

DOCUMENT

Document /
Deliverable Name: **Analysis and prioritization of ex-
ploitable cross-domain services
with focus on aerospace**

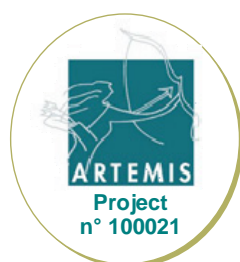
Document /
Deliverable Nr. **D 3.1**

Version: **draft** **1.0**
 final

Document Type: **confidential**
 public

Responsible: **Christian Fidi, TTTech**

Date of creation:: **30 December 2009**
Last modification **30 December 2009**



List of INDEXYS Beneficiaries

No	Name	Short	Country
01	TTTech Computertechnik AG	TTT	Austria
02	AUDI AG	AUDI	Germany
03	Delft University of Technology	DUT	Netherlands
04	EADS Deutschland GmbH	EADS-IW	Germany
05	NXP Semiconductors Netherlands B.V.	NXP-NL	Netherlands
06	OptXware Research and Development Ltd.	OPT	Hungary
07	Thales Rail Signalling Solutions GmbH	TRSS-AT	Austria
08	Technical University of Darmstadt	TUDA	Germany
09	Technical University of Kaiserslautern	UNIKL	Germany
10	Vienna University of Technology	TUVI	Austria

Author(s)

Name	Company
Christian Fidi	TTT
Stefan Schneelee	EADS-IW

Project Coordination

TTTech Computertechnik AG

Schoenbrunner Strasse 7
1040 Vienna, Austria

Technical Matters:

D.I. Andreas ECKEL, MBA
Email: andreas.eckel@tttech.com
Tel: +43 1 585 34 34 – 16
Fax: +43 1 585 34 34 – 90

Financial Matters:

D.I. Andreas BAUMGARTNER
Email: andreas.baumgartner@tttech.com
Tel: +43 1 585 34 34 – 942
Fax: +43 1 585 34 34 – 90

Copyright 2009: The INDEXYS Consortium
www.indexys.eu

Revision chart and history log

Version	Date	Reason
0.1	2009-09-29	Initial Version
0.2	2009-10-08	Adding RDC requirements and features
0.3	2009-12-13	Mapping of the features to GENESYS
0.4	2009-12-14	Adding Introduction
1.0	2009-12-30	Table with requirements references added, typos

Table of Contents

1. INTRODUCTION.....	6
2. REQUIREMENTS OF AEROSPACE DOMAIN.....	7
Real-time Communication Requirements	9
Error containment regions.....	10
Predictability Requirements	12
HW-COM Requirements	12
Redundancy and Reintegration.....	13
3. FEATURES OF DOMAIN SPECIFIC PLATFORM SOLUTION	15
Remote Data Concentrator (RDC)	15
Network Access Controller (NAC).....	20
4. AEROSPACE PLATFORM IN THE GENESYS CONTEXT	25
5. MAPPING OF FEATURES TO GENESYS REFERENCE ARCHITECTURE TEMPLATE.....	27
5.1 Core Architectural Services	28
5.1.1 Basic Configuration Services	28
5.1.2 Basic Execution Control Services	28
5.1.3 Basic Time Services	28
5.1.4 Basic Communication Services.....	29
5.2 Optional Architectural Services.....	29
5.2.1 Diagnostic Services	29
5.2.2 Legacy Integration	29
5.2.3 Reference Table Requirements to Features	30
6. ANNEX A: ABBREVIATIONS	34
7. REFERENCES	35

List of Figures

Figure 2-1 Hardware Life Cycle Data by Hardware Design Assurance Level and Hardware Control	8
Figure 3-1 TD-COM Layer overview	16
Figure 3-2 Generic LRU with HW-COM	17
Figure 3-3 Cabin communication network architecture	20
Figure 3-4 Block diagram of Network Access Controller (NAC).....	21
Figure 5: Supported Core and Optional Architectural Services	27

List of Tables

Table 1: Cross Reference Table for the Core Services	30
Table 2: Cross Reference Table for the Optional Services (Part 1).....	31
Table 3: Cross Reference Table for the Optional Services (Part 2).....	32
Table 4: Cross Reference Table for the Optional Services (Part 3).....	33

1. Introduction

In Integrated Modular Avionics (IMA), distributed functional computations are centralized on a group of Central Processing Modules with powerful CPUs interfacing small, easy to maintain and reliable Remote Data Concentrators. This state-of-the-art platform concept is widely used for safety-critical control functions of the airplane, but currently not for cabin control applications.

The objective of this document is to define the requirements of the aerospace domain with focus to the Remote Data Concentrator (RDC) and the Network Access Controller (NAC) for cabin control applications. Furthermore the features of the RDC and the NAC are described and a mapping to the GENESYS reference architecture will be illustrated.

2. Requirements of Aerospace domain

The requirements in aerospace are not only driven by performance and costs, but in particular by safety. Additionally, main drivers are the reduction of weight and complexity by introducing new concepts and mechanism on architecture level. Network requirements are not driver, but should support architectural design considerations. The safety requirements are enforced by certification authorities like the EASA or FAA. Depending on the safety assessment process and the size and utilization of an aircraft, several standards apply. The best known standards are the DO-254 for Hardware, DO-178B for Software and the DO-160 for environmental considerations. Aircraft functions are categorized into so called Design Assurance Level (DAL), having DAL A as highest and DAL E as lowest. The categories represent the effects of the failure conditions on the aircraft, crew, and passengers. Hardware and Software that support DAL A and DAL B functions, which means a failure condition could lead to catastrophic or hazardous scenarios, are under very restrictive design and process limitations. Furthermore, an extensive V+V process applies. Examples of necessary steps and documentation can be extracted from the next table (source RTCA DO254) called "Hardware Life Cycle Data by Hardware Design Assurance Level and Hardware Control".

Requirements concerning digital communication are not consistently defined. Traditional and well proven point to point data-buses (e.g. A 429, RS 232, RS 485) are covered by traditional validation and verification principles. For the Avionic Full Duplex Ethernet (AFDX) a specific certification review item (CRI) is valid. All other data buses depending of the criticality of the functions they may support, the requirements should address at least the following issues:

- Safety
- Data integrity
- Performance
- Design assurance
- Electromagnetic compatibility
- Verification and validation
- System configuration management
- Continued airworthiness.

