



DELIVERABLE

D2.2 – Methodology and Workflow

Project Title	COMP4DRONES
Grant Agreement number	826610
Call and topic identifier	H2020-ECSEL-2018
Funding Scheme	Research & Innovation Action (RIA)
Project duration	36 Months [1 October 2019 – 30 September 2022]
Coordinator	Mr. Rodrigo Castiñeira (INDRA)
Website	www.COMP4DRONES.eu

Document fiche	
Authors:	See Page 4
Internal reviewers:	Eugenio Villar [UNICAN] Fedor Ester [DEMCON]
Work Package:	WP2
Task:	T2.2
Nature:	R
Dissemination:	PU

Document History			
Version	Date	Contributor(s)	Description
V0.8	25/05/2020	Mahmoud Hussein, Reda Nouacer	First draft of the deliverable
V1.1	01/07/2020	See Page 4	Complete draft of the deliverable ready for review
V1.3	10/07/2020	Eugenio Villar Fedor Ester	Internal review process
V1.5	28/07/2020	Mahmoud Hussein, Reda Nouacer	Final Version of the deliverable

<p>Keywords:</p>	<p>Methodology, Agile, Workflow, Qualification, Architecture, Safe-decision, Trusted communication, Tools</p>
<p>Abstract (few lines):</p>	<p>This deliverable provides an initial COMP4DRONES methodology and workflow. First, as the project is in its early stages, methodologies for requirements collection and measuring the project success criteria are provided. Second, we introduce a set of key concepts that are needed to define the overall project methodology such as drone system development process, drone categories, U-Space, and SORA (Specific Operations Risk Assessment). Third, the needs for developing drone systems from different perspectives are presented. Forth, the states of the system engineering methodologies for developing avionics software are described. Finally, our early vision of a reuse-based agile methodology for the development of drone system, and the challenges and expected project contributions to improve the existing technologies to enable ease customization of drones, and their safe operation are presented.</p>

DISCLAIMER

This document does not represent the opinion of the European Community, and the European Community is not responsible for any use that might be made of its content. This document may contain material, which is the copyright of certain **COMP4DRONES** consortium parties, and may not be reproduced or copied without permission. All **COMP4DRONES** consortium parties have agreed to full publication of this document. The commercial use of any information contained in this document may require a license from the proprietor of that information.

Neither the **COMP4DRONES** consortium as a whole, nor a certain party of the **COMP4DRONES** consortium warrant that the information contained in this document is capable of use, nor that use of the information is free from risk, and does not accept any liability for loss or damage suffered by any person using this information.

ACKNOWLEDGEMENT

This document is a deliverable of **COMP4DRONES** project. This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement N° 826610

D2.2 Authors:

Partner Name	Contributors
INDRA	Rodrigo Castiñeira, Adrian Irala
ENAC	Fabien Bonneval, Gautier Hattenberger
SCALIAN	Raphaël Lallement
ENSMA	Yassine Ouhammou, MatheusLadeira
ABI	Katiuscia Zedda, Giuseppe Tuveri
UNIMORE	Andrea Marongiu, Alessandro Capotondi
UNISANNIO	Giuseppe Silano, Luigi Iannelli, Valerio Mariani
UNISS	Francesca Palumbo, Tiziana Fanni
TEKNE	Carlo Tieri
TOPVIEW	Alberto Mennella
TNL	Maurits De Graaf
SkyA	Philipp Knopf
ACORDE	Fernando Herrera, David Abia
CEA	Mahmoud Hussein, Reda Nouacer, Fabio Arnez, Ansgar Radermacher, Huascar Espinoza, Chokri Mraidha
AIK	Giovanni Marco Cordella
ROT	Niccolò Cometto, Diego Grimani, Maria Luisa Scalise
MODIS	Daniela Parletta
ALTRAN	Soniya Nibhani, Vincent Bompert

Table of Contents

DEFINITIONS, ACRONYMS AND ABBREVIATIONS	8
EXECUTIVE SUMMARY	10
1 INTRODUCTION	11
2 METHODOLOGY FOR COLLECTING REQUIREMENTS AND EVALUATING KPIS	12
2.1 REQUIREMENTS COLLECTION METHODOLOGY	13
2.1.1 <i>Systems Engineering (SE) benefits</i>	13
2.1.2 <i>Functional analysis methodology</i>	13
2.1.3 <i>RBE and MBSE benefits</i>	14
2.1.4 <i>Proposed methodology and activities</i>	15
2.2 AN APPROACH FOR MEASURING COMP4DRONES OBJECTIVES.....	16
3 U-SPACE, DRONE CATEGORIES, AND SORA	18
3.1 PROCEDURE FOR DEVELOPING UAS	18
3.2 U-SPACE.....	19
3.3 DRONE CATEGORIES	20
3.4 SORA: SPECIFIC OPERATIONAL RISK ASSESSMENT	22
4 NEEDS FOR DRONE SOFTWARE DEVELOPMENT	25
4.1 END-USERS REQUIREMENTS.....	25
4.1.1 <i>Flying Stages/Operations</i>	25
4.1.2 <i>Mission Specific Operations</i>	26
4.2 DRONE SYSTEM NEEDS	27
4.2.1 <i>Drone Customization</i>	27
4.2.2 <i>Autonomous Flying</i>	27
4.2.3 <i>Drone Safety</i>	28
4.2.4 <i>Drone Communication</i>	28
4.2.5 <i>Minimizing Drone System Development Effort</i>	28
4.2.6 <i>Drone Platform and Execution</i>	29
4.3 REGULATIONS CONSTRAINTS (SORA, EASA, U-SPACE).....	29
5 STATE OF THE ART SYSTEMS ENGINEERING APPROACHES IN AVIONICS DOMAIN	31
5.1 SYSTEM ENGINEERING PROCESS	31
5.1.1 <i>Standards for Avionics System Engineering Process</i>	32
5.1.2 <i>DO-178C Software Development Lifecycle</i>	33
5.1.3 <i>Challenges in Applying DO-178C</i>	34
5.2 APPROACHES TO SPEED-UP AVIONICS SOFTWARE DEVELOPMENT.....	35
5.2.1 <i>Airbus Agile Development Process</i>	35
5.2.2 <i>A Model-Based Agile Process for DO-178C Certification</i>	36
5.2.3 <i>Towards a DO-178-Aligned Agile Approach</i>	37
5.2.4 <i>Agile for Aerospace</i>	39
5.2.5 <i>Gains from Agile Development Methodology</i>	40
6 COMP4DRONES METHODOLOGY AND WORKFLOW	41
6.1 C4D REUSE-BASED AGILE DEVELOPMENT PROCESS WORKFLOW	42
6.1.1 <i>Conflicts Between Agile and DO-178C and their Resolution</i>	43
6.1.2 <i>C4D Development Process</i>	44
6.1.3 <i>A System Composition Approach</i>	46
6.2 DRONE EMBEDDED SOFTWARE.....	49
6.2.1 <i>Challenges in Existing Approaches/Practices</i>	49
6.2.2 <i>Specification/Guidelines</i>	49
6.3 MINIMIZING THE DESIGN AND VERIFICATION EFFORT	51
6.3.1 <i>Challenges in Existing Approaches/Practices</i>	51

