Document Title	**Cover page for Systems ITD Grant Agreement for Partners (GAP) DELIVERABLES**
Issue date	A01 - Apr 05 2017

Clean Sky 2 - Systems ITD

Project ID	864475
Acronym	FAVIT
Title	FEASIBILITY ANALYSIS OF INNOVATIVE PRACTICES IN VIRTUAL TESTING METHODS FOR AIRCRAFT CERTIFICATION
Starting date	01/10/2019
End date	31/03/2022
Project Period	2
Deliverable id	D3.1
Deliverable Title	D01.c Best practices for virtual testing

Author(s)	Partner:	Date and Signature
Neil Snodgrass	ORBITAL	07/03/2022 [illegible]
Reviewed	Topic Manager:	Date and Signature
Soeren Reglitz	dSPACE	18/05/2022 Sören Reglitz

GAPs deliverables should be uploaded in COMPASS/SyGMA by the partner with this cover page signed by the Topic manager formalizing the reception and review of the deliverable.

Orbital CS
Carretera de Artica 29, 3ª Planta
31013 ARTICA (Navarra)
Tel. +34 848 47 31 81

FEASIBILITY ANALYSIS OF INNOVATIVE PRACTICES IN VIRTUAL TESTING METHODS FOR AIRCRAFT CERTIFICATION

Project: FAVIT

Grant Agreement nº: 864475

D01.c. Definition of the best practices, methods and processes for the use of Virtual Testing in the certification of aircraft systems

DUE DATE OF DELIVERABLE: 31/07/2021

ACTUAL SUBMISSION DATE: 30/05/2022

START DATE OF THE PROJECT: 01/10/2019

PROJECT DURATION: 30 months

DISSEMINATION LEVEL: PU (Public)

	Name	Date
Prepared	Neil Snodgrass	07/03/2022
Checked	Adriana Echauri	07/03/2022
Accepted	Adriana Echauri	07/03/2022

This project has received funding from the Clean Sky 2 Joint Undertaking (JU) under grant agreement No 864475.
The JU receives support from the European Union’s Horizon 2020 research and innovation programme and the Clean Sky 2 JU members other than the Union.

Document Change Log

Iss./Rev.	Date	Section / Page	Change Description
D01.c_v1	22/09/2021	All	1st version sent to TM
D01.c_v2	07/03/2022	All	Final version sent to TM

This project has received funding from the Clean Sky 2 Joint Undertaking (JU) under grant agreement No 864475.
The JU receives support from the European Union's Horizon 2020 research and innovation programme and the Clean Sky 2 JU members other than the Union.

Contents

LIST OF FIGURES

LIST OF TABLES

EXECUTIVE SUMMARY

FAVIT's main objective is to deliver a set of knowledge-based proposals for the improvement of aerospace standards and guidelines for the system suppliers and aircraft manufacturers. FAVIT will analyse the current aerospace standards and guidelines to identify how the design and verification processes can be enhanced to accelerate the processes using the state-of-the-art technologies based in virtual testing.

The purpose of this deliverable is to propose a way forward to allow the incorporation of virtual testing technologies (VTTs) into the development process for an aircraft system so that they may better contribute to the certification evidence for that system.

This will include proposals for new processes, practices and methods, and a new approach to the current development processes that include better consideration of VTTs from the very beginning.

This document has also given rise to the need for a new top-level definition of an aircraft platform. Such a definition will be used in a top-down manner for the development of both the system (under development) specification and a Virtual Testing Environment (VTE).

The need for a new document to specifically deal with Virtual testing has arisen and an outline of that document's contents has been provided. Modifications to the existing documents that would facilitate this have also been given.

REFERENCED DOCUMENTS

[D01.a]	*Gap Analysis*
[D01.b]	*Identification of Challenges*
[SAE-ARP 4754]	*Certification Considerations for Highly-Integrated or Complex Aircraft Systems, November 1996*
[RTCA DO-297]	*Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations November 2008*
[RTCA DO-178/C]	*Software Considerations in Airborne Systems and Equipment Certification, December 2011*
[RTCA DO-254]	*Design Assurance Guidelines for Airborne Electronic Hardware, April 2009*
[D100.3.6.1_a]	*Virtual Testing Methods Definition, December 2017*
[D100.3.2.1_e]	*Documentation to the Technology Readiness Level 5 Demonstration of the Software Core Environment for MISSION Framework*

1 INTRODUCTION

1.1 PURPOSE

This document follows on from two previously-delivered documents:

[D01.a] – which identified possible gaps in the current certification standards documents that impede the use of virtual testing technologies (VTTs) in the certification of airborne systems.

[D01.b] – which identified the challenges that exist for the incorporation of virtual testing technologies (VTTs) in the certification of airborne systems.

(It is assumed that the reader is familiar with these two documents)

The purpose of this document is to:

- Define the best practices for the use of VTTs
- Describe the new methods for the use of VTTs
- Develop new processes to use the different types of VTTs (MIL, SIL, VPIL and HIL)
- Generate change proposals for the modification of current guidelines to better promote the use of VTTs for the certification of airborne systems

The goal of this document is to determine what needs to be done differently in order to allow the use of VTTs to be more acceptable in the gathering of certification evidence for an aircraft subsystem.

The context of this document is in the application of VTTs to the testing activity of hardware or software development but it could also be applied to system level testing.

1.2 CONTENTS

Section 2 of this document shall present a brief summary of ***[D01.a]***, ***[D01.b]*** and another relevant document, ***[D100.3.2.1_e]***.

Sections 3, 4 and ***5*** shall introduce the concept of a platform definition and how that impacts the system and virtual test environment (VTE) specifications.

Section 6 identifies new methods that can be adopted in the use of VTTs.

Section 7 identifies new methods that can be adopted in the use of VTTs.

Section 8 defines new processes that will better incorporate the use of VTTs.

Section 9 discusses necessary changes to specific certification guidelines documents that will allow better use of VTTs.

1.3 EXAMPLE PROJECT

Throughout this document, an example project shall be used to illustrate the ideas promoted herein.

The project is the development of a simplified Landing Gear Control Unit (LGCU) subsystem that forms part of a simplified, integrated aircraft system.

The LGCU will be implemented in software running on a dedicated Line-Replaceable Unit (LRU).

A schematic of the aircraft system is presented as part of the platform definition (***section 3***).

1.4 TERMINOLOGY USED

In this document the following terminology is used:

System: *can apply to the overall aircraft system or a subsystem of that (depending on context).*

Subsystem: *a subsystem of the aircraft system, equivalent to a Line Replaceable Unit (LRU).*

LRU: *equivalent to a subsystem of the aircraft system.*

Process: *the development process of the product. Contains several activities (see below).*

Activity: *a task of the process. E.g. requirements specification, design, implementation, testing.*

Certification Process: *The "process" in the context of providing lifecycle data to certify the product.*

2 SUMMARY OF PREVIOUS WORK

Here is a summary of the contents and conclusions of the two previously-delivered documents that are concerned with the use of VTTs in the certification of airborne systems. There is also a summary of a MISSION project document that is concerned with the process and tools for developing model-based systems engineering projects using VTTs.

2.1 [D01.A]

The title of the document is "Gap Analysis". It was a study of the current certification standards and guideline documents with a view to identify any "gaps" within their contents that may otherwise promote the use of VTTs in the certification process.

Here is a summary of the conclusions of this document:

Some formal documentation is needed that deals specifically with different test environments (different VTT methods).