Data Section	Hardware Life Cycle Data ①	Objectives ②	Submit	Level A	Level B	Level C	Level D
10.1	Hardware Plans						
10.1.1	Plan for Hardware Aspects of Certification	4.1(1,2,3,4)	S	HC1	HC1	HC1	HC1
10.1.2	Hardware Design Plan	4.1(1,2,3,4)		HC2	HC2	HC2	NA
10.1.3	Hardware Validation Plan ③④	4.1(1,2,3,4); 6.1.1(1)		HC2	HC2	HC2	NA
10.1.4	Hardware Verification Plan	4.1(1,2,3,4); 6.2.1(1)	S	HC2	HC2	HC2	HC2
10.1.5	Hardware Configuration Management Plan	4.1(1,2,3,4); 7.1(3)		HC1	HC1	HC2	HC2
10.1.6	Hardware Process Assurance Plan	4.1(1,2,4); 8.1(1,2,3)		HC2	HC2	NA	NA
10.2	Hardware Design Standards						
10.2.1	Requirements Standards ③	4.1(2)		HC2	HC2	NA	NA
10.2.2	Hardware Design Standards ③	4.1(2)		HC2	HC2	NA	NA
10.2.3	Validation and Verification Standards ③	4.1(2)		HC2	HC2	NA	NA
10.2.4	Hardware Archive Standards ③	4.1(2); 5.5.1(1); 7.1(1,2)		HC2	HC2	NA	NA
10.3	Hardware Design Data						
10.3.1	Hardware Requirements	5.1.1(1,2); 5.2.1(2); 5.3.1(2); 5.4.1(3); 5.5.1(1,2,3); 6.1.1(1,2); 6.2.1(1)		HC1	HC1	HC1	HC1
10.3.2	Hardware Design Representation Data						
10.3.2.1	Conceptual Design Data ③	5.2.1(1)		HC2	HC2	NA	NA
10.3.2.2	Detailed Design Data	5.3.1(1); 5.4.1(2)		⑤	⑤	⑤	⑤
10.3.2.2.1	Top-Level Drawing	5.3.1(1); 5.4.1(2); 5.5.1(1)	S	HC1	HC1	HC1	HC1
10.3.2.2.2	Assembly Drawings	5.3.1(1); 5.4.1(2); 5.5.1(1)		HC1	HC1	HC1	HC1
10.3.2.2.3	Installation Control Drawings	5.4.1(2); 5.5.1(1)		HC1	HC1	HC1	HC1
10.3.2.2.4	Hardware/Software Interface Data ③	5.3.1(1); 5.5.1(1)		HC1	HC1	HC1	HC1
10.4	Validation And Verification Data						
10.4.1	Hardware Traceability Data	6.1.1(1); 6.2.1(1,2)		HC2	HC2	HC2 ⑥	HC2 ⑥
10.4.2	Hardware Review and Analysis Procedures ③	6.1.1(1,2); 6.2.1(1)		HC1	HC1	NA	NA
10.4.3	Hardware Review and Analysis Results ③	6.1.1(1,2); 6.2.1(1)		HC2	HC2	HC2	HC2
10.4.4	Hardware Test Procedures ③	6.1.1(1,2); 6.2.1(1)		HC1	HC1	HC2	HC2 ⑦
10.4.5	Hardware Test Results ③	6.1.1(1,2); 6.2.1(1)		HC2	HC2	HC2	HC2 ⑦
10.5	Hardware Acceptance Test Criteria	5.5.1(3); 6.2.1(3)		HC2	HC2	HC2	HC2
10.6	Problem Reports	5.1.1(3); 5.2.1(3); 5.3.1(3); 5.4.1(4); 5.5.1(4); 6.1.1(3); 6.2.1(4); 7.1(3)		HC2	HC2	HC2	HC2
10.7	Hardware Configuration Management Records	5.5.1(1); 7.1(1,2,3)		HC2	HC2	HC2	HC2
10.8	Hardware Process Assurance Records	7.1(2); 8.1(1,2,3)		HC2	HC2	HC2	NA
10.9	Hardware Accomplishment Summary	8.1(1,2,3)	S	HC1	HC1	HC1	HC1

Figure 2-1 Hardware Life Cycle Data by Hardware Design Assurance Level and Hardware Control

Real-time Communication Requirements

ID	Req-Aero-1
Description	Small protocol latency and minimal jitter: In order to achieve fault isolation in the temporal domain, the TTP nodes ensure that message transmissions comply with specified minimum message latency and jitter.
Importance	High
Implementation Hints	-

ID	Req-Aero-2
Description	Simultaneous delivery in multicast: Transmissions arrive at the receivers in the same order and at (within the precision) the same time also in dual-channel networks.
Importance	High
Implementation Hints	-

ID	Req-Aero-5
Description	Error detection within a node: A node must detect all internal failures within a short latency, and must map these failures to a single external failure mode, a fail-silent node failure. The error detection in the node must be concerned with value failures and timing failures.
Importance	High
Implementation Hints	-

ID	Req-Aero-6
Description	Legacy TD-COM support: There are already existing TTP-Nodes with TD-COM implementation. The TTP Hardware communication layer nodes must also be able to communicate with TD-COM nodes in one network. It shall be possible to create configurations for TD-COM and HW-COM nodes within the same network, using the TTP-Tools.
Importance	High
Implementation Hints	-

Error containment regions

Provided that the components of a properly configured TTP based system are in different fault containment regions, each can fail in an arbitrary way. Under this assumption the probability of two concurrent independent component failures is small enough to be considered a rare event that can be handled by an appropriate never-give-up (NGU) strategy. However, it should be noted that a very prompt error detection mechanism is needed to ensure that two consecutive single faults are not becoming concurrent.

As for hardware faults, TTP is designed to isolate and tolerate single node faults. By introducing a bus-guardian it is guaranteed that a faulty node cannot prevent correct nodes from exchanging data. The bus guardian ensures that a node can only send once in a TDMA round, thereby eliminating the problem of babbling idiots that monopolize the communication medium.

Moreover, TTP implements a never-give-up strategy (NGU) for multiple fault scenarios: if a node detects faults that are not covered by the fault hypothesis, it notifies the application. The application may now decide either to shutdown in fail-safe environments or to restart in fail-operational environments with an agreed consistent state among all nodes of the distributed system.

These mechanisms ensure fault tolerance at the communication subsystem level in TTP. These mechanisms of the communication subsystem guarantee that faulty nodes cannot prevent correct nodes from communicating and serve as communication platform for the application. At the application level fault tolerance needs to be implemented by a fault tolerance layer and an appropriate application design. Fault tolerance can be realized by replicating a software subsystem on two fail-silent nodes. Tolerance of a single arbitrary node failure can be ensured by TMR (Triple Modular Redundancy) voting.

