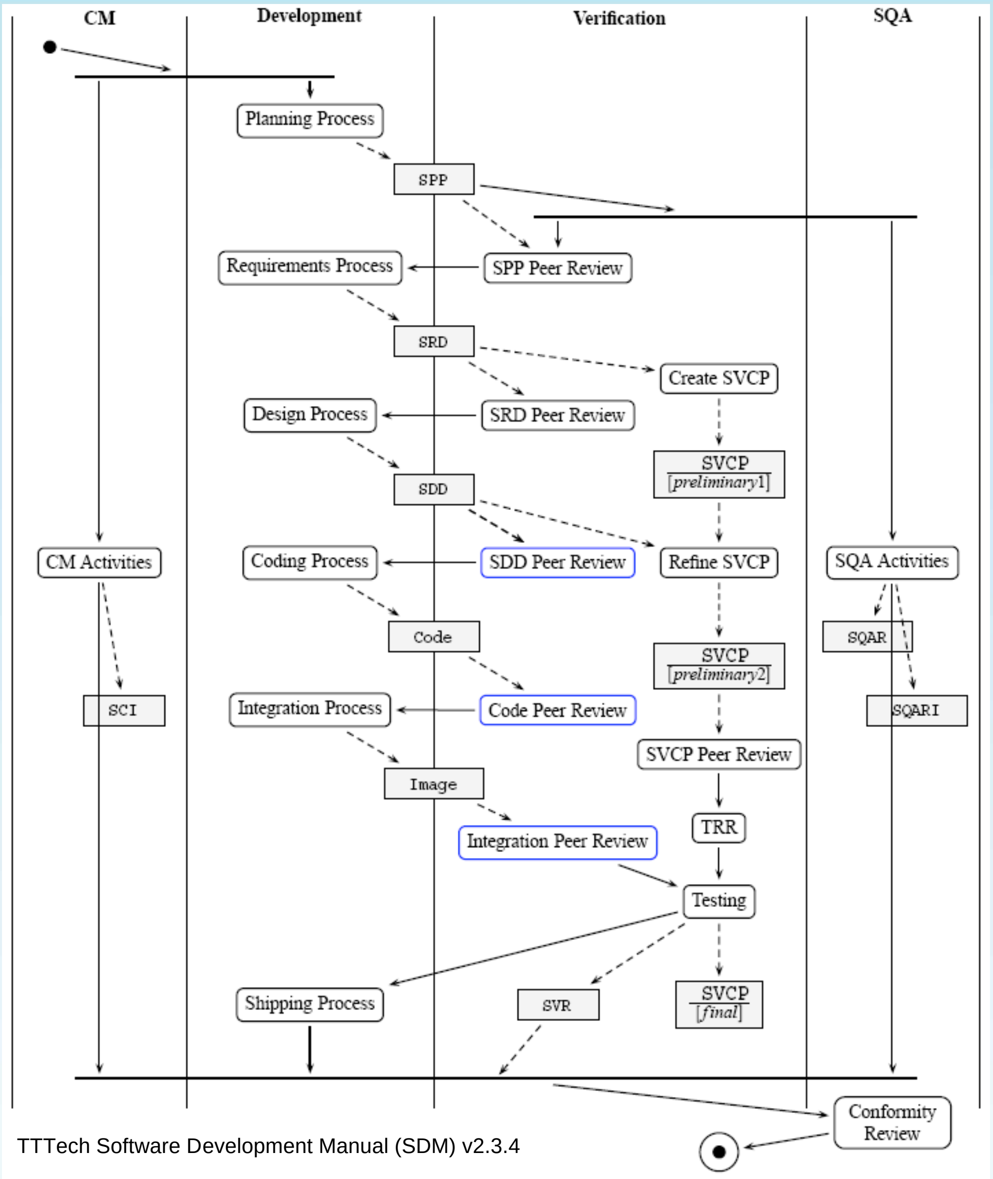
- Industry consensus document on complex hardware development in 1st edition (2000).
- Only recently recognized by the internation certification authorities as applicable means of compliance.
- Counterpart of DO-178B for hardware certification and *very* similar in spirit.
- DO-254 shares many process objectives with DO-178B