This would include a guideline for how to use them in certifiable product development processes and a means of categorizing such test environments so that they can be more appropriately applied to different assurance levels of the product under test.

Another conclusion is that the existing standards have their fair share of ambiguities when it comes to what test environment can currently be used in the gathering of certification evidences. This may well explain why AC/0 type testing is still so heavily relied upon in the industry.

2.2 [D01.B]

The document is titled "Challenges of New Virtual Testing Methods in the Current Aircraft Certification". It aims to identify what obstacles must be overcome to be able to make better use of VTTs in the certification of airborne systems.

A summary of the conclusions of this document is:

- *A means to match a Virtual Test Environment (VTE(to the Design Assurance Level (DAL) of the component under test is needed*
- *Adopt a requirements/design philosophy that would not only promote portability between a target environment and a VTE, but also reduce the amount of testing needed in a real environment.*
- *Guidelines (standards) for the use of VTTs are needed*

2.3 [TRL5]

The document is a technology readiness demonstration of a framework for model-based systems engineering (MBSE) that uses VTTs to verify the requirements of the modelled system within the scope of the MISSION project.

The framework used is called SYNECT and is a centralized development platform for the management of requirements, models, tests and traceability and configuration management through all stages of development. Aside from demonstrating the viability of SYNECT for collaborative, large-scale projects the document also shows the consistency of results obtained through several virtual testing methods (MIL, SIL and VPIL).

The development process used for the demonstration is based on the V-Model. Of particular interest to this document is that [TRL5] does not mention existing standards nor how to use a particular VTT to satisfy the certification criteria of different assurance levels.

Since the document is concerned only with MBSE, it also does not consider the development of non-model based systems.

3 PLATFORM DEFINITION

As can be seen by the conclusions drawn from the previous work, there is a need to ensure to what degree a Virtual Testing Environment (VTE) is representative of the real target environment, and also to what Design Assurance Level (DAL) of the subsystem under test that VTE can be used as certification evidence.

This introduces the idea that there must be a way to measure the level of coherence between a VTE and the real target based on the underlying characteristics of each platform.

So a top-level definition of such characteristics is needed from which the VTE and the real target environment can derive their own implementations. *****

***** *In the scope of a developer of a particular subsystem, this implementation would begin with the System Specification for that subsystem.*

Therefore, a clear definition of a platform is needed.

For the purposes of this document, the platform is considered to be the encompassing environment in which the product/system under development will reside. Essentially the subsystems of the platform (LRUs) and the infrastructure that connects those subsystems. It does not describe internal, functional behaviour of any subsystems that comprise the platform.

As will be seen in subsequent sections of this document, the platform definition will influence not only any system requirements of the real target environment and the subsystem requirements of the product under development but also the requirements of the VTE to be used for developing and testing that product.

For this reason, the platform definition must be more abstract than what may usually be specified in a system/subsystem specification. For example, specific types of hardware devices would not be specified in the platform definition.

3.1 CHARACTERISTICS OF A PLATFORM

A platform definition can include:

- Physical characteristics:
    - Operational temperature range
    - Motion/vibrational considerations
    - Physical materials and dimensions of the infrastructure
- Topographical considerations:
    - Network infrastructures
    - Subsystem boundaries
    - Distribution of LRUs
- Electrical characteristics:
    - Power consumption demands
    - Power supplies and redundancy
    - Insulation / electro-magnetic shielding

- Performance characteristics:
  - Data rates of the LRUs and interconnecting buses
  - Computational rates of the LRUs
  - Power consumption of each of the LRUs and infrastructure
- Interoperability requirements:
  - LRU data exchanges
  - LRU failure strategies (e.g. dual-redundant systems)
  - Operational modes (platform-level and LRU level)
- Design Assurance Levels (DALs):
  - Identifies the DALs of each LRU of the platform
- Non-subsystem components:
  - For example, a flight simulator or a monitoring tool

There is scope for re-using a platform definition across different projects. Whether that project is the development of a different subsystem of a particular aircraft, for example, or if a different aircraft that will have a similar platform (this would imply that the platform is configurable for different projects).

The framework used in ***[D100.3.2.1_e]*** could be extended to include the platform definition and manage any necessary configurations.

3.2 INTERFACE CONTROL DOCUMENT (ICD)

The use of a top-level ICD is a common practice and for good reason. A single, controlled source of information is necessary when developing different subsystems that interact with each other.

The idea of a platform definition introduces the possibility of including the information previously-held in an ICD within the definition of the platform itself although if the platform is to be reusable across projects a separate, project-specific ICD would be needed.

3.3 EXAMPLE PLATFORM

The diagram below shows a schematic of the platform to be used in the example project for this document:

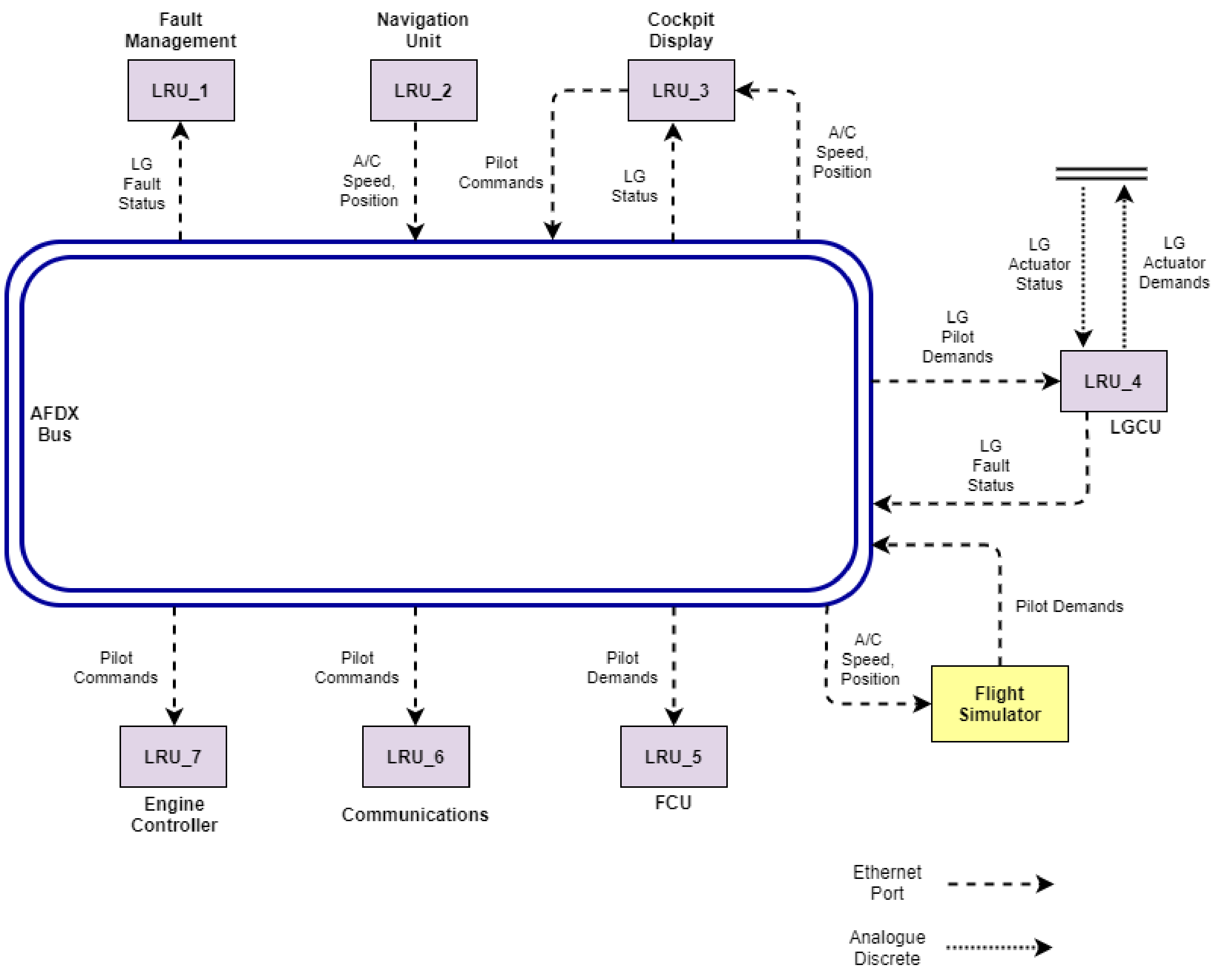

Figure 1 - Platform Definition

The LGCU (LRU_4) has more detail shown because it is the subsystem in focus for this document. This diagram is not meant to show all the details specified above, just to put the whole platform into context for the remainder of this document.