6.3.2	<i>Specification/Guidelines</i>	52
6.4	SAFE AUTONOMOUS DECISIONS.....	54
6.4.1	<i>Challenges in Existing Approaches/Practices</i>	54
6.4.2	<i>Specification/Guidelines</i>	54
6.5	TRUSTED COMMUNICATION	57
6.5.1	<i>Challenges in Existing Approaches/Practices</i>	57
6.5.2	<i>Specification/Guidelines</i>	57
7	CONCLUSION	60

Table of Figures

Figure 1: Example of functional analysis with Octopus diagram	14
Figure 2: System Requirements decomposition	14
Figure 3: Model layers in MBSE representation	15
Figure 4: C4D requirements collection methodology	15
Figure 5: Goal-Question-Metric approach	17
Figure 6: COMP4DRONES implementation of Goal-Question-Metric approach	17
Figure 7: Step by step procedure for developing UAS.....	19
Figure 8: U-Space services.....	20
Figure 9: Drone Classification	21
Figure 10: The SORA process	24
Figure 11: The different stakeholders and their viewpoints of the requirements	25
Figure 12: V-Model of a Conventional, Large-System Development Process.....	32
Figure 13: Standards for Avionics System Engineering Process	33
Figure 14: DO-178C software development lifecycle.....	33
Figure 15: Airbus agile development process.....	36
Figure 16: Application of Scrum phases to DO-178 processes.....	38
Figure 17: Agile DO-178	39
Figure 26: Software processes compared	40
Figure 18: The overall work flow of the COMP4DRONES project	42
Figure 19: Reuse-based agile development process.....	45
Figure 20: Reuse-based agile development process workflow	45
Figure 21: A structure for system composition has requirements originating from composability, composition workflow, and support via tooling	46
Figure 22: Occurrence of composability	47
Figure 23: The composition workflow ³⁸	47
Figure 24: Stakeholders collaborating and interacting in an ecosystem ³⁸	48
Figure 25: Adequate support via tooling for participants is critical towards system composition ³⁸	48
Figure 27: Modular drone Embedded COMP4DRONES Reference Architecture	50
Figure 28: Agile and interoperable COMP4DRONES tool ecosystem (WP6)	52
Figure 29: Safe Autonomous Decision	55
Figure 30: COMP4DRONES trusted communication framework: key properties	58

Table of Tables

Table 1: DO-178C Problems and Suggested solutions	43
---	----

Definitions, Acronyms and Abbreviations

Acronym	Title
Air traffic management (ATM)	Consists primarily of air traffic control, air traffic flow management and aeronautical information services.
Command and control (C2) link	Data link between the drone and the remote pilot station, which manages the flight.
DAI/DAS	Data Acquisition Interface, Data Acquisition System
Dependability	Dependability is the ability to provide services that can defensibly be trusted within a time-period. It is also a measure of a system's availability, reliability, and its maintainability.
DoS	Denial of Service
Drone	See UAV
'Detect and avoid' technology	Capability of the drone to remain at safe distance from, and to avoid collisions with other aircraft.
FPGA	Field-programmable gate array: programmable hardware
Geofencing	Software using GPS signals to stop drones flying into certain areas.
GNSS	Global Navigation Satellite System receivers, using the GPS, GLONASS, Galileo or BeiDou system
IMA	Integrated Modular Avionics, a system architecture enabling to run multiple avionic functions on a single device.
LIDAR	Measure distance to a target by illuminating it with pulsed laser light and measuring the reflected pulses with a sensor.
LTE	Long-Term Evolution (LTE) is a standard for high-speed wireless communication for mobile devices and data terminals.
LoRa	Long Range (and low power communication technology)
LPWAN	Low Power Wide Area Network
MIMO	Multiple input multiple output: wireless middleware.
NB-IoT	Narrowband Internet of Things
MIMO	Multiple input multiple output: wireless middleware.
Precision agriculture	A farming management concept based on observing, measuring and responding to variability in crops to reduce resource consumption.
QoS	Quality of Service, performance properties of a service
ROS	Robot Operating System. Widely used operating system in robotics and drone domain.
RPAS	Remotely Piloted Aircraft System
Remote pilot	Person who is in control of the flight path of the aircraft.
Safety	The condition of being protected from or unlikely to cause danger, risk, or injury.
Security	The protection of systems and networks from the theft of or damage to their hardware, software, or electronic data, as well as from the disruption or misdirection of the services they provide.
Reliability	The probability that a system will produce correct outputs up to some given time t.
Segregated airspace	Airspace of specified dimensions assigned for exclusive use to specific users.