Hardware/Software Life Cycle

- DO-254/DO-178B compliant HW/SW life cycles should comprise of the following processes:
  - **Planning Process**
  - **Development Process**
    - **Requirements Process**
    - **Conceptual Design Process**
    - **Detailed Design (HW) / Coding Process (SW)**
    - **Implementation (HW) / Integration Process (SW)**
  - **Verification and Validation Process**
    - **Reviews and Analyses**
    - **Testing Process**
  - **Configuration Management Process**
  - **Process/Quality Assurance Process**
  - **Certification Liaison Process**

Integral Processes

Example: TTTech´s Software Life Cycle



TTTech Software Development Manual (SDM) v2.3.4

- Each development process creates an artifact as output (documents or code).
- *Software Verification Cases and Procedures (SVCP)* are developed in parallel to the refinement steps of the development process.
- All development, planning and verification artifacts are *peer reviewed* prior to release.
- The Testing Process creates the *Software Verification Results (SVR)* as objective evidence for the correct implementation of all high- and low-level requirements.

What is driving cost in Design Assurance Levels?

TTTech

Higher Design Assurance Levels require more testing

- Level D: Tests for all high-level requirements required
- Level C: Tests for all low-level requirements, **Statement Coverage** required
- Level B: Like Level C, but **Decision Coverage** required
- Level A: Like Level B, but **Modified Condition/Decision Coverage** required

This is true for SW (DO-178B), similar criteria hold for complex HW (DO-254)

Cost can be constrained by

- Limiting code complexity
  - smaller state space, fewer configuration options => less if‘s, less testing
- Partitioning system architectures
  - lower DALs for less critical components of the system
- Re-use of “partially accepted“ components designed for an IMA system

Integrated Modular Avionics (IMA)

Motivation for IMA

- Functional integration for improved services
- Reduced number boxes and cabling to save weight
- Hardware transparency and obsolescence support
- upgradeability

- IMA: Integrated Modular Avionics
  - Boeing 777 first plane to deploy IMA implemented by Honeywell, 1992 (SafeBus)
  - TSO C153 (Type Standard Order for re-usable Hardware) first FAA acknowledgement of IMA
  - SC 200 WG 60 design and development guidance for IMA