Both mechanisms will tolerate single component faults with the respective failure semantics and are thus fit to handle both transient and permanent hardware faults. To re-establish tolerance to single component faults despite the presence of a permanent hardware fault, TTP supports the implementation of transparent hot stand-by spares.

Fault tolerance is realized by a redundant unit in the network taking over the function of a defective unit, without being noticed at the function level. This is why data consistency is necessary.

ID	Req-Aero-7
Description	Controller State: Every TTP-Node in the network shall check the C-state consistency and the membership. The controller state (C-state) consists of the global time, the message descriptor list position, some mode change bits, and the membership vector of the nodes. By continuously exchanging and checking the C-state (including the membership), all controllers can detect if they performed identical updates to their C-state.
Importance	High
Implementation Hints	-

ID	Req-Aero-8
Description	Implicit Acknowledgement: The Acknowledgement of TTP takes advantage of the broadcast facility of the communication medium. It is known a priori that every correct member of the ensemble hears every message transmitted by a correct sender. As soon as one receiver has acknowledged a message from a sender, it can be concluded that the message has been sent correctly and that all correct receivers have received it.
Importance	High
Implementation Hints	-

ID	Req-Aero-9
Description	Clique Detection: TTP continuously performs C-state agreement and implicit acknowledgement. But this is not sufficient for a special class of faults, where several nodes come into disagreement at once (they build a “clique”). Clique detection allows detecting a fault where potentially no node is faulty, but the system may still fail due to inconsistency.
Importance	High
Implementation Hints	-

ID	Req-Aero-10
Description	Bus Guardian: A node that sends messages at the wrong moment in time is called babbling idiot. Babbling idiot timing failures are the most serious failure of a node in a system with shared communication channel, such as in a bus system. The bus guardian monitors the temporal access pattern of the TTP controller to the replicated buses, and terminate the controller operation in case a timing violation in the regular access pattern is detected. The bus guardian ensures a fail-silent behaviour of the node.
Importance	High
Implementation Hints	-

Predictability Requirements

ID	Req-Aero-11
Description	Global time Service: The global synchronized time is a requirement and a feature in time-triggered systems. The global time must be established by a periodic clock synchronization in order to enable a time triggered communication and computation. In TTP the global time is established by a fault tolerant clock synchronization algorithm.
Importance	High
Implementation Hints	-

ID	Req-Aero-12
Description	Static network schedule: In TTP the network schedule is described in the message descriptor list (MEDL). To get a highly predictable network this MEDL has to be static and scheduled offline. The TTP controller uses this internal data structure to know when a message must be sent on or received from the communication channel and it also contains the position in the controller network interface (CNI). During protocol operation, the MEDL serves as a dispatching table for the TTP controller.
Importance	High
Implementation Hints	-

HW-COM Requirements

ID	Req-Aero-13
Description	Asynchronism between Application and HW-COM: The HW-COM Layer shall use double-buffers to support an asynchrony between the application and the HW-COM Layer. Otherwise the application must be synchronized with the communication layer and moreover with the network to avoid blocking buffers.
Importance	High
Implementation Hints	-

ID	Req-Aero-14
Description	Packing and Unpacking Messages: The HW-COM Layer shall pack/unpack messages sent/received from/by the application into/from the frame format used by the CNI.
Importance	High
Implementation Hints	-

ID	Req-Aero-15
Description	Endianess and CNI-Permutation: Endianess and CNI-Permutation shall be configurable to support different types of CPUs.
Importance	High
Implementation Hints	-

ID	Req-Aero-16
Description	Configuration Tables: The configuration for the HW-COM Layer shall be stored in highly configurable configuration tables.
Importance	High
Implementation Hints	-

Redundancy and Reintegration

ID	Req-Aero-17
Description	Active Replication: Active Replication requires replica determinism. The constraints for the replicas are: No dynamic scheduling decisions Same input and output messages Global time base Agreement on replicated readings In TTP all this requirements were implemented.
Importance	High
Implementation Hints	-

ID	Req-Aero-18
Description	Reintegration Service of Replicas: In normal operation of the network the history state (h-state) is transmitted together with the normal messages. During re-integration the h-state is received from the replica. The h-state is transmitted at least once a cluster cycle.

Importance	High
Implementation Hints	-

ID	Req-Aero-19
Description	HW-COM handles multiple TTP-Controller: Active- or Passive-Replication of TTP-Controller per node require a HW-COM that is able to handle data from multiple communication network interfaces (CNIs). The implementation of the HW-COM Layer shall have the ability to handling the data from multiple controllers as well as using the first valid principle.
Importance	High
Implementation Hints	-

3. Features of domain specific platform solution

Remote Data Concentrator (RDC)

The task of a Remote Data Concentrator (RDC) is to interface transducers (i.e., sensors and actuators) in the aircraft. Remote Data Concentrators are connected via communication systems to the Central Processing Modules. To achieve a highly deterministic behaviour for the RDC communication a time triggered communication protocol (TTP) has to be used.

In present the complexity of distributed communication systems is increasing rapidly resulting in huge software modules resulting in huge certification efforts consuming high development and certification costs. To cope with such certification issues designs with static certified source code and application specific configuration tables got very popular in the aerospace domain.

Table Driven Communication Layer (TD-COM)

In the past TTEch developed a table driven communication layer (TD-COM Layer) for TTP in software to make use of the reduced certification effort during reuse. The TD-COM Layer implements a high-performance communication layer between TTP networks and host applications. The TD-COM Layer can support up to two TTP networks, each being connected by a separate TTP controller. To meet these requirements the TD-COM Layer is divided into two sublayers:

- On the TTP network side, there is the Frame-Copy Layer (FCL), which copies data between the CNI of the TTP controller and the frame buffer.
- On the application side, there is the Message-Handling Layer (MHL), which copies data between the frame buffer and the message-handling buffer.

The TD-COM is designed in such a way that the FCL runs synchronously with TTP networks, whereas the MHL can run asynchronously with TTP networks.

The application can make use of the TD-COM Layer by using defined API calls. Configuration data tables generated by the configuration tool provide the TD-COM Layer with all information needed for data handling. A pointer makes reference to those configuration data tables.