The inclusion of a flight simulator in the diagram would be applicable to an aircraft zero (A/C 0) type of implementation for ground testing of the real A/C subsystems. For flight testing, there would be no flight simulator component.

4 SYSTEM / SUBSYSTEM

The example of an aircraft system used in this document comprises several subsystems and accompanying infrastructure (as would be specified by the platform definition).

Within the scope of a particular developer, the term System is normally used to describe the actual Subsystem (of the whole aircraft system) under development.

This use of terminology shall be maintained throughout this document. That is to say, in the context of this document, the System Specification shall actually be a specification of the LGCU (***LRU_4***) subsystem of the platform, not the entire aircraft system.

Therefore the system specification will actually specify how the LGCU is to be implemented within the constraints defined by the platform definition. This is achieved by elaborating on the platform requirements to specify such details as:

- Hardware specifications
- Operating system to be used
- Hardware functions to be used
- Software functions to be used
- Specific fault scenarios to be handled
- Internal functionality of the subsystem (e.g. landing gear deployment constraints, anti-skid breaking algorithms)
- Contents of the data interfaces to be used *****

***** *This can also be specified by the platform definition itself or a similarly high-level document such as an interface control document (ICD).*

5 VIRTUAL TEST ENVIRONMENT

A virtual test environment (VTE) is the implementation of a platform that will be used to facilitate virtual testing of a product for that platform.

The VTE comprises one or more VTT implementations (see below), the necessary infrastructure to connect those VTTs to each other and any other auxiliary tools for control or monitoring purposes.

The specified characteristics of the platform will be replicated or simulated by the VTE where each VTT implementation shall represent a subsystem (LRU).

There is scope for making a VTE configurable so that it can be used between projects that contain similar subsystems. For example an aircraft manufacturer may use re-use a particular subset of LRUs across different aircraft projects.

The VTE for a platform could (and probably should) also be re-used by the developers of each subsystem of the same aircraft project by simply replacing the VTE-provided simulation of their subsystem with the one under development.

5.1 VIRTUAL TESTING TECHNOLOGY

In the context of this document the use of a virtual testing technology (VTT) is the implementation of a LRU of the platform by using one of the established virtual testing methods.

The example platform contains 7 subsystems (***LRU_1 … LRU_7***), and one of those subsystems (***LRU_4***) will be the one under development (the LGCU). Each other ***LRU*** would be included in the VTE as one of the following:

- **MIL**: *A model-based simulation of the subsystem running on a host PC*
- **HIL**: *Complied code of the subsystem executing on the real hardware, or the real hardware implementation of the subsystem* *******
- **SIL**: *A software simulation of the subsystem running on a host PC*
- **VPIL**: *Compiled code of the subsystem executing on a CPU simulation of the real hardware on a host PC* *******

***** *Can be the real code (if available) or simulation code.*

The implementation of ***LRU_4*** itself would depend upon the verification strategy of the project. If some of the requirements can only be verified by testing on the real target hardware, then at some point ***LRU_4*** must implemented as a HIL VTT with the real object code executing on it.

If some requirements can be verified independently of the hardware, a SIL or VPIL implementation could be used. For model-based development, a MIL implementation would be used here.

So the implementation of the product under development may change at various stages of the project, especially if prototyping is to be used as part of the requirements/design elaboration.

There may also be other components implemented as simulations (e.g. a flight simulator) but the development and implementation of those are beyond the scope of this document.

The diagram below is an example VTE for the platform:

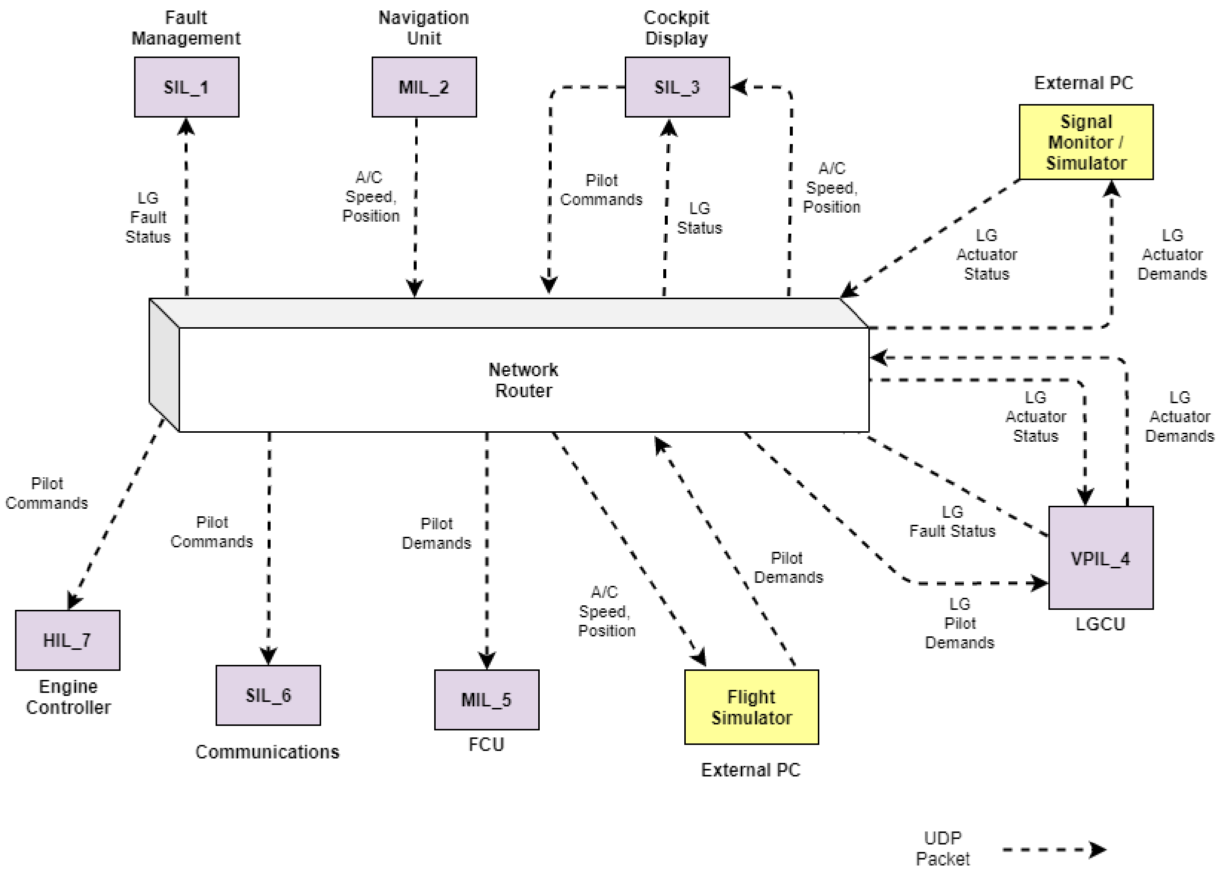

Figure 2 - Environment Specification

In this example, the engine controller (***LGU_7***) is actually a real system using the HIL Virtual Testing methodology. This is possible if a platform uses an existing product that is not to be modified for the aircraft system. The other subsystems are defined as being MIL, SIL or VPIL.