Certification Aspects of IMA

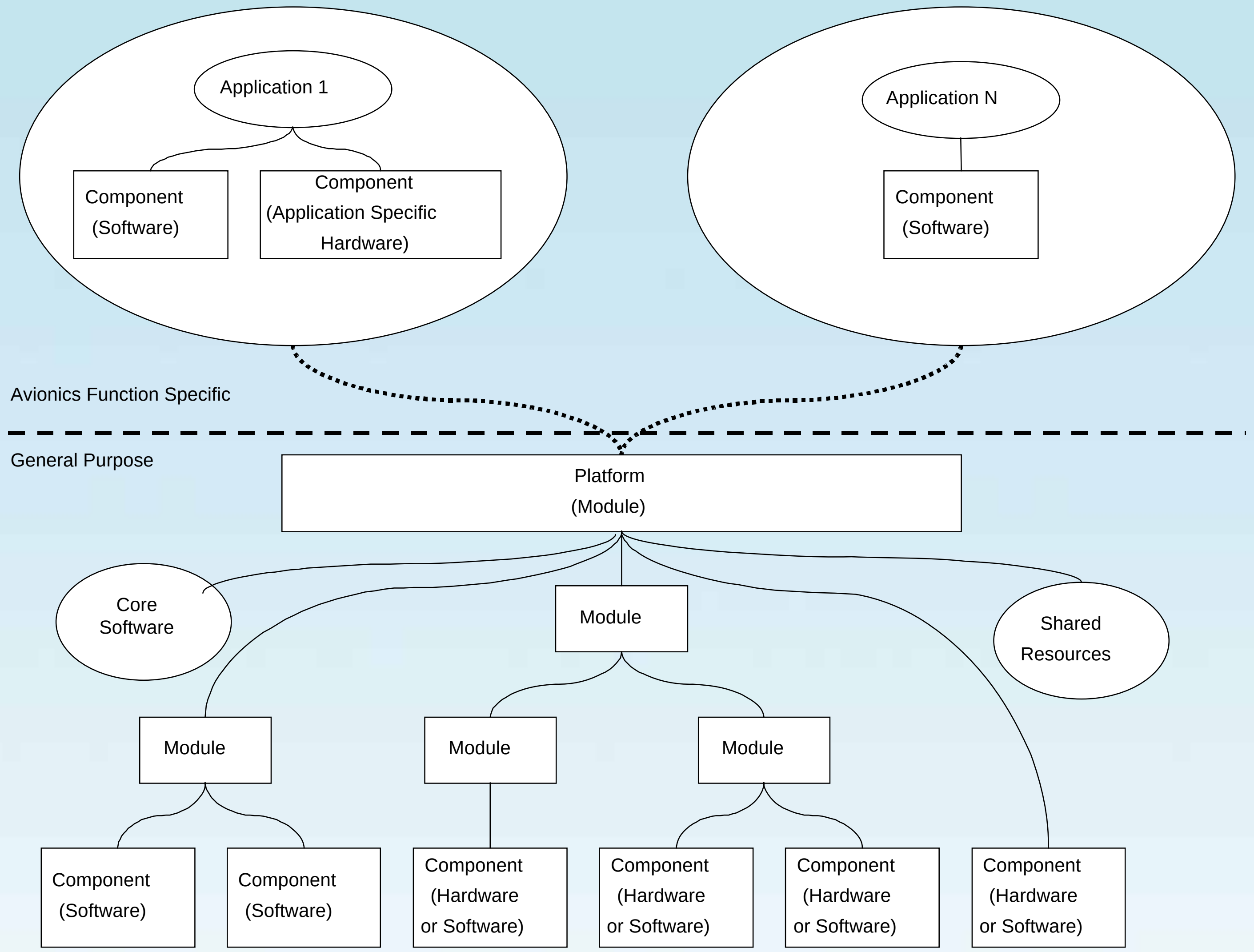
“IMA is a shared set of flexible, reusable, and interoperable hardware and software resources that, when integrated, form a platform that provides services, designed and verified to a defined set of safety and performance requirements, to host applications performing aircraft functions.”

(DO-297/ED-124 Glossary Definition of IMA)

- Current certification regulations only allow for certification of an aircraft as a complete system. Only few subsystems (engines and propellers) are accepted independently without reference to any specific aircraft type.
- Reuse of e.g. hardware or software between aircraft types is only informal, ad-hoc, and requires re-inspection and potential amendments at every instance.

IMA System Structure

TTTech


Application 1
Component (Software)
Component (Application Specific Hardware)
Application N
Component (Software)
Avionics Function Specific
General Purpose
Platform (Module)
Core Software
Module
Shared Resources
Module
Module
Module
Component (Software)
Component (Software)
Component (Hardware or Software)
Component (Hardware or Software)
Component (Hardware or Software)
Component (Hardware or Software)

IMA Key Characteristics

Platform Characteristics

- Platform resources are shared by multiple applications
- An IMA platform autonomously provides robust partitioning of shared resource
- An IMA platform only allows hosted applications to interact with the platform and other hosted applications through well defined interface
- Shared IMA platform resources are configurable to support reuse of the platform

Application Characteristics

- An application may be designed independent of other applications and obtain incremental acceptance on the IMA platform independently of other applications
- Applications can be integrated onto a platform without unintended interactions with other hosted applications
- Applications may be reusable
- Applications are independently modifiable

IMA Key Characteristics

Shared Resources

- Each shared resource has the potential to become a single point of failure that can affect all applications using that resource. Accordingly, an IMA platform has to apply appropriate mitigation techniques as determined by the system safety assessment process.