As the TD-COM Layer does not contain any global data, the user must provide a global data structure which the TD-COM Layer can reference. This lets the TD-COM Layer store frequently used data, so that data need not be passed as an argument when the API is called. In addition, the user has to reserve the Message Handling Buffer (MHB) and the Frame Copy Buffer (FCB). With the configuration data tables, the TD-COM Layer then makes reference to these buffers.

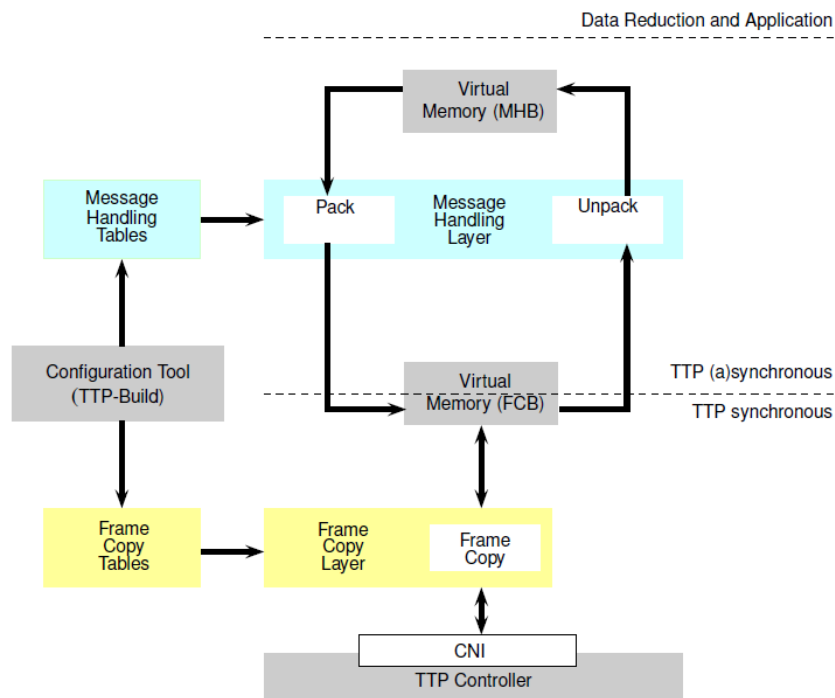


Figure 3-1 TD-COM Layer overview

The TD-COM Layer allocates messages to frames and packs/unpacks them into the application code messages coming from the bus. The TD-COM Layer is written in the C programming language and runs on the host CPU [1].

The TD-COM Layer interfaces with the CNI of the TTP controller and with the TTP Driver, with the latter being required for the TD-COM Layer to function properly. The TD-COM library provides the following services:

- packing/unpacking of messages to/from frames
- endianness handling and CNI permutation

The TD-COM plugin provides an interrupt scheduler that is used instead of the standard TTP-Build task scheduler. The interrupt scheduler is invoked by the Make new schedule command and schedules interrupts for the TD-COM library. The code generator, which is called with the Generate code command, generates the MEDL and the configuration tables for the TD-COM library.

Figure 3-1 TD-COM Layer overview shows the TD-COM layer and its parts in their functional environment. The following sections describe the main constituents in a little more detail.

Table Driven Hardware Communication Layer (HW-COM):

For simple distributed communication nodes as Remote Data Concentrators are a solution with the TD-COM layer requires a lot of CPU power for executing the packing and unpacking of messages. This hints to a TD-COM implementation based on a FPGA or moreover as ASIC. Figure 3-2 Generic LRU with HW-COM shows a line replaceable unit with implemented HW-COM.

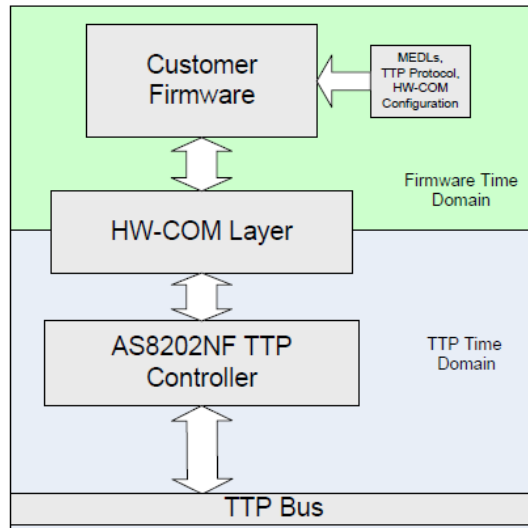


Figure 3-2 Generic LRU with HW-COM

The TTP HW-COM will be developed as an IP module that is part of an FPGA providing the following features:

ID	RDC-Aero-1
Description	Reduce CPU Utilization: Outsourcing the TD-COM activities during runtime reduces the CPU utilization. The packing and unpacking tasks require a lot of performance of the host CPU and furthermore reduce the resources available for the applications.
Importance	High
Implementation Hints	-

ID	RDC-Aero-2
Description	Asynchronism between Application and HW-COM: The HW-COM Layer uses buffer-types that support an asynchrony between the application and the HW-COM Layer.
Importance	High
Implementation Hints	-

ID	RDC-Aero-3
Description	TTP Controller Maintenance: Start-up/Life-sign support for an AS8202NF TTP controller.
Importance	High
Implementation Hints	-

ID	RDC-Aero-4
Description	Data Exchange Timing: Mapping synchronous TTP traffic to asynchronous buffers accessible by the firmware.
Importance	High
Implementation Hints	-

ID	RDC-Aero-5
Description	Customizable HW-COM Tables: Highly customizable configuration tables for user defined message handling.
Importance	High
Implementation Hints	-

ID	RDC-Aero-6
Description	Sender Status Information: Sender status information to indicate non-updated data.
Importance	High
Implementation Hints	-

ID	RDC-Aero-7
Description	High Speed Packing: The HW-COM IP will be much faster than the TD-COM implementation because of the higher clock of the FPGA. This means that data will have lower latencies introduced by packing and unpacking data.
Importance	High
Implementation Hints	-

ID	RDC-Aero-8
Description	Implementation into Legacy Platforms and Applications: A HW-COM Node has a defined interface between the HW-COM and the host CPU that is easy to implement in legacy platforms or application. The TD-COM has to be compiled for other Targets.
Importance	High

Implementation Hints	-
----------------------	---

ID	RDC-Aero-9
Description	Small physical Format and Size: An important criteria for RDCs are the physical Format and the Size of the PCB. With the HW-COM approach the application and the HW-COM can be synthesized in one FPGA. Moreover this IP can easily be used to build an ASIC which enables a solution of a System-on-a-Chip with minimum physical size.
Importance	High
Implementation Hints	-