The LGCU subsystem in this example is defined as being VPIL which would allow testing of prototype code and the real code on a CPU emulation of the target hardware. Early prototyping would commonly be done with the SIL method also.

Note that if this VTE were to be reused by a developer of a different subsystem for the same aircraft, the LGCU subsystem could be implemented by another VTT.

The implementation of the VTE could be one a single host PC or distributed across several PCs.

In this VTE, the analogue and discrete data communication channels of the LGCU have been replaced by UDP data packets that connect to a simulator/monitoring tool. Such a practice would also be common for other types of data (e.g. Serial, Mil-1553).

6 BEST PRACTICES FOR THE USE OF VTTS

In general, there is only one "best practice" that would allow better use of VTTs in the certification process. That is to consider the use of VTTs at each relevant stage of project development.

Rather than just considering the real target environment, the use of VTTs should also be considered when specifying requirements, making design decisions, and formulating test plans.

The following sub-sections discuss this further for each of those scopes.

6.1 COLLABORATIVE VTE

The development of a VTE that fully represents a large system, such as an aircraft, would be a large project in itself.

Each developer of a subsystem may only need a portion of a fully-representative VTE in order to test their own subsystem. It is therefore undesirable and unrealistic to expect that each developer creates their own VTE.

Therefore, a common VTE (plus documentation) should be provided to each subsystem developer that is itself a collaboration of the partners involved.

For example, the developer of the LGCU may be able to provide a simple SIL-based emulation for use by other partners in the VTE. Similarly the developer of a hardware-only subsystem may be able to provide a MIL-based prototype that can be included in the VTE for other developers to use.

There should also be an effort to make the VTE framework itself re-usable between aircraft projects. Some sort of plug-and-play architecture should be adopted.

6.2 HIGH LEVEL REQUIREMENTS SPECIFICATIONS

Consideration of VTTs is needed to make the high-level requirements (HLRs) easier to test in a VTE, but such consideration does not mean that the subsystem HLRs would contain features that they would not otherwise do so in the absence of VTTs.

It should also be possible to implement the HLRs as a VTT (see above). That is to say, there should be no blocking requirements ***** that depend on target-specific components or features.

**** It may be that some requirements are specific to the target hardware, but they should not prevent a VTT implementation of the subsystem from being created for use in the collaborative VTE.***

Another important consideration is to isolate any requirements that cannot be verified by a VTE. For example; there may be an algorithm that depends upon data from a board support package (BSP) function. If the accessing of that data (by calling a BSP function) is not specified within the algorithm requirement itself (i.e. it is a separate requirement) the algorithmic requirement is more testable in a VTE that can simulate the data required.

7 NEW METHODS FOR THE USE OF VTTS

7.1 USE CASE ANALYSIS FOR THE VTE

This will identify which parts of the platform definition can be represented by which VTT implementations.

This analysis may also be used to determine which VTTs of that environment shall be used for formal certification or development work.

It is possible that each subsystem developer will not need a full VTE representation of the entire aircraft in order to test their own product. So a use case analysis for the VTE within the scope of a particular subsystem would help to define the necessary implementation of the VTE for that subsystem.

Similarly, a use case analysis may be useful to plan ahead if an aircraft is to have various stages of development. If planned functional milestones necessitate different levels of functionality from the various LRUs, then a VTE can also follow the same functional milestone plan.

These last two points are also important when considering different VTE configurations (***section 7.3***).

7.2 PROTOTYPING

To be used as part of the requirements elaboration, to identify and isolate those requirements that can be tested by VTTs only.

Prototyping in general is not a new practice but the idea here is to use the prototype not only as a general way of checking functional requirements or early design decisions, but also as a way of checking which of those requirements can be fully tested in a VTE and which must need to be tested on the real target.

7.3 VTE CONFIGURATION

The scope for different VTE configurations could be very large. A way of defining and managing such configurations is needed that is aligned with the ICD and a particular suite of LRU versions.

If virtual testing is to be used to gain a high-level of certification evidence, then the configuration control of the environment and its VTT components should be applied with the same diligence as for the airborne system under development that will use the VTE.

A particular subsystem may undergo a phased development process whereby successive, stable releases are delivered. Each more functional than the last. It may be possible to incorporate such releases into successive versions of a VTE.

8 NEW PROCESSES FOR THE USE OF VTTS

8.1 CURRENT PROCESS

Within the applicable scope of this document, the typical development process used is that of the V-Model, for both hardware and software development.

A top-down development process is used starting from the System/Subsystem specification.

The high-level requirements (HLRs) are then derived from the system requirements. A design specification is then created to specify how the HLRs will be implemented. This may or may not generate low-level requirements (LLRs).

The design is then implemented. If prototyping is used then the implementation itself would typically provide heavy feedback to the design activity before testing begins.

The test plan specifies how the requirements will be verified and validated (V&V). The V&V activity itself would specify the test procedures/test cases that test each requirement, and document the results.

Those results may then provide feedback to the previous activities before performing another iteration of the process.

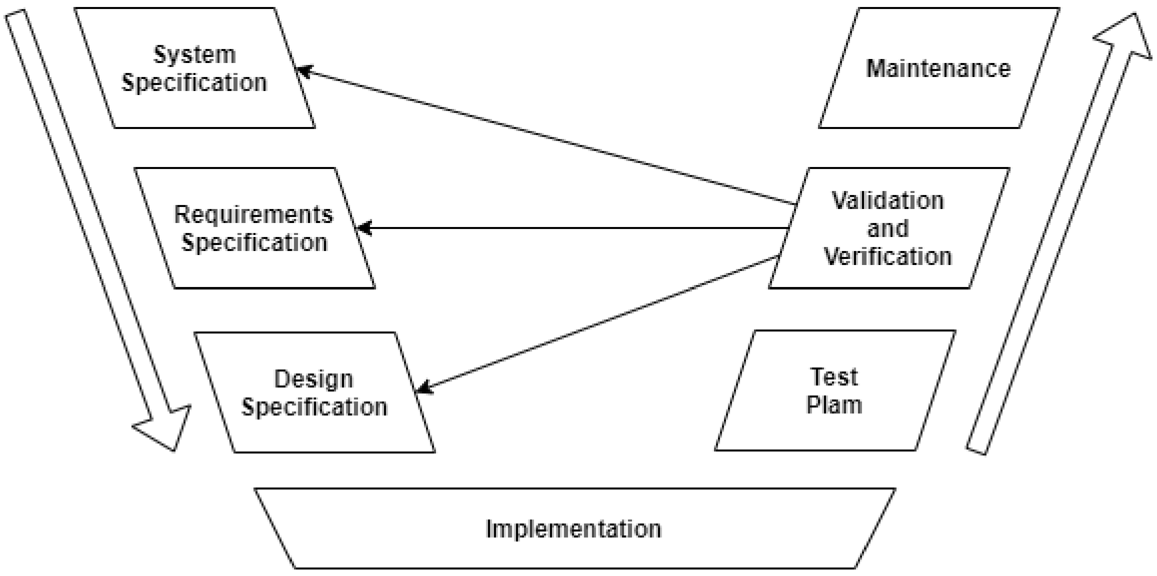

Figure 3 - Current Process

8.2 NEW DEVELOPMENT PROCESS

A new development process would not be practical because the industry standard is the V-Model. However, a new parallel process and activities are needed to specify how to use the available VTTs.