Robust Partitioning

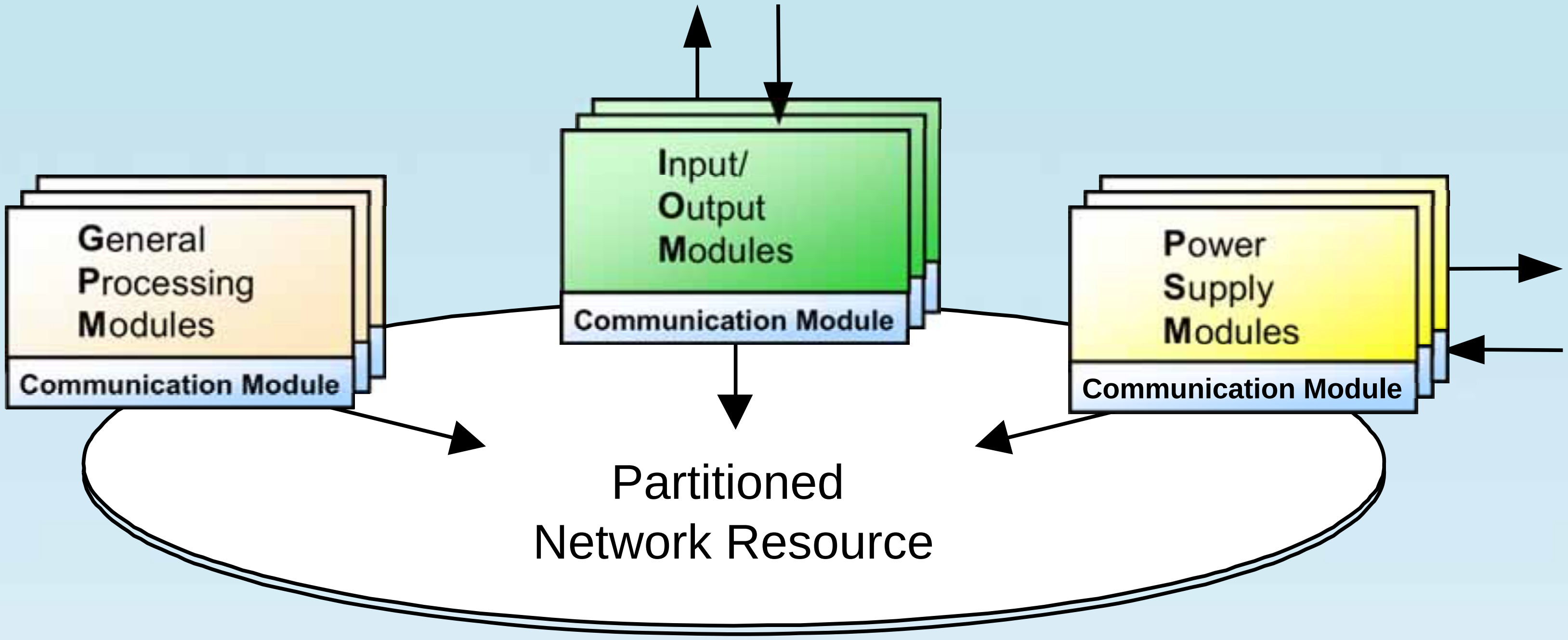
- A mechanism for assuring the intended isolation of independent aircraft functions and applications residing in IMA shared resources in the presence of design errors and hardware failures that are unique to a partition or associated with application specific hardware.