Network Access Controller (NAC)

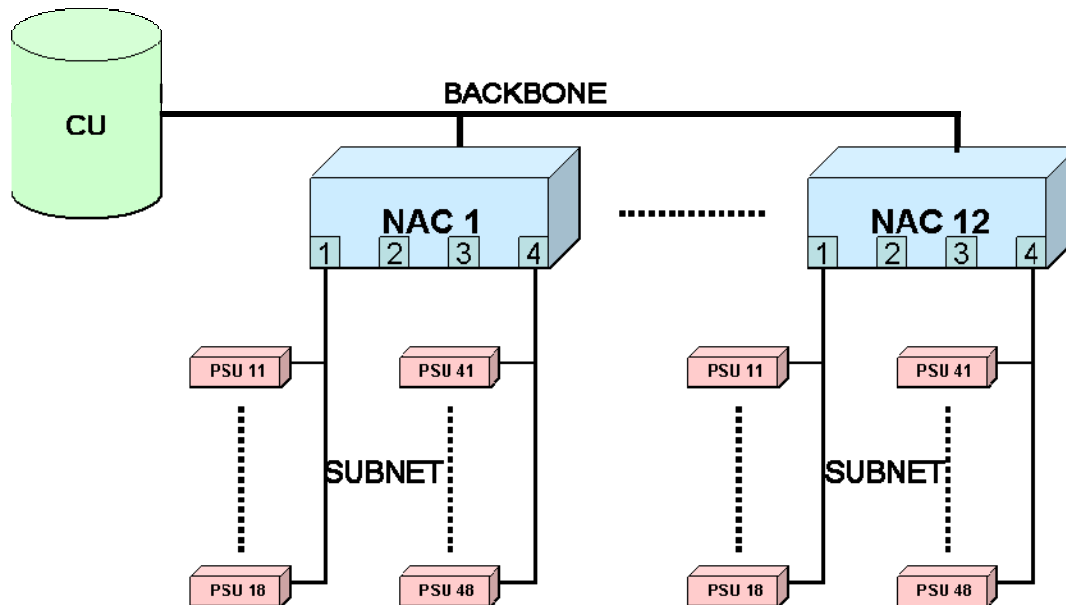


Figure 3-3 Cabin communication network architecture

Figure 2-1 shows the principle structure of a network architecture used to provide connectivity from a central unit like a server to passenger oriented devices like a PSU in an aircraft cabin for cabin control applications. This cabin communication architecture should incorporate all electronic cabin systems.

The network consists of one or more Central Units (CU), Network Access Controllers (NAC) and Passenger Service Units (PSU). A high data rate backbone connects up to 12 NACs with the CU. Each NAC provides at least four subnets for connecting to the network up to eight PSUs per subnet.

With this configuration it is possible to connect 32 passenger oriented devices to one NAC. In total up to 384 passengers oriented devices are possible with the use of 12 NACs.

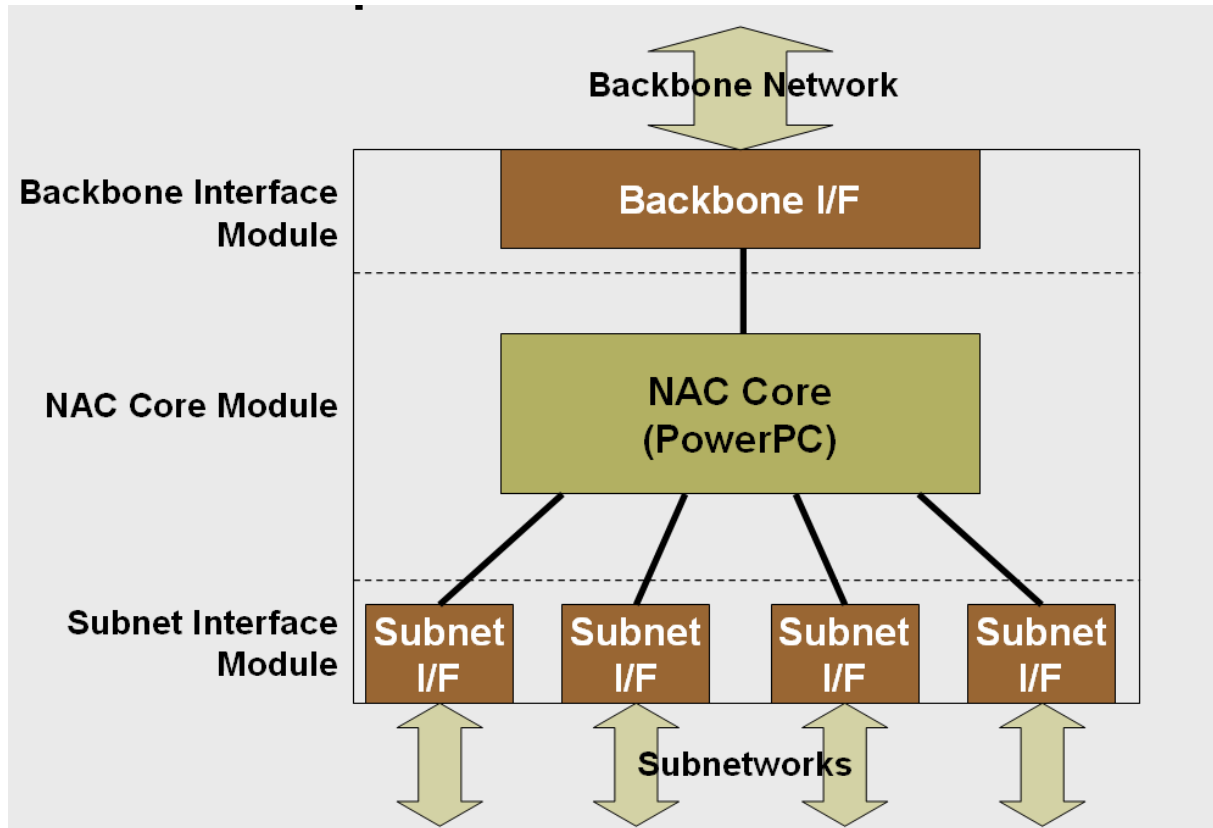


Figure 3-4 Block diagram of Network Access Controller (NAC)

The NAC itself connects the high data rate backbone with the subnetworks. Figure 3-4 shows the block diagram of a NAC. The NAC has a modular based structure with generalized interfaces to the backbone and to the subnetworks. The NAC Core Module provides gateway functionality between the backbone and the subnetworks. The backbone network is based on Ethernet communication protocol whereas the subnetworks can be implemented to handle protocols like CAN, Ethernet, etc.