SoC	System-on-chip. Multiple circuits on a single, integrated chip (IC), e.g. processor, I/O controllers and memory.
UAS	Unmanned Aircraft System
UAV	Unmanned Aerial Vehicle
USV	Unmanned Surface Vehicle
UTM	UAS Traffic Management
V2I	Vehicle to Infrastructure

Executive Summary

Drones/UAVs can perform air operations that manned aircrafts struggle with. Their use brings significant economic savings and environmental benefits whilst reducing the risk to human life. Drone-based service and product innovation, as driven by increased levels of connectivity and automation, is limited by the growing dependence on poorly interoperable proprietary technologies and the risks posed to people, to other vehicles and to property (see the deliverable D2.1). Therefore, the aim of the **COMP4DRONES** project is to provide an **engineering environment** to support the **development and customization** of the drone embedded platform.

In this deliverable, we will provide an **initial COMP4DRONES methodology and workflow**. First, as the project still in its early stages (i.e. the more focus is on use-cases and requirements collection), we introduce methodologies for **requirements collection** and for **measuring the project success criteria**.

Second, we introduce a **general procedure** for developing drone systems, and a set of **key concepts** that are needed to specify the project methodology and workflow. These concepts include **U-Space**, **drone categories**, and **SORA** (Specific Operations Risk Assessment).

Third, the drone system has different stakeholders with different needs. Thus, the **requirements/needs for developing drone systems** are described from the users' perspective as a high-level summary of D1.1, and from the perspectives of service providers and system integrators.

Forth, **state of the art system engineering approaches** in the avionics domain are presented where they provide useful insights and recommendations to specify the **COMP4DRONES** methodology.

Finally, an initial methodology for drone systems development is presented. The methodology is based on **reuse** and **agility** to speed the system development and its qualification. This methodology will be detailed and elaborated in the coming deliverables (D2.3 and D2.4). In addition, the project **challenges** and its **contributions** to improve the existing technologies to enable easy customization of drones, and their safe operation have been described. The improvements are divided into four groups: integrated modular reference architecture, minimization of the system design and verification, safe autonomous decisions, and trusted communication.

1 Introduction

Unmanned Aircraft Systems (UASs) are envisioned in a wide range of applications and functions in smart cities. Drones can be used for delivering packages to the customer's door, monitoring and assisting in the event of (medical) emergencies, as well as for monitoring traffic and key infrastructures. However, challenges such as safety, security and privacy in densely populated regions remain concerns in connecting drones to smart cities. The major challenges facing large adoption of UASs are:

- **Regulation:** The development of the UAS market highlights critical regulatory and certification challenges that must be addressed in order to bring UAS into mainstream consumer use. There have been recent promising developments in Europe with the publication of (1) Commission Implementing¹ Regulation (EU) 2019/947 of 24 May 2019 on the rules and procedures for the operation of unmanned aircraft, (2) Commission Delegated² Regulation (EU) 2019/945 of 12 March 2019 on unmanned aircraft systems and on third-country operators of unmanned aircraft systems, and (3) the VTOL certification pioneered by EASA through the issuance of a Special Condition³ on 2 July 2019.
- **Airspace management:** Maintaining an increasingly diverse airspace while keeping all air traffic moving safely and efficiently will be a significant challenge. A key enabler for the future of drones will be Unmanned Aircraft System Traffic Management (UTM) systems, which will need to work in conjunction with existing air traffic control (ATC). UTM requirements will vary by altitude and location. A new definition of Very Low Level (VLL) airspace volumes has been proposed by the U-space CORUS project, with the introduction of X, Y, Z_u and Z_a airspace volumes defined in the final CONOPS⁴.
- **Infrastructure:** UAS can become a practical and widely used component of urban and suburban mobility only when eVTOLs are integrated with overall city transportation networks along with well-developed ground-based facilities and efficient air traffic management and robust communication systems. Infrastructure requirements include suitable take-off and landing areas, as well as parking and battery charging stations.
- **Technology Advances:** There are various technology considerations for VTOL/UAS manufacturers. These aircraft will require advanced technologies, such as artificial intelligence and cognitive systems, to enable advanced detection and avoidance capabilities. Machine learning could be essential as operations move from piloted to autonomous.
- **Safety:** Drones deployed in highly dense cities for various civilian applications raise serious safety issues as large amounts of damage can be caused due to the crashing of drones. This may be the result of technical malfunctions or the inadequate maintenance of equipment, mid-air collisions, or misuse by their operators. Severe weather conditions, turbulence (around buildings), lightning and battery life capacity have also triggered concerns about drones falling on congested areas of cities, towns, or settlements. Furthermore, because airspace must be shared with civil services and sometimes commercial aviation in larger cities, there is also a serious risk of airborne collisions leading to widespread destruction.
- **Security:** The biggest security concern of using drones is that the drones could be hijacked or destroyed by attackers, causing not only disruption of the services they provide, but also present a safety issue. Drone navigation and communication modules are vulnerable to different kinds of security breaches. GPS enables a drone's navigation system, and due to the open nature of unencrypted and unauthenticated GPS signals, they can be easily spoofed. WIFI jamming is another possible attack that could cause the loss of control of the drone's communication system