Health Monitoring and Fault Management

- The IMA platform provides health monitoring and fault management capabilities for the platform and hosted applications. The IMA system may have to provide higher level (aircraft function) health monitoring and fault management capabilities to support availability and integrity requirements.

Example: Time-Triggered Architecture

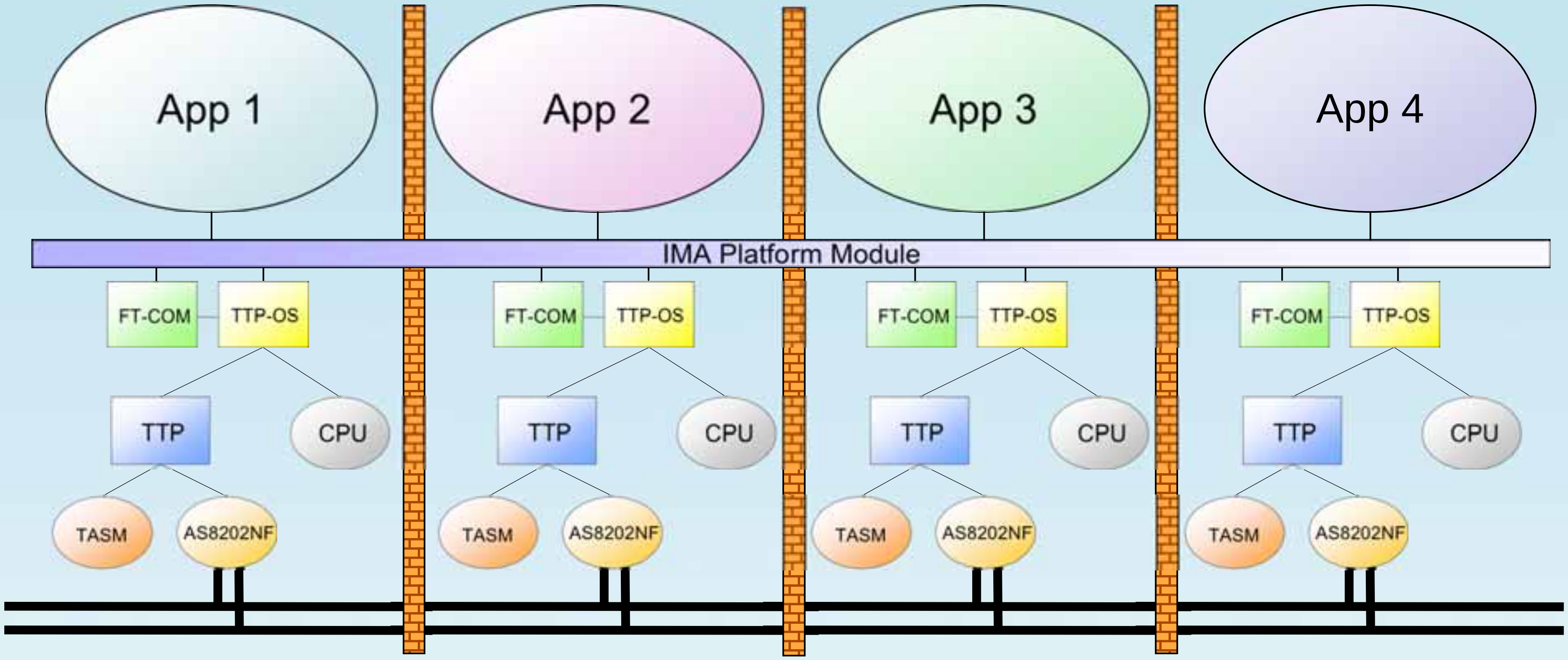
TTTech



- **Processing Modules (LRUs):** Multitude of *general-purpose LRUs* on which applications can be integrated to perform aircraft functions. May contain application-specific hardware.
- **Communication Modules:** *TTP communication controllers* robustly partition access to the shared network resource. That resource may either be a dual-redundant bus- or star-topology. The star topology may include an additional *Central Bus Guardian module* to provide even stronger partitioning and fault-containment claims (not shown).
- **IMA Platform:** The integration of (a subset of) these modules forms a general-purpose IMA Platform