In the following the features of the NAC are described in a short table form with description of the feature, the importance, a rational description and if applicable some implementation hints:

ID	NAC-Aero-1
Description	Fail Safe
Importance	High
Rational	The failure of one NAC must not effect the backbone network of any other NAC
Implementation Hints	In case of failure bypass the backbone

ID	NAC-Aero-2
Description	Build In Test
Importance	Low
Rational	The NAC should check the correct functionality by itself
Implementation	

Hints	
--------------	--

ID	NAC-Aero-3
Description	Connection to backbone
Importance	High
Rational	The NAC should have an interface to the backbone
Implementation Hints	

ID	NAC-Aero-4
Description	Connectivity of up to 4 subnets to each NAC
Importance	High
Rational	The NAC should have interfaces for up to 4 subnets
Implementation Hints	

ID	NAC-Aero-5
Description	Connectivity of up to 8 devices to each subnet
Importance	High
Rational	The NAC should have connectivity for up to 8 devices per subnet
Implementation Hints	

ID	NAC-Aero-6
Description	Miniaturized latency of dataflow between CU and PSUs
Importance	High
Rational	The latency of dataflow between the CU to a PSU and vice versa must not exceed 200ms
Implementation Hints	

ID	NAC-Aero-7
Description	Synchronisation of audio output
Importance	High
Rational	The synchronous output of an audio stream on different PSUs must be within 1ms
Implementation Hints	Synchronisation of PSU internal clock

ID	NAC-Aero-8
Description	Power supply of NAC must be compatible to A/C
Importance	High
Rational	
Implementation Hints	

ID	NAC-Aero-9
Description	Scalability of network
Importance	High
Rational	Both network, the backbone and sub-network should be scalable
Implementation Hints	

ID	NAC-Aero-10
Description	Connectivity of up to 12 NACs to the backbone
Importance	High
Rational	
Implementation Hints	

ID	NAC-Aero-11
Description	Time-triggered communication will not be used for the NAC and therefore the Networks should be asynchronous
Importance	High
Rational	
Implementation Hints	

ID	NAC-Aero-12
Description	The Sub-Network should support different physical layer, e.g. optical
Importance	High
Rational	
Implementation Hints	

ID	NAC-Aero-13
Description	The backbone should support a data rate in minimum of 100 MBIT/s up to 1000 Mbit/s
Importance	
Rational	
Implementation Hints	

ID	NAC-Aero-14
Description	The subnetworks should support a data rate of up to 100 MBIT/s
Importance	
Rational	
Implementation Hints	

ID	NAC-Aero-15
Description	The backbone network and the subnetworks should be segregated
Importance	
Rational	A failure in a subnet should have no impact to the backbone or other subnets
Implementation Hints	

ID	NAC-Aero-16
Description	The NAC should operate with different network implementations
Importance	High
Rational	The technology used for the backbone is not necessarily the one of the subnets
Implementation Hints	

ID	NAC-Aero-17
Description	The design of a NAC should be modular
Importance	High
Rational	It should be possible to integrate easily different subnetworks by changing a subnet interface module
Implementation Hints	

4. Aerospace Platform in the GENESYS context

The GENESYS architecture includes an architectural style with fundamental principles for the design of embedded systems. This section establishes the relations between the GENESYS architectural principles [1] and the requirements and features of the aerospace platform.

Architectural Principles in the GENESYS Style	Aerospace Platform Implementation
Component-based design	<p>The GENESYS architectural style follows a strict component orientation from the hardware/software point of view. A component is a self-contained subsystem that can be independently developed and used as a building block in the design of a larger system.</p> <p>The aerospace platform adheres to this principle. The cabin communication network architecture introduces Central Units (CUs), Network Access Controllers (NACs) and Passenger Service Units (PSUs) each representing a component according to the GENESYS architectural style.</p>
Message-based communication infrastructure	<p>The communication infrastructure of the architecture is based on the paradigm of message passing. The static communication schedule is defined with respect to messages (Req-Aero-11) and the HW-COM layer supports the packing and unpacking of messages (Req-Aero-14).</p>
Multi-level structure	<p>The cabin communication network architecture introduces two levels of integration, namely the subnet level and the overall system level. The overall system consists of the subnets interconnected by the backbone network.</p>
Common time base	<p>A common time base is supported by Req-Aero-10 and enables time-triggered communication and computational activities.</p>
Communication Modes	<p>The GENESYS architectural style recognizes that different application subsystems can have different requirements concerning the underlying communication infrastructure. For this reason, three</p>

	communication modes are introduced: periodic, sporadic and streaming communication. These communication modes are supported for the aerospace domain as described in Section 5.1.4.
Heterogeneous networks and internet connectivity	Support for heterogeneity is reflected in features NAC-Aero-12, NAC-Aero-13, NAC-Aero-14 and NAC-Aero-16. The network needs to support different physical layers, data rates, and network implementations.
Fault and error containment	Fault and error containment are important issues when dealing with both physical faults and design faults. NAC-Aero-15 demands that a failure in a subnet should not have any effect on other subnets or on the backbone. Likewise, the failure of a network access controller should not affect the backbone network or other network access controllers (NAC-Aero-1). Error containment through active redundancy is supported through requirement Req-Aero-16.
State awareness	A state aware design is required for making explicit the state of components and providing support for the restoring of state (e.g., restoration from replicas or environment). This architectural principle is captured in Req-Aero-17.
Diagnosis	Several diagnostic capabilities are defined for the aerospace domain, including node-local error detection (Requirement Req-Aero-5), clique detection (Req-Aero-7) and state externalization (Requirement Req-Aero-17).

5. Mapping of features to GENESYS reference architecture template

The GENESYS reference architecture template [2] provides architectural services as a baseline for the development of applications. GENESYS distinguishes between core services and optional services. The core services are mandatory in every GENESYS-based system. They provide a stable baseline for the realization of applications and higher architectural services. The optional architectural services are optional in the sense that they are not required in every instantiation of the architecture. If needed, developers can pick them out of the GENESYS reference architecture template, which includes a set of existing service specifications for the different levels of integration.