¹https://eur-lex.europa.eu/eli/reg_impl/2019/947/oj

²<https://eur-lex.europa.eu/legalcontent/EN/TXT/PDF/?uri=CELEX:32019R0945&from=EN>

³<https://www.easa.europa.eu/sites/default/files/dfu/SC-VTOL-01.pdf>

⁴<https://www.eurocontrol.int/project/concept-operations-european-utm-systems>

with serious consequences. The concerning aspect of such attacks is that they can be carried out with off-the-shelf products. In some scenarios, a simple mobile device is sufficient to attack the drone if it is operating with an unsecured WIFI network. On the other hand, a new promising service that will be deployed by the European GNSS Galileo (OS-NMA⁵ downstream service) will let GNSS receivers ensure that the satellite signals they receive are, indeed, from Galileo satellites, and that they have not been modified. The approach makes it more difficult for hackers and other bad actors to spoof GNSS receivers by feeding them fraudulent signals.

- **Ethics and Privacy:** Drones utilize a large variety of sensing technologies, such as high-resolution cameras, sensors, and other recording devices that create large flows of possibly personal data, especially as they can be controlled remotely. This data processing and the analytics used raise important questions and concerns regarding privacy and data protection. It is important that privacy by design principles are utilized to ensure that the rights of individuals are respected, and that security safeguards are in place to prevent access to this data by malicious actors. This will yield a value-sensitive design for the technologies developed within this project, which will increase levels of trust, user acceptance, and public acceptance.
- **Propulsion and energy source:** The great majority of UASs use batteries to power the on-board systems and motors. The advantages are no local emissions, reduced noise levels and easy thrust control. Although batteries are generally more efficient, they are heavy and remain part of the aircraft for the entire flight. They currently represent the most limiting factor for endurance. Energy management is crucial: carrying a sufficient load of energy to perform the mission, maintaining a safety margin and recharging for the next flight.

The aim of the **COMP4DRONES** project is to provide a framework of **key enabling technologies** for safe and autonomous drones. In particular, **COMP4DRONES** will leverage composability and modularity for customizable and trusted autonomous drones for civilian services. The project will take into account recent regulation developments in this area from EASA and, by extension, JARUS. One of the main rules directly linked to **COMP4DRONES** is “EASA has proposed a risk-based approach to settle a performance-based framework for regulation related to drones”. We will also consider the SESAR-JU studies concerning civilian drones, and will adhere to the U-space approach and protocols. To support the **COMP4DRONES** framework development, the project will provide an **engineering methodology**. The methodology will be based on **reuse** and **agility** to speed the system development and its qualification.

In the following sections, we start, in Section 2, by describing methodologies for requirements collection and measuring the project success criteria as the project still in its early stages. Then, we describe some concepts and general procedure for developing drone systems (i.e. Section 3), and presenting the requirements/needs for drone systems development in Section 4. Then, the state of art engineering methodologies for the avionics domain is presented in Section 5. Finally, we introduce the project initial methodology (Section 6.1), and we indicate in detail the challenges and guidelines for the main ambitions that will be addressed by the **COMP4DRONES** consortium as follows: integrated modular reference architecture for Drones (Section 6.2), minimizing the design and verification effort (Section 6.3), safe autonomous decisions (Section 6.4), and trusted communication (Section 6.5).

2 Methodology for Collecting Requirements and Evaluating KPIs

As the **COMP4DRONES** project still in its early stages (i.e. the more focus is on use-cases and requirements collection), in this section, we describe methodologies for requirements collection, and for measuring the key performance indicators (KPIs).

⁵<https://insidegnss.com/what-is-navigation-message-authentication/>

2.1 Requirements Collection Methodology

In current phase of the project, the interest/focus is on collecting the drone system requirements from the different use-cases (demos). Thus, in this section, we describe the **COMP4DRONES** methodology for the requirements collection phase.

2.1.1 Systems Engineering (SE) benefits

The **COMP4DRONES** project combines several characteristics; necessary to provide a successful identification and design of the key enabling technologies for safe and autonomous drones, but there are potential project and technical issues if they are not addressed by the process and the methodology:

- **50 partners**, with different backgrounds, different practices, different levels of involvement. **How to understand each other?**
- Complex design items interactions: 5 use cases, 11 demonstrators, many more key enabling technologies, needs, and requirements. **How to track the relations between the design items?**
- Complex targets: every partner has its own target, with its own level of complexity. All of them must be delivered to the project. **How to ensure the quality of deliveries?**

Deploying a shared and agreed Systems Engineering process (with associated methodology and supporting tools) is a necessary step, in order to handle complexity and change, in order to reduce risk. Expected benefits of such SE process are:

- To avoid omissions (stakeholders, needs, constraints, lifecycle stage) and invalid assumptions.
- To track the requirements and the tests.
- To manage the change and the configuration.
- To compare the effective progress to the project plan.