Example: TTA as IMA Platform

TTTech



- **TTP-OS** is a reusable software module which *integrates* on the TTP Controller Module and a CPU component. It executes a static task schedule in synchrony with the global reference time provided by the TTP Controller Module.
- The **FT-COM** module is implemented as local tasks in TTP-OS. It reduces redundant messages from replicated applications to single, agreed values before they are presented to the local application.
- **Robust Partitioning** between applications is ensured by the static TDMA bus access schedule *enforced* by the *TTP Controller Module* (or, optionally, Central Guardian Modules)

DO-297: Architecture and Certification

Modular platforms enable

- Reuse of pre-certified (“partially accepted“) components
- Platform-level service guarantees
- Shared resources
- Fault-Containment / Robust Partitioning
    - lower Design Assurance Levels for less critical components

Two levels of IMA platforms

- Centralized IMA - CPU level partitioning (e.g. ARINC 653 compliant OS)
- Distributed IMA – Network level partitioning (time-triggered communication)

Partitioned OS vs. Multi-Core SoCs

TTTech

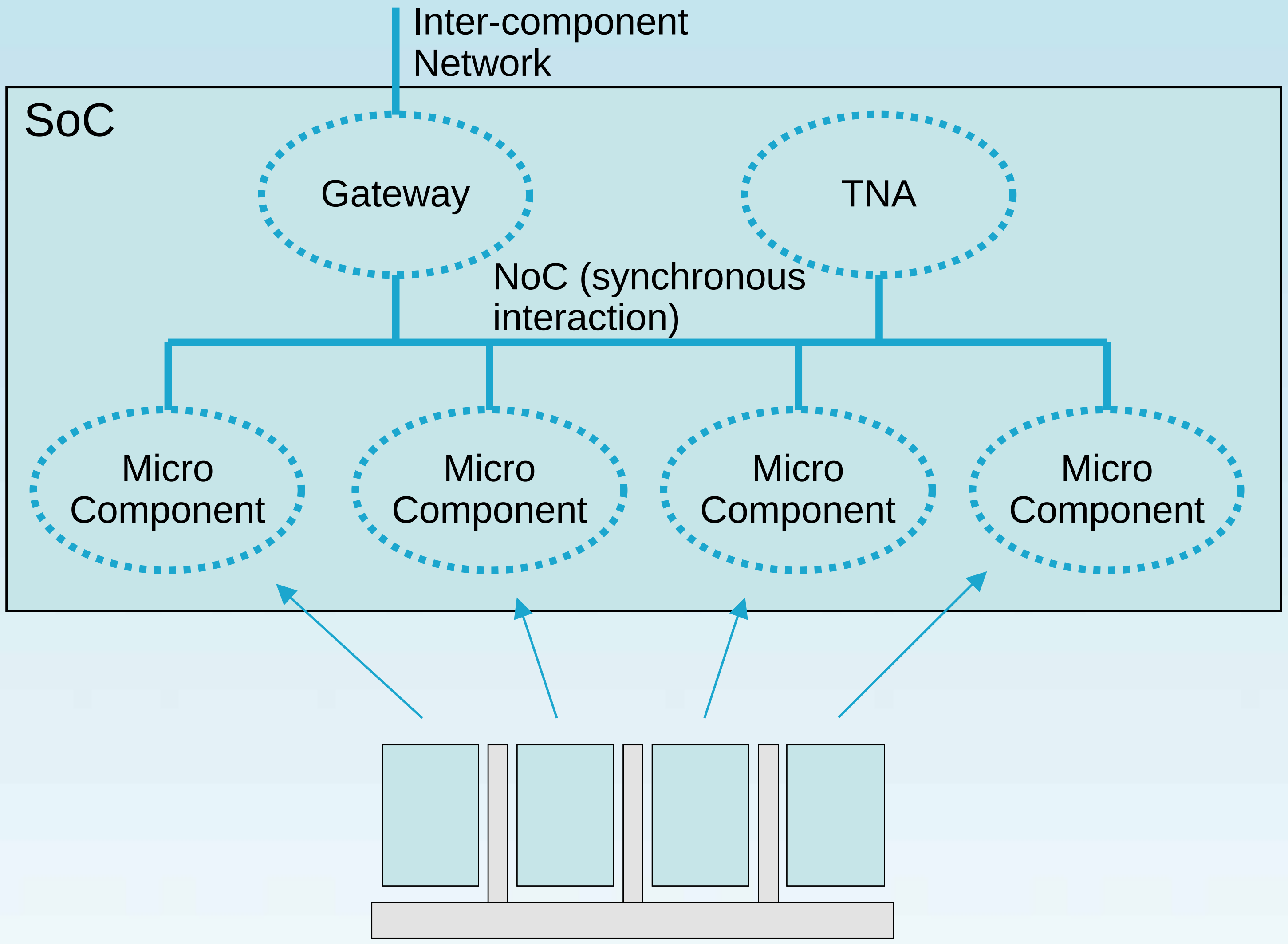
A case contra partitioned OSes…

- Hardware is inherently parallel
- Chip designers exploit hardware parallelism for speeding up single-threaded software (pipelining, caches, out-of-order speculative execution, etc…)
- But most real-time software is multi-threaded
- Partitioned OS are designed to share the CPU among multiple applications
- Eliminating any unintended interactions among partitions requires the OS to defeat all those HW features designed to parallelize execution!

… and pro Multi-Core SoCs

- We suggest to rather utilize transistors better by implementing many simple CPU cores on one die for natural parallel execution and partitioning
- A time-triggered network-on-chip (NoC) can be used for deterministic interaction of on-chip components

TT Multi-Core SoC



Evolution of Certification Requirements

- **Certification guidance gets updated an more constraining in turn**