In the following the identified aerospace requirements and features are mapped to the GENESYS reference architecture template.

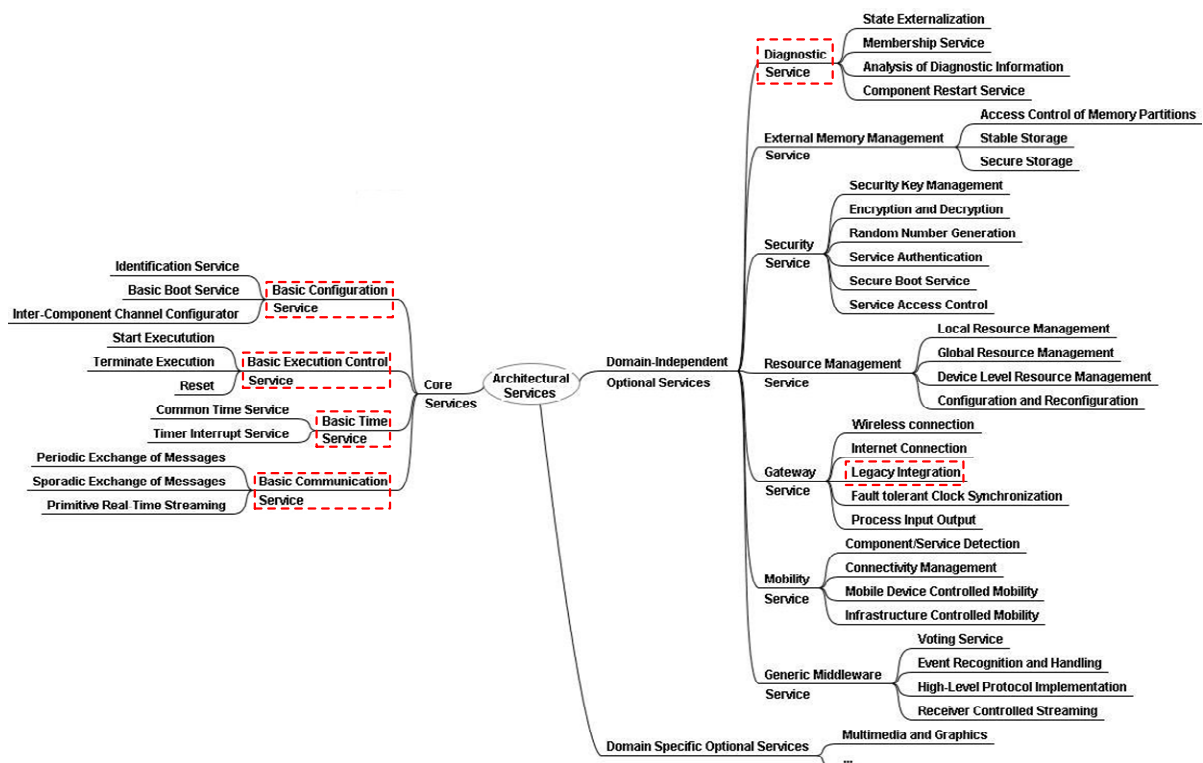


Figure 5: Supported Core and Optional Architectural Services

5.1 Core Architectural Services

The core services can be grouped into four categories, the basic configuration services, the component execution services, the basic communication services and the basic time services.

5.1.1 Basic Configuration Services

The basic configuration services are needed to introduce the components to the platform and to connect the ports of the components, thus establishing communication channels among the components. In order to facilitate certification to the highest criticality levels (e.g., class A according to DO-178B [3]), the GENESYS architecture [2] also supports static configurations in which the set of components and the communication channels are defined statically.

This approach is followed for the aerospace domain. Req-Aero-11 “Static network schedule” introduces the requirement of a static communication schedule that is defined in a message descriptor list. Req-Aero-11 demands that the system is configured offline in order to ensure high predictability at design time.

Configuration is supported within components through a configurable HW-COM layer (Req-Aero-15). The configurable tables can be modified in order to support different configurations.

5.1.2 Basic Execution Control Services

The basic execution services are controlling (e.g., start and stop) the execution of software components. In particular, the basic execution services enable the recovery from transient failures by performing a restart of components. To this end, the architecture has to provide mechanisms to regularly distribute the component’s state and to restore it in the component after a restart. During system design, this requires the definition and externalization of those parts of the component’s state that are relevant for the component reintegration.

Req-Aero-17 (“Reintegration service of replicas”) introduces basic execution control capabilities. This requirement demands the ability for restarting components and restoring the component state from replicas.

5.1.3 Basic Time Services

The basic time service performs clock synchronization in order to establish a common time base for the temporal coordination of distributed activities (e.g., synchronized messages), as well as to interrelate timestamps assigned at different components.

Req-Aero-10 defines the requirement of such a global time base, which is established by fault tolerant clock synchronization.

5.1.4 Basic Communication Services

The GENESYS architecture specifies three services for the communication among components: periodic message exchange, sporadic message exchange and real-time streaming communication.

Periodic message exchanges are supported in the subnets by Req-Aero-10. A static message schedule defines the instants with respect to the global time base when messages must be sent or received. The backbone, on the other hand, should be asynchronous (NAC-Aero-11) and support sporadic message exchanges. In addition, the backbone supports real-time streaming. NAC-Aero-7 defines the ability for audio streaming with synchronous output of audio streams within 1ms.

5.2 Optional Architectural Services

On top of the core services, which are used in all INDEXYS application domains, the optional platform services establish higher-level capabilities for the aerospace domain (i.e., diagnostic services, aerospace legacy services).

5.2.1 Diagnostic Services

The aerospace domain defines requirements and features for a number of services that relate to diagnostics and robustness. Although these services are optional from the point of view of the GENESYS architecture, they should be included in an instantiation of the architecture for the aerospace domain.

Requirement Req-Aero-5 is a diagnostic service for the detection of errors within a node. This diagnostic service is used for mapping different internal failure modes into fail-silent behaviour.

Req-Aero-7 “Clique Detection” introduces a diagnostic service for the detection of inconsistencies in the controller state. Based on the implicit acknowledgment mechanism (Req-Aero-7), cliques with different controller states are detected.

Requirement Req-Aero-17 “state externalization” is a diagnostic service for externalizing the component state. This externalized state enables the execution of diagnostic checks on the component state through another node (e.g., a diagnostic unit). In addition, the externalized state provides the basis for state restoration.