2.1.2 Functional analysis methodology

A key step of WP2 is the identification of the key enabling technologies (KETs), as described in D2.1. This step is mandatory in order to define generic components and to propose system architecture alternatives (the aim of WP3). As for the elicitation of system requirements rising from every demonstrator (WP1, D1.1), it is recommended to elaborate this identification by performing functional analyses on every demonstrator, considered as a System of Interest:

- **Identify** all the stakeholders involved in the System of Interest (SoI) along the whole Life Cycle, with the system promoter / customer (e.g. every demonstrator or Use Case leader).
- **Interview** every stakeholder on his vision of the SoI, based on agreed and homogeneous survey forms (or request to every promoter / customer to share his understanding).
- **Consider** other possible techniques to elicit needs (workshops, observation, and documentation).
- **Identify** the main functions (or MF, purposes of the SoI).
- **Identify** the constraints imposed by each stakeholder (or CF).
- **Validate** the functional description before launching any development.

The Octopus diagram (“beast horns”, as depicted in Figure 1 below) can be used to carry out the functional analysis.



Figure 1: Example of functional analysis with Octopus diagram

2.1.3 RBE and MBSE benefits

Using a shared RBE (Requirement Based Engineering) tool is recommended in order to have a shared, single format and framework for definition, documentation, traceability and V&V of requirements. This tool should carry the upstream workflow coming from all the system requirements considered in the Use Cases (demonstrators), as well as the framework requirements from WP2. The system requirements can be decomposed as shown in Figure 2. The system requirements are divided into groups which are its sub-system. These sub-systems composed of a number of components. For each component there are functional and non-functional requirements (e.g. safety, security, performance, integrations, etc.).

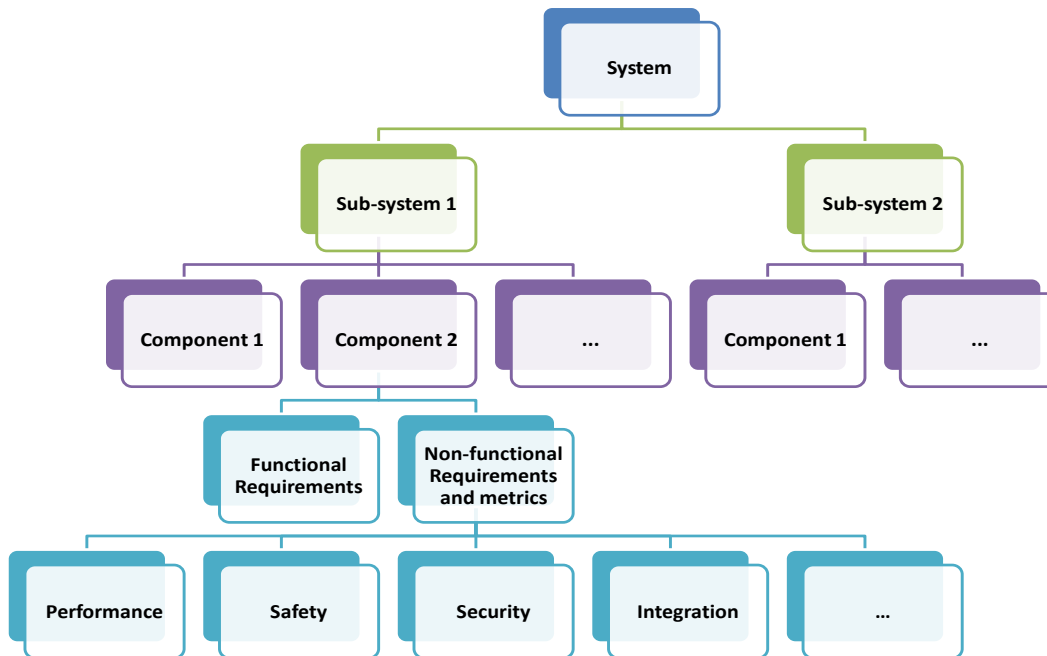


Figure 2: System Requirements decomposition

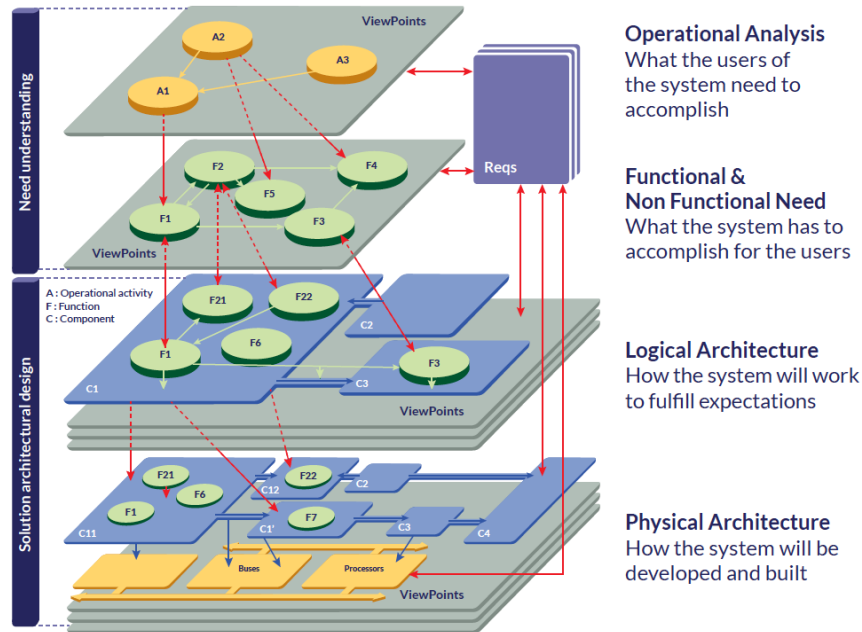


Figure 3: Model layers in MBSE representation

Beyond requirements, it is recommended to support all the software engineering (SE) process with a MBSE (Model Based System Engineering) approach that provides:

- A single representation for system data storage and maintenance.
- A singular definition for any element within the model.
- A model integration.