Rather than have a specific MIL process, for example, the use of MIL should be considered during each activity of the current process (i.e. Requirements, Design, Implementation, and Testing).

As such, under the following section, consideration of the VTTs will be added to current process for both the product development and VTE processes.

8.3 AMENDMENTS TO THE CURRENT PROCESS

In current product development, the target platform (or target environment) is already defined either by a system specification or in another higher-level document.

If virtual testing is to be more successfully used in the certification of the product, the virtual testing environment to be used in such certification activities must also be defined at some point in the process.

This can be considered as a parallel activity within the V-model development process with a higher-level platform definition being produced first:

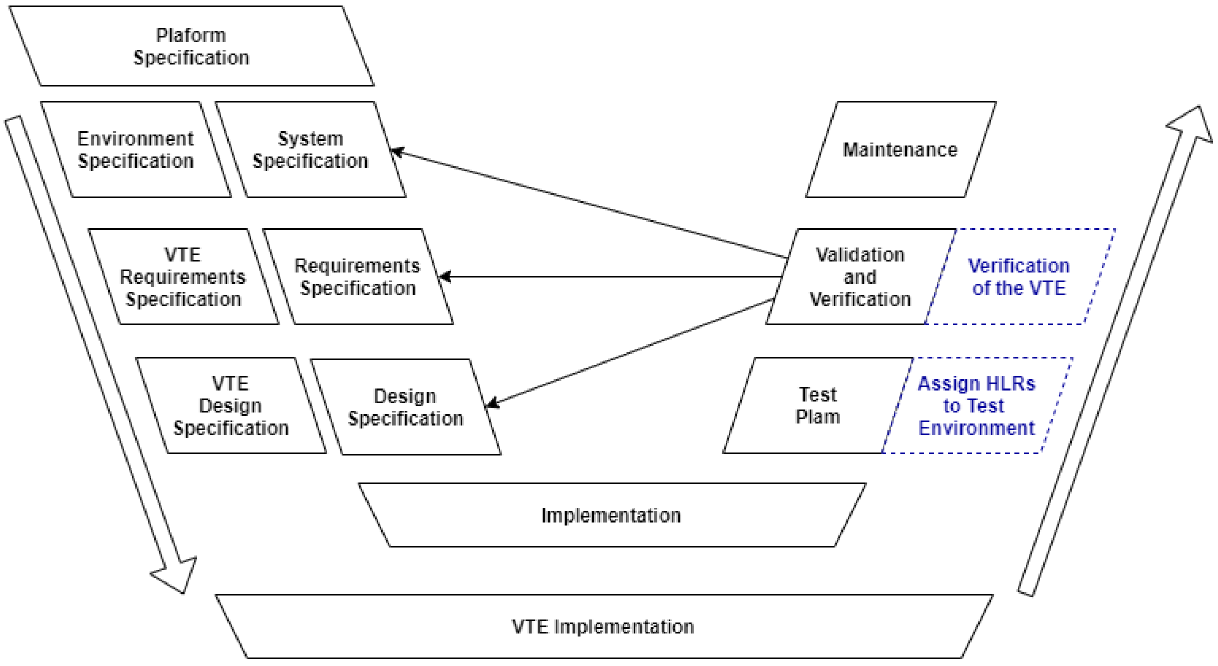

Figure 4 - New Process

The Environment Specification, VTE Requirements Specification and VTE Design Specification could all be sections of the System, Requirements and Design specifications of the project for which the VTE will be used. However, by keeping the documents separate it promotes the reuse potential of a VTE across different projects that use the same target environment.

The verification of the VTE is included in the project because a VTE would probably have configurable operational parameters (e.g. execution period of models, inter-component data, specific fault conditions), that would be defined by the project which uses the VTE.

There can also be a VTE-only, generic verification document that covers all the non-mutable VTE requirements.

8.3.1 Platform Definition

The process for creating the platform definition is beyond the scope of this document. It is in the domain of an aircraft manufacturer, not a subsystem developer, which is the target audience for this document.

The only caveat for defining the platform, however, is to keep it generic enough that it can be implemented by both a VTE and the real target environment.

8.3.2 System Requirements

This document would contain any necessary requirements to satisfy the desired functionality of the product under development.

If a separate Platform Definition is used, then the System Specification can also contain requirements that more precisely specify how to implement the platform requirements.

By way of an example, we can show how the system requirements would trace to the higher-level platform requirements:

Platform Requirement		System Requirement	
ID	**Description**	**ID**	**Description**
PL_LGCU_010	A dedicated LRU shall control the aircraft Landing Gear (the LGCU).	***SYS_LGCU_001***	The LGCU shall control the deployment of the landing gear and prevent the wheels from skidding.
PL_LGCU_011	The LGCU shall have two computational rates: 100 ms and 5 ms	***SYS_LGCU_002***	The LGCU shall comprise two software functions executing on a PPC-XXX processor using the YYY operating System.
		SYS_LGCU_003	The LGCU shall monitor deployment status and control deployment of the landing gear every 100 ms.

		SYS_LGCU_004	The LGCU shall employ an anti-skid algorithm when the aircraft has weight-on-wheels every 5 ms.
PL_LGCU_012	The LGCU shall transmit status information via AFDX every 100ms.	***SYS_LGCU_005***	The landing gear status shall be sent to the cockpit displays every 100ms on IP port 5051.
PL_AFDX_008	The LGCU status information shall use source port 5051.		
PL_LGCU_013	The LGCU shall transmit fault information via AFDX every 100ms.	***SYS_LGCU_006***	The landing gear fault status shall be sent to the health monitor subsystem every 100 ms on IP port 5052.
PL_AFDX_009	The LGCU fault information shall use source port 5052.		

Table 1 - Requirements Example

8.3.3 Environmental Specification

The VTE Specification is also derived from the definition of a platform and is a parallel activity to the System Specification.

Ideally, a platform definition could be reused between projects, and therefore the Environment could also be reused but in practice this may be unrealistic unless a heavy element of configurability is incorporated because the majority of projects do not seem to reuse the exact subsystem setups.

The activity of creating the environment specification would include some kind of use case analysis and resource/feasibility study to identify which subsystems will be implemented by which VTT (e.g. which LRU will be MIL, SIL, HIL, VPIL etc.). (See also ***section 7.1***).

It may be the case that all LRUs are available in all VTTs but the project which uses the environment then chooses which VTT to use for which LRU – depending on the DAL of the requirements to test with that LRU and resource availability.

It is important to note that as a project develops, the subsystem under development will mature and the VTT used to test it may change (e.g. from VPIL to HIL as the real hardware becomes available). So it is expected that the environment used by a particular developer will change as the project advances. (See also ***section 7.1***).

There is the option to generate successive versions of the environment specification through an iterative process, or to anticipate such changes in advance and include different configurations of the environment in the original specification.