- **1985-2005**
  - DO-178B required for SW, but no requirements for complex hardware
    - ➢ trend to complex HW observable
  - Relaxed certification requirements for verification tools
    - ➢trend to table-driven SW with table verification tools observable
- **2005-now**
  - DO-178B required for SW
  - DO-254 for HW, with exceptions for COTS components and CPUs
    - ➢ trend to COTS and SoCs observable
- **More strict guidance on COTS and SoCs to be expected**
  - AC 20-156 on Databus Qualification requires DO-254/DO-178B for <u>all</u> COM components

DO-254: Product Service Experience

Can service history substitute design assurance?

- only acceptable if similar application, function, and environment
- only acceptable if used in application of same level of criticality
- requires assessment of all design errors found during service period
- requires evidence for actual failure rates in operation

Can automotive help to with service history?

- Example: FlexRay for DO-254 Level A applications?
- safety-critical x-by-wire automotive application are unlikely before 2012-15
- Different SoC‘s, upgrades typical every 5 years
- To get 10^9 h service history we need to have the production quatity of the VW Golf for 5 years

DO-254: Reverse Engineering

Possible under certain preconditions
we‘ve done it for a stand-alone communication controller!

- Only reasonably possible with access to original developers
- Hard to extract low-level/high-level requirements at reasonable level of abstraction (need to understand what the code is *supposed* to be doing)
- Hard to establish satisfactory traceability between requirements/source code
- Hard to justify *derived* requirements (why have certain decisions been made?)
- Hard to argue unneeded functionality (dead or deactivated code?)

(Ref: Position paper CAST-18: “Reverse Engineering in Certification Projects“)

SoC‘s

- For a SOCs these challenges are mouch tougher
- Is it feasible from a cost perspective to do reverse certification of a SoC
- Is it more cost effective to forward-engineer an FPGA?

Thank You!