MBSE tools (e.g. Capella (Eclipse framework), Papyrus4Robotics (CEA), etc.):

- Allow for structuring complex needs in a graphical representation.
- Allow for tracking requirements up to the physical component parts.

Moreover, this kind of tool is used with version control and management software such as Git or SVN.

2.1.4 Proposed methodology and activities

The proposed methodology for requirements collection for the **COMP4DRONES** project is shown in Figure 4. It has four steps: performing functional analysis using a template, design a system architecture, track the requirements, and then re-use or develop software components.

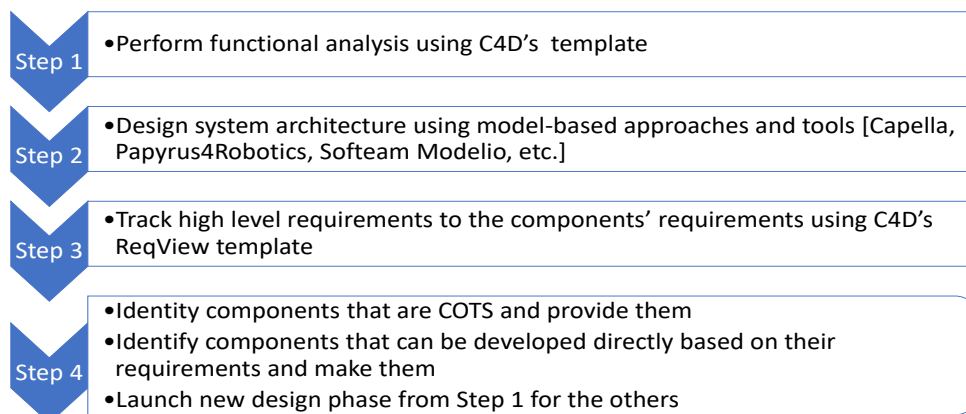


Figure 4: C4D requirements collection methodology

2.2 An Approach for Measuring **COMP4DRONES** Objectives

Not only requirements are collected at this stage of the project, but also the project key performance indicators are specified. Thus, in this section, we describe the overall **COMP4DRONES** success criteria associated with the different project technical objectives, and an approach to measure the project objectives.

The overall success criteria seek to demonstrate measurable gains for the **COMP4DRONES** users, in terms of efficiency, risks, reuse, and sustainability, by comparing estimated levels at the end of the project with the levels at the beginning. The numbers used in the success criteria are derived from internal discussions among the partners and estimations coming from practical experience in the field and from the state of the art. The following are the project objectives and their success criteria:

Objective 1: Easing the integration and customization of drone embedded systems

- SC1.1. Demonstrate a potential gain for design efficiency of drone embedded platforms by reducing their integration, customization and maintenance efforts by 35%.
- SC1.2. Demonstrate a potential reuse of pre-qualified drones' application components, leading to 35% of effort reduction for system-level assurance activities.

Objective 2: Enabling drones to take safe autonomous decisions

- SC2.1. Demonstrate a potential raise of technology innovation led by increasing autonomy for 40% of dull, dirty, dangerous, and difficult tasks with an acceptable level of safety.
- SC2.2. Increase safety during mission execution by 35%

Objective 3: Ensure the deployment of trusted communications

- SC3.1. Demonstrate a potential raise of technology trustworthy led by 30% reduction of cyber-security risks of drone communications.

Objective 4: Minimizing the design and verification efforts for complex drone applications

- SC4.1. Demonstrate a potential gain for design and assurance efficiency of drones embedded platforms by reducing specification, verification & validation and implementation efforts by 30%.
- SC4.2. Demonstrate a potential gain for certification of drones embedded systems by reducing the certification effort by 25%.

To measure the progress of the project and to evaluate the effectiveness of the proposed technologies (i.e. assuring the success criteria are met), **COMP4DRONES** follows the Goal-Question-Metric approach shown in Figure 5⁶. This approach has been widely used for product and process assessment, including improvement assessment.

For improvement assessment, we have specified qualitative indicators, notably Low, Medium and High. The associated quantitative value, e.g. an effort, is criterion dependent. The important value for the assessment is the deviation between the target value and the actually achieved value at the end of the project. We measure this achievement in percentage, i.e. 0% corresponds to no achievement (achieved value is identical to current value, i.e., value at the beginning of the project) to 100% (achieved value is identical to target value). In some cases, we express relative target improvements and can compare these with the achieved actual relative improvements. This way of characterizing the current and target situations allows us to (1) estimate the improvement quantitatively based on qualitative information, (2) mitigate uncertainties from only and directly using quantitative values (e.g., due to insufficient information about the current status), and (3) provide indicators that overall represent **COMP4DRONES** success criteria in all the application domains addressed in the project. Finally, the envisioned targets of the overall success criteria have been derived from the average gain of their indicators.