8.3.4 High-Level Requirements Specification

The current activity of specifying high-level requirements (HLRs) uses the system specification and ICD as its inputs. As discussed in ***section 6.2***, more consideration of VTTs is needed when specifying requirements.

So another input to the HLR elaboration activity would be the Environment Specification which would allow the HLR specification to:

- Specify Requirements that would more easily allow the use of VTTs, for example:
  - Less HW dependency
  - Isolate HW-dependent requirements from others
  - Define clear interfaces
  - Include buffering requirements that make a system more robust to inter-component timing errors/data loss
- Identify those requirements that CAN ONLY be tested on the target.
- Identify those requirements that MUST be tested on the target but can first be tested with VTTs to mitigate against failures
- Identify those requirements that can be certified in a VTE

This gives rise to the classification of requirements by DAL. Usually a DAL is assigned to a CSCI/Hardware Component in its entirety. The assignment of DALs to different requirements of the same component is a new concept and would imply the use of some kind of analysis for this task within the requirements elaboration activity.

A requirement can also be assigned an attribute that identifies the target upon which it must be tested (e.g. MIL, HIL, SIL, VPIL, or the real target).

8.3.5 VTE Specification

Specifies the functional details of each LRU simulation and infrastructure requirement (as defined in the Environment specification). The level of detail would be similar to that of the HLRs in that they will allow design, implementation and testing of the VTE.

For LRUs that are to be MIL or SIL implementations, software requirements would be needed.

For VPIL implementations, there will also need to be some requirements that specify the target hardware that is to be emulated.

For HIL implementations probably no requirements need be specified. *

**** If the subsystem under development is to be HIL-based then the parallel subsystem and requirements specifications cover these requirements. If the HIL subsystem is from another supplier then it can be considered to be equivalent to a COTS product.***

This activity would be similar to that used for the elaboration of the HLRs with the higher-level environment specification as the principal input document.

8.3.6 Design Specification

As for the requirements elaboration activity, consideration of the use of VTTs is also encouraged during the subsystem design activity.

Ideally, most such VTT considerations will have been captured in the HLRs but there is also room for design decisions to be taken that further facilitate the use of VTTs.

The most important design decision would be the identification of components so that requirements of the same DAL can be implemented in the same component(s) wherever it is possible to do so. This means that parts of the implementation can be fully tested in a VTE without being dependent on those other parts of the implementation that need to be tested on real hardware.

8.3.7 VTE Design

The VTE design activity will define the following subsystems:

- All LRUs of the platform
- Infrastructure to support the LRUs executions and intercommunications
- External tools

The purpose of the VTE design specification is to allow the developers to implement the VTE and specify tests that will subsequently verify that implementation.

The actual design activity is no different from the real product's design specification activity but there will be more consideration as to how the VTE is to be tested.

It may well be the case that the same tools used to test the actual subsystem under development are not to be used for verifying the VTE.

Since the VTE is not airborne software and is not subject to the same safety requirements, it is acceptable to include non-functional design elements that facilitate the verification of the VTE (e.g. hooks to call functions, visibility of internal data, broadcasting of inter-LRU data, etc.). *****

**** It can also be argued that such considerations could be specified during the elaboration of the environment's requirements.***

8.3.8 Test Plan

8.3.8.1 Tools Identification

Will specify the tools used to support the use of the VTTs and how they will contribute to the verification of the requirements (e.g. data signal monitors).

Some work has been done on the MISSION project regarding tools (see ***[D100.3.2.1_e]***) and is beyond the scope of this document.

8.3.9 Test Procedures

8.3.9.1 HLR Testing

Since the HLRs will have been categorized by DALs, one important task of the test plan is to assign those HLRs to the VTT that will verify them.

So the highest level HLRs (DAL A) may only be verifiable on real hardware (a HIL VTT implementation or the real target environment).

HLRs of lower DALs may be verifiable on a MIL, SIL or VPIL VTT implementation.

This point is also touched upon in ***section 8.3.4***.

8.3.9.2 VTE Testing

This activity will specify how to test the requirements of the VTE specification.

It is not necessary to test the VTE requirements in the same way as for airborne software and quite possibly the same tools cannot be used to test both types of implementation.

Therefore, the test plan will need to define how to test the VTE which may mean the specification of testing tools (to be developed or purchased). Most probably, if a testing tool is to be developed it would be re-usable between subsystem projects of the aircraft and therefore have its own suite of development documentation.

9 CHANGE PROPOSALS

The process, methods and practices discussed above, if they are to be adopted, will need to be documented. This could be achieved by adding to the existing standards documents and/or creating a new document.

This document will propose that both approaches be undertaken.

A new standards/guidelines document to specifically deal with the use of VTTs is needed because the topic is so large. It will also be necessary to update the existing documents to incorporate considerations of the new one.

9.1 NEW DOCUMENT

Guidelines in a single, source document will coherency between all the other standards documents that may reference it.

The following will present an outline and brief description of the proposed contents for such a document.

Precise and more detailed contents of this document will be defined in subsequent deliverables of the FAVIT project.

9.1.1 Virtual Testing Technology Guidelines for the Certification of Airborne Systems

Introduction

- Purpose
  - *States that the document is to serve as a guideline for the use of Virtual Testing Technologies in the certification of airborne systems.*
- Scope
  - *Applicability of the document, intended users, and to be used together with existing documents. Intended users can be subsystem developers or dedicated VTE providers.*
- Background
  - *History of airborne systems certification.*
  - *History of virtual testing technologies and their use to date.*
  - *The impetus for promoting the use of VTTs in the certification process.*
    - *Resource availability*
    - *Cost of testing of real target*
    - *Early development environment by using VTEs with prototyped subsystems*
- Development Hierarchy
  - *Platform Definition drives the VTE implementation*

- *Compliance with ICD*

Virtual Testing Environment (VTE)

- Definition:
  - *Subsystem VTTs*
  - *Infrastructure*
  - *External tools*
  - *Configurations*

Virtual Testing Technology (VTT) Implementation

- Definitions:
  - MIL
  - HIL
  - SIL
  - VPIL
- Configuration Control of VTT Implementations

Platform Definition

- Acts as the only input to a VTE
- Briefly describe what is expected of a platform definition
- Refer to ***[SAE-ARP 4754]*** for an example platform definition template (see ***section ARP 4754***)

Development Process

Explains the recommended development process (further elaboration of the ideas in ***section 8.3***).

It can be part of the subsystem process as a parallel activity or a solitary process if the user of the document is specifically a VTE developer.

Subsections will be the typical development ones:

- Environment Specification
- VTE Requirements Specification
- VTE Design
- VTE Verification

Some consideration of the applicability of VTTs to the testing of different DALs would be useful here also. This will need to be synchronized with modifications to the current standards documents.

Configuration Management

- Collaborative VTEs
- Independent VTEs
- Re-usability
- Manage changing VTEs as the subsystem VTTs mature
- Reuse of a VTE between partners of the same aircraft project
- Reuse of a VTE for different aircraft projects

Example VTE

- Example Platform Definition (following the example schema to be added to ***[SAE-ARP 4754]***)
- Example walkthrough of the VTE development process
- Example configurations

9.2 EXISTING DOCUMENTS

The previous work done on analysing the existing documents will, be repeated here to some extent. There will also be some new change proposals to the existing documents discussed that would accommodate the ideas already presented in this document.

In general, all the documents would need to better integrate the use of virtual testing into the current guidelines, making reference to the proposed new document.

The following subsections will present an outline of some proposed changes. More detailed and specific change proposals will form part of a subsequent deliverable of the FAVIT project.