5.2.2 Legacy Integration

The reuse of legacy systems is important in order to preserve investments in existing applications. Therefore, requirement Req-Aero-6 “Legacy TD-COM” ensures that existing applications developed for TTP can be reused. This requirement is reflected in Feature RDC-Aero-8 “Implementation into Legacy Platforms and Applications”. The interface between the CPU and the HW-COM can easily be implemented in legacy platforms.

5.2.3 Reference Table Requirements to Features

The tables provided below show a cross reference between the requirements and the features implemented. They are is split into the tables for

- The “Core Services”
- The “Optional Services”.

Type of Service	Category	Service Name	Important for considered domain (yes / no)	Rationale for importance	Requirement	Feature
Core Service	Basic Configuration Services	Identification Service	yes	Every node in a TTP-network has a dedicated identification		
		Basic Boot Service	yes	The boot service is executed by the TTP bootloader		RDC-Aero-3
		Inter-Component Channel Configurator	yes	The configuration of the inter-component channels is done by the operating system		
	Basic Execution Control Services	Start Execution	yes	Execution control is done by the operating system		
		Terminate Execution	yes	—//—		
		Reset	yes	—//—		
	Basic Time Services	Common Time Service	yes	Every node in a TTP network calculates an average time out of the local node clocks by executing a fault-tolerant clock synchronization algorithm	Req-Aero-11	
		Timer Interrupt Service	yes	Timer interrupt services are used for frame copying, message pack and unpacking, common time calculation, ...		
	Basic Communication Services	Periodic Exchange of Messages	yes	The messages are scheduled in a periodic manner at dedicated points in time		
		Sporadic Exchange of Messages	no	TTP does not support sporadic message exchange		
		Primitive Real-Time Streaming	no	Data streaming is not important in TTP-networks		

Table 1: Cross Reference Table for the Core Services

Type of Service	Category	Service Name	Important for considered domain (yes / no)	Rationale for importance	Requirement	Feature
Optional Services	Diagnostic Services	State Externalization	yes	The controller state (C-state) in a TTP network consists of the global time, the message descriptor list position, the mode change bits and the membership vector of the nodes	Req-Aero-7	
		Membership Service	yes	TTP provides a membership service	Req-Aero-8, Req-Aero-9	
		Analysis of Diagnostic Information	yes	Periodic diagnostic is done by implicit acknowledgment and clique detection	Req-Aero-8, Req-Aero-9	RDC-Aero-6
		Component Restart Service	yes	Restart services of the TTP-controller can be enabled for different kinds of faults (membership, clique error, BIST, ...)	Req-Aero-18	
	External Memory Management Service	Access Control of Memory Partitions	no	External memory access mechanism are not used for the TTP protocol - the nodes solely uses local memories and strictly message based communication		
		Stable Storage	no	—//—		
		Secure Storage	no	—//—		
	Security Services	Secure Key Management	no	The TTP protocol does not offer any inherent security services, as it operates on a closed network.		
		Encryption and Decryption	no	TTP does not offer inherent security services		
		Random Number Generation	no	—//—		
		Service Authentication	no	—//—		
		Secure Boot Service	no	—//—		
		Service Access Control	no	—//—		

Table 2: Cross Reference Table for the Optional Services (Part 1)

Type of Service	Category	Service Name	Important for considered domain (yes / no)	Rationale for importance	Requirement	Feature
Optional Services	Resource Management Services	Local Resource Management	yes	Local resource management at node level is done by the operating system (time/scheduling management, memory management).		
		Global Resource Management	no	There is no additional global resource management to the basic configuration services		
		Device Level Resource Management	no	There is no additional device resource management to the basic configuration services		
		Configuration and Reconfiguration	no	Configuration and Reconfiguration is done by the application		
	Gateway Services	Wireless connection	no	TTP does not support wireless services		
		Internet Connection	no	TTP is a closed network without the necessity of internet connectivity		
		Legacy Integration	yes	The HW-COM implementation shall be able to communicate with TD-COM nodes within the same TTP network	Req-Aero-6	RDC-Aero-8
		Fault-tolerant Clock Synchronization	yes	TTP uses the fault tolerant average algorithm to establish a global time base	Req-Aero-11	
		Process Input Output	yes	The I/O management is done by the operating system		

Table 3: Cross Reference Table for the Optional Services (Part 2)

Type of Service	Category	Service Name	Important for considered domain (yes / no)	Rationale for importance	Requirement	Feature
Optional Services	Mobility Services	Component/Service Detection	no	Dynamic configuration changes are not allowed in a TTP network, therefore no component/service detection is necessary.		
		Connectivity Management	no	Devices are stationary in a TTP network.		
		Mobile Device Controlled Mobility	no	TTP does not require mobility		
		Infrastructure Controlled Mobility:	no	—//—		
	Generic Middleware Services	Voting Service	yes	Voting is done at the node level	Req-Aero-17	
		Event Recognition and Handling	yes	Incoming events like timer interrupts are handled by the operating system		
		High-Level Protocol Implementation	no	There is no high-level protocol implementation needed		
		Receiver Controlled Streaming	no	Data streaming is not important in TTP-networks		

Table 4: Cross Reference Table for the Optional Services (Part 3)

6. ANNEX A: Abbreviations

PSU	Passenger Service Unit
NAC	Network Access Controller
CU	Central Unit
I/F	Interface
CAN	Control Area Network
TTP	Time-Triggered Protocol
MEDL	Message Descriptor List
LRU	Line Replaceable Unit
CNI	Communication Network Interface
FCB	Frame Copy Buffer
MHB	Message Handling Buffer
FCL	Frame Copy Layer
MHL	Message Handling Layer
TD-COM	Table Driver Communication Layer
HW-COM	Hardware Communication Layer
DAL	Design Assurance Level
CRI	Certification Review Item

7. References

- [1] R. Obermaisser, C. El Salloum, B. Huber, and H. Kopetz, "Fundamental Design Principles for Embedded Systems: The Architectural Style of the Cross-Domain Architecture GENESYS," in *Proceedings of the 8th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC 2009)*, 2009, pp. pp. 3-11.
- [2] R. Obermaisser and H. Kopetz (Eds.), *GENESYS: A Candidate for an ARTEMIS Cross-Domain Reference Architecture for Embedded Systems*: Südwestdeutsche Verlag für Hochschulschriften, 2009.
- [3] Radio Technical Commission for Aeronautics (RTCA), "DO-178B: Software Considerations in Airborne Systems and Equipment Certification.," 1992.