⁶https://courses.cs.ut.ee/MTAT.03.243/2015_spring/uploads/Main/GQM_book.pdf

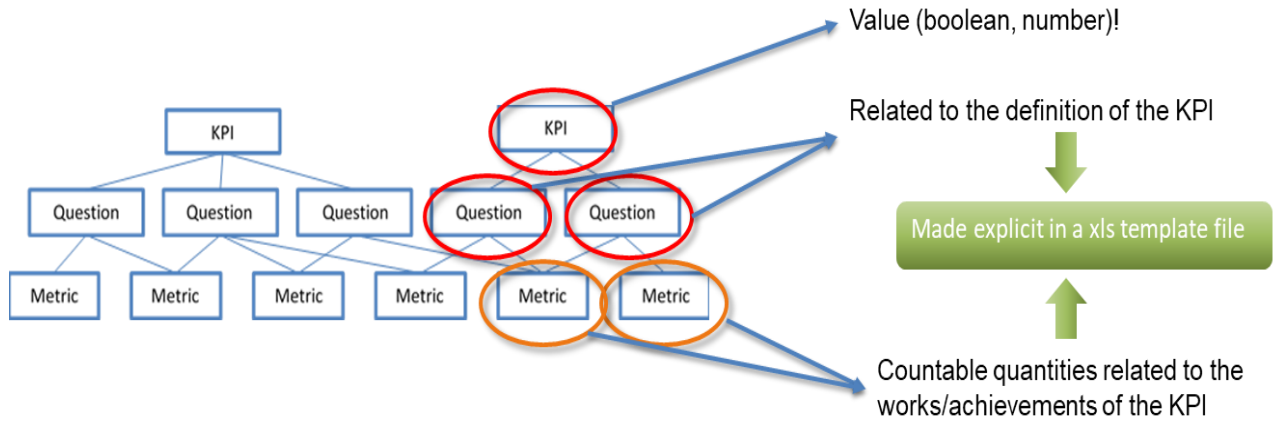


Figure 5: Goal-Question-Metric approach

Examples following the Goal-Question-Metrics approach:

- In success criteria SC1.1, let us assume that we defined a metric for system level V&V effort, i.e. associate a number in terms of costs/time what a low, medium or high level. Let us further assume that the success rate formula is $100\% - (\text{actual-target}) / (\text{current-target})$. If the initial effort for V&V validation is 40k€, the target “low” is 20k€ and the actually achieved value is 25k€, the formula results in $100\% - 5k€ / 20k€ = 75\%$.
- In SC2.2, let us assume that we enumerate a set of six different critical situations of which one is already handled at the project start and all six should be handled in the end. If we can handle five in the end, the success corresponds to $100\% - 1 / 5 = 80\%$.
- For a KPI of use case: Level of automation for inspection of offshore wind turbine structures, target is reduction of 20% of costs in monitoring. The success rate formula gets very simple in this case: $\text{actual} / \text{target}$, where actual and target are reductions in percent relative to the initial (current) value. If we actually achieve a reduction of 17%, the success rate is $17\%/20\% = 85\%$

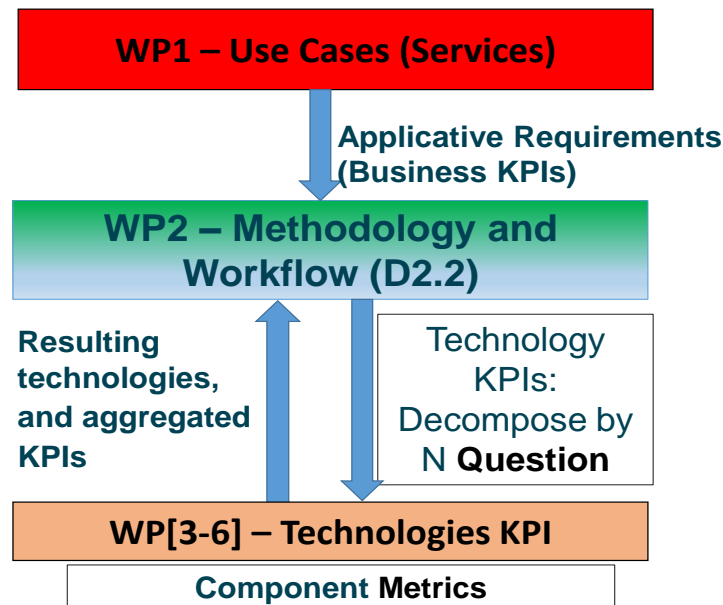


Figure 6: COMP4DRONES implementation of Goal-Question-Metric approach

There are different stockholders with different views of the system requirements and success criteria. Thus, following the Goal-Question-Metric approach, the project success criteria are specified from different viewpoints (see Figure 6). First, a number of KPIs from the business perspective are specified

from the system users' point of view in WP1. These business KPIs are then linked in WP2 with the system/project goals (i.e. technology oriented KPIs). Second, a number of questions are formulated for each project goal (i.e. goals are decomposed by questions). Then, for each question, a number of metrics are defined in the project technical work packages (i.e. WP3-6). The final questions and metrics to be used will be formulated in the deliverable D1.2 (System-Under-Test Requirements and Test System Requirements), and will be measured in the deliverable D1.4 (Evaluation Result).

3 U-Space, Drone Categories, and SORA

In order to propose a methodology and workflow for drone system development, there is a need to know the general process for developing drone systems, and to take into account some concepts about U-space, drone categories, and SORA (Specific Operational Risk Assessment). In the following, we describe the general development process and these concepts.

3.1 Procedure for Developing UAS

When considering to use a UAV technology there are two options: make or buy. The service provider can either dive into UAV operation and produce the results according to his needs or hire a professional service provider (see Figure 7). The next question that should be asked is this: is it allowed to fly the intended mission and do special requirements apply? The appropriate regulatory bodies will inform him about what types of operations are allowed and what requirements must be fulfilled.

After that, a precise definition of the mission will define the UAS^{7,8}. Therefore, it is important to collect as many parameters as possible to guide the plans. It is also important to consider what conditions the UAV is going to fly (urban area, freezing, or tropical conditions). After defining the aims and objectives, the focus can be then on the UAS. Collect all the specifications for payload (weight, power consumption, quality of results, costs, etc.) and software (system requirements, cloud solution, costs, etc.).