As mentioned in ***[D01.b]***, there does exist a hierarchy amongst the current standards documents:

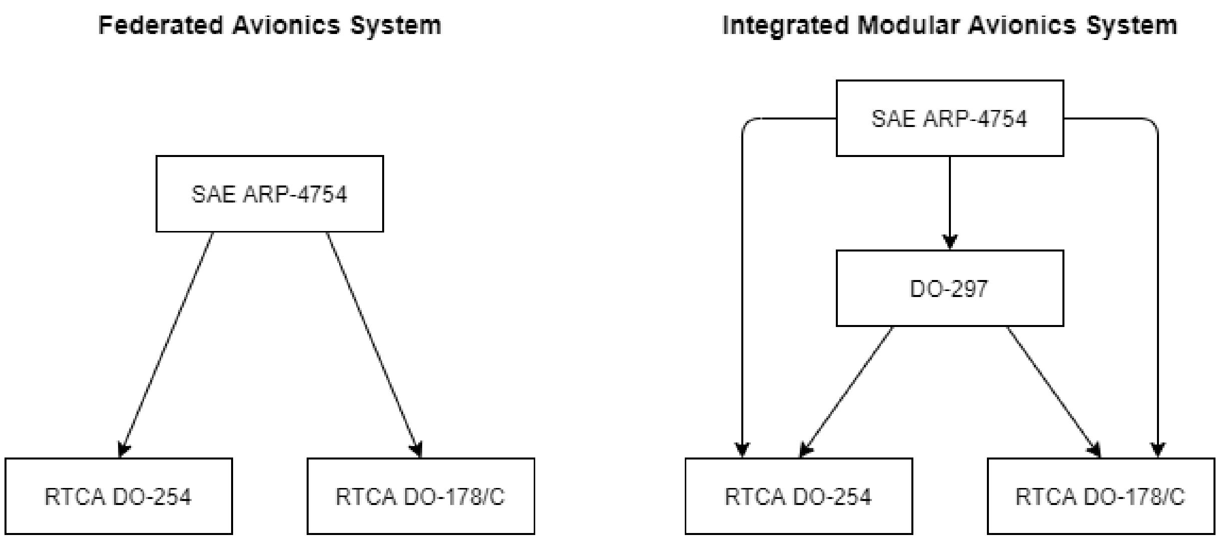

Figure 5 - Document Hierarchies

The Integrated Modular Avionics (IMA) hierarchy will be adopted for the remainder of this section but the same proposals could be applied to the federated avionics systems hierarchy.

9.2.1 New Document Hierarchy

Taking into account the need for the development process for a VTE to have a platform definition as an input, a new document hierarchy would be:

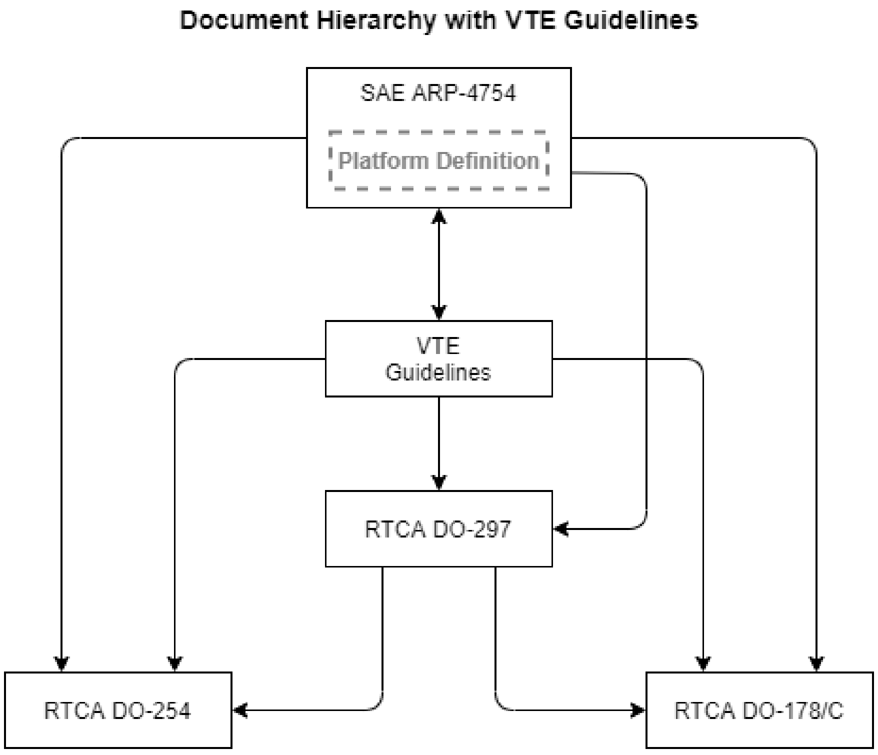

Figure 6 - New Document Hierarchy

The above diagram is further explained by the subsequent sections.

9.2.2 ARP 4754

The most evident change needed to this document is to add the concept of a platform definition that serves as the top-level specification for the aircraft system.

The document should also more strongly emphasize the need for a consolidated ICD that is used throughout the product and VTE development processes (and also used by the platform definition, or part of it).

References to the new virtual testing document are also needed so that the VTE development process can also be introduced.

In general, the use of virtual testing in the product development process needs to be integrated into the current guidelines.

9.2.2.1 Platform Definition Template

Since the VTE and the System Specification will both need to be compatible and, as much as possible, equivalent to each other, there is no room for ambiguity.

It is quite possible, if not probable, that the persons responsible for the product development of a project are not the same as those who develop the VTE for that same project. So any descriptive and vague definition of a platform is subject to being interpreted differently.

The platform definition is tentatively included within *[SAE-ARP 4754]* on the above diagram. This is to illustrate that it should probably be added as an annex and the main body of the document only be updated to make reference to it.

Documents such as ***ARINC 653 – Avionics Application Software Standard Interface*** go so far as to specify a XML template for the detailed definition of a module of a computing platform that hosts several software partitions and inter-communicating data channels.

The proposal here is that *[SAE-ARP 4754]* should recommend that the platform definition for a project should not only have descriptive text to say what it is, but it should also be encapsulated by a precise XML definition, similar to that of an ARINC 653 module.

The details of such a template are almost certainly going to be project specific but to illustrate the idea, here is part of a possible representation of the example platform definition used in this document:

```
<Aircraft Name="A400M">
     <Subsystems count="7">
          ...
          ...
          <!--    Fault Management -->
          <Subsystem
               ID="LRU_1"
               Provider=”ACME”
               Function="Fault Management"
               VTT="SIL"
               <ExecPeriodsMs 1="100">
               <Channels count="1">
                    <Channel="lgcu_to_fault_manager"/>
               </Channels>
          </Subsystem>
          <!--    LGCU -->
          <Subsystem
               ID="LRU_7"
               Provider=”ACME”
               Function="LGCU"
               VTT="VPIL"
               <ExecPeriodsMs 1="100" 2="5">
```

```
                <Channels count="6">
                    <Channel="lgcu_to_fault_manager"/>
                    <Channel="lgcu_to_cockpit_display"/>
                    <Channel="lgcu_to_analogue">
                    <channel="analogue_to_lgcu">
                    <Channel="lgcu_to_discrete">
                    <Channel="discrete_to_lgcu">
                </Channels>
            </Subsystem>
        </Subsystems>
        <!--    Global Data Channel Definitions -->
        <DataChannels count="24">
            <DataChannel
                Name="lgcu_to_fault_manager"
                Source="LRU_4"
                Dest="LRU_1"
                PeriodMs="100"
                SizeBytes="128">
            </DataChannel>
            ...
            ...
        </DataChannels>
</Aircraft>
```

It is not the place of ***[SAE-ARP 4754]*** to precisely define the actual parameters of a platform definition of a project.