Then, make sure platforms are able to carry the intended payload (camera, sensor, etc.) within the mission requirements (flight-time, etc.). In addition, make sure that the data analysis software is compatible to the chosen payload. Sometimes, a software suiting the mission first can be found.

Finally, after listing all the specifications, define the platform. To achieve that, check if commercially available platforms meet the intended requirements in flight performance. If not, consider do it-yourself or customized solutions. With the UAS defined and price tags attached (do not forget costs for training, authorization, insurance and maintenance). It can now be possible to assess if the intended quality of data can be acquired in a more cost-efficient way. If all these procedures seem to be too challenging, a professional service provider can be hired for the intended tasks. Even though by outsourcing the job a compromise on flexibility and operational costs is needed, it can save –depending on the repetition rate of the task – a great deal of time and energy.

⁷<https://www.easa.europa.eu/sites/default/files/dfu/SC-VTOL-01.pdf>

⁸<https://www.sesarju.eu/sites/default/files/documents/u-space/SESAR%20principles%20for%20U-space%20architecture.pdf>

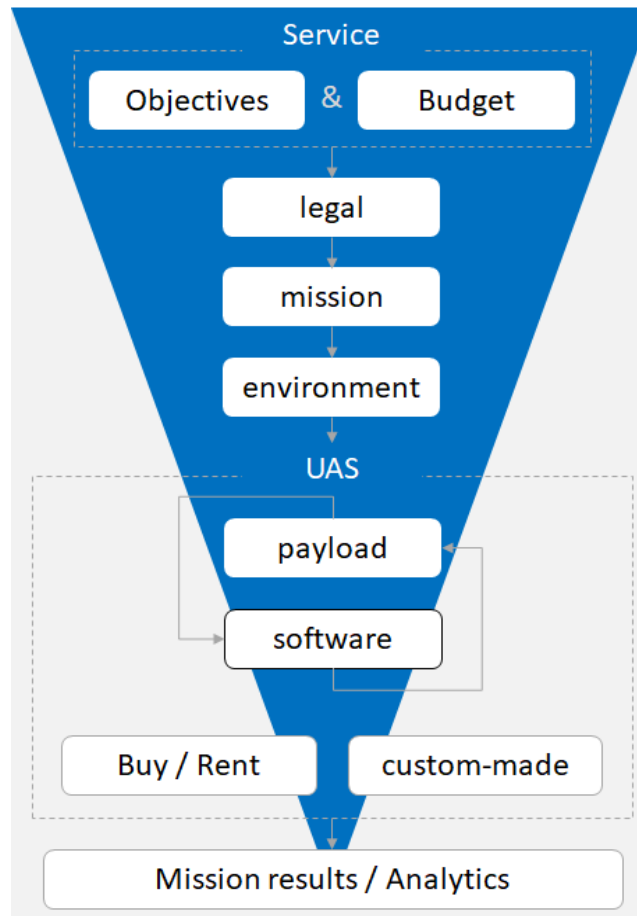


Figure 7: Step by step procedure for developing UAS

3.2 U-Space

U-space is a set of services and procedures relying on a high level of digitalization and automation of functions to support safe, efficient, and secure access to airspace for large numbers of drones (see Figure 8). It provides an enabling framework to support routine drone operations and addresses all types of missions including operations in and around airports. Ultimately, U-space will enable complex drone operations with a high degree of automation to take place in all types of operational environments.

These services rely on a high level of digitization and automation of functions, whether they are on board the drone itself, or are part of the ground-based environment. U-space provides what is needed to enable and support routine drone operations, as well as a clear and effective interface to manned aviation, Air Traffic Management (ATM)/Air Navigation Service (ANS) for service providers and authorities.

U-Space will be capable of ensuring smooth operation of drones in all operating environments, including urban areas, and in all types of airspace, in particular to very low level (VLL) airspace. It will address the need to support the widest possible variety of missions, and may concern all drone users, as well as every category of UAV, as defined by EU Commission proposed regulation on unmanned aircraft operations. According to the criticality of the provided services, performance requirements will be established for both structural elements and service delivery, covering safety, security, availability, continuity, resilience and so on.

U-Space services will be delivered by service providers within the given U-space environment. They do not replicate the function of Air Traffic Control (ATC), as known in ATM: instead, they will deliver key

services to organize the safe and efficient operation of drones and ensure a proper interface with manned aviation, ATC and relevant authorities.





 U1	Foundation	<ul style="list-style-type: none"> ▶ Registration ▶ Registration assistance ▶ e-identification ▶ Geo-awareness ▶ Drone aeronautical information management 	
 U2	Initial	<ul style="list-style-type: none"> ▶ Tracking (Position report submission) ▶ Surveillance data exchange ▶ Geo-fence provision (includes dynamic geo-fencing) ▶ Operation plan preparation /optimisation ▶ Operation plan processing ▶ Risk analysis assistance ▶ Strategic Conflict Resolution ▶ Emergency Management ▶ Incident/ Accident reporting 	<ul style="list-style-type: none"> ▶ Citizen reporting service ▶ Monitoring ▶ Traffic information ▶ Navigation infrastructure monitoring ▶ Communication infrastructure monitoring ▶ Legal recording ▶ Digital logbook ▶ Weather information ▶ Procedural interface with ATC
 U3	Enhance	<ul style="list-style-type: none"> ▶ Dynamic Capacity Management ▶ Tactical Conflict Resolution ▶ Geospatial information service ▶