[SAE-ARP 4754] should only emphasize the need for such a precisely defined platform for any given project.

The new annex to the document could provide an example XML platform definition to illustrate the concept (as has this document, above).

9.2.2.2 Previously Identified Changes

The work previously done in ***[D01.a]*** identified specific gaps in ***[SAE-ARP 4754]*** that can also provoke change proposals (refer to that that document to put these into context):

Section 7.6.1:

More detail about the modelling validation method would be useful.

Section 8.4:

Elaborate upon the "other purposes" for which model-based testing may be used and discuss how such things as auto-generated code can feed into the lifecycle.

Section 8.4.3:

No mention of other virtual testing methods, only the use of models.

9.2.3 DO-178/C

The first change needed is to incorporate the platform definition into the activity for developing a system specification.

There must also be discussion of VTTs throughout all the activities of the development process. That means consideration of VTTs when generating requirements, making design decisions, and how to use them for verifying the requirements.

This means not only developing a subsystem that is more testable with VTTs, but also using VTTs to assist in the process. For example using MIL to validate requirements before entering into the design activity, or using SIL for early prototyping to help make design decisions.

The idea of assigning a DAL to each requirement rather than the entire CSCI should be discussed and how to subsequently justify the use of a VTT on a requirement-by-requirement basis to gather certification evidence.

It is most probable that each project will have to present an argument for which VTT method is used to test requirements of a particular DAL. ***[RTCA DO-178/C]*** may be able to provide some guidelines in how to do this, what are the issues that need to be considered?

There is a large ambiguity in the document when it comes to the testing of different DALs. It only really states that DAL A must be tested on the real target, although there are possible contradictions to this in the document also. So the testing of different DALs must be more clearly described incorporating VTT methods.

9.2.3.1 Previously Identified Changes

The work previously done in ***[D01.a]*** identified specific gaps in ***[RTCA DO-178/C]*** that can also provoke change proposals (refer to that that document to put these into context):

Section 5.4.2:

Why does test activity b (software integration) allow non-target computer platforms which seems to contradict the objective (of using target hardware)?

Section 6.3 & 6.4:

Need a more detailed definition of "compatible"? Does it mean that all possible tests have been run on the target, or just a subset specific to that target? Or something else?

Section 6.4:

When testing a single LRU, does "target computer environment" mean only that LRU or all the real LRUs that communicate with the LRU under test?

Section 6.4:

What is meant by the term "correct operation of the software"? How to demonstrate this?

Section 6.4.1:

Cohesion between the test objectives / test activities and the testing environment is missing.

Appendices A-3, A-4, A-6:

No consideration of test environments for software levels.

9.2.4 DO-254

As is the case for ***[RTCA DO-178/C]***, the document will also need to incorporate the platform definition into the activity for developing a system specification.

Many of the points raised in ***section 9.2.3***, can also be applied here but whereas software development can make use of all the VTT methods, the VTT methods of most interest to the development of a hardware system would be MIL and HIL. So there should be more emphasis on these methods.

However, if the hardware project will make use of HDL code to program a FPGA, for example, there is also the possibility to use SIL and VPIL to test the VHDL code. In which case the document could also refer to ***[RTCA DO-178/C]*** when applying these VTT methods.

For a MBSE project, the document could encourage the use of the VTT MIL method not only for testing, but also for early validation of requirements and prototyping for the design activity.

If MIL is to be used for gathering certification evidence, the document will need to provide guidelines on the user can show that the model is representative of the real hardware function.

9.2.4.1 Previously Identified Changes

The work previously done in ***[D01.a]*** identified specific gaps in ***[RTCA DO-254]*** that can also provoke change proposals (refer to that that document to put these into context):

Section 4.1:

A test environment could be chosen and defined as part of the planning process.

Sections 5.1 & 5.2:

No mention of modelling to facilitate requirements or design elaboration.

Section 6.3.1:

Some discussion about test environments and specifically "non-intended operational environments" would be useful here.

Section 6.3.2:

What if models were used to generate HDL for the item?

9.2.5 DO-297

As for the other documents, the platform definition can be incorporated into this document. Not as part of the development process but because it will provide direct input into the definition of the IMA system that is to be developed.

For example, the ICD (whether part of the platform definition or not) will define all of the communication channels that the IMA system will need.

Similarly the platform definition itself will describe all of the communication endpoints (LRUs), execution periods of the functions, and any global system configuration modes that could map to different ARINC 653 operating modes.

In fact, much of the platform definition contents will be repeated in one or several IMA module configurations. So rather than have multiple copies of the same data, the document could recommend that the platform definition should be able to be reused by the Operating System for the IMA project. *

***	This does not necessarily mean that they use the same schema. Tools could be used to automatically convert the data between different schema. The key point is that the parameters defined in the platform schema are compatible with those needed by the OS.***

[RTCA DO-297] is not really concerned with the development process or assuredness levels of the system. It is more about promoting re-usability of the system and how to avoid retesting the entire product when incremental changes are introduced by promoting a modular approach.

This document has previously discussed how a VTE may be configurable and therefore re-usable between projects. This could also be highlighted in ***[RTCA DO-297]*** and the principles and philosophy that it promotes could be applied equally to the development of a VTE.

9.2.5.1 Previously Identified Changes

The work previously done in ***[D01.b]*** identified specific gaps in ***[RTCA DO-297]*** that can also provoke change proposals (refer to that that document to put these into context):

Section 2.1.2:

When defining the scope of the incremental acceptance tasks, some consideration of VTT environments could be added.

Section 2.2:

The IMA platform development process could encourage the modules to be designed into two categories: target platform-dependent and independent (re-usable in VTTs).

Section 2.2:

Why is there no significant consideration of “re-use” of acceptance data between VTTs and the target platform? Only re-use between different aircraft is considered.

Section 2.3.1:

The IMA Platform process could promote the use of data signal concentrator modules/applications to facilitate platform-independence of applications which may allow re-use of VTT based application verification data.

10 ACRONYMS & ABBREVIATIONS

AC/0	Aircraft Zero (testing rig)
AFDX	Avionics Full-Duplex Switched Ethernet
API	Application Programming Interface
ARINC	Aeronautical Radio, Incorporated
ARP	Aerospace Recommended Practice
CAST	Certification Authority Software Team
CSCI	Computer Software Configuration Item
DAL	Design Assurance Level
EASA	European Aviation Safety Agency
FAA	Federal Aviation Authority
HDL	Hardware Description Language
HIL	Hardware In-The-Loop virtual testing methodology
HLR	High-Level Requirement
IDAL	Item Development Assurance Level
INTA	Instituto Nacional de Técnica Aerospacial
I/O	Input/Output
IMA	Integrated Modular Avionics
LLR	Low-Level Requirement
MBSE	Model-Based Systems Engineering
MIL	Model In-The-Loop virtual testing methodology
OS	Operating System
RTCA DO	Radio Technical Commission for Aeronautics DOcument
SAE	SAE International (formerly Society for Automotive Engineers)
SIL	Software In-The-Loop virtual testing methodology
VPIL	Virtual Processor In-The-Loop virtual testing methodology
VTE(s)	Virtual Testing Environment(s)
VTT(s)	Virtual Testing Technology(ies)

End of Document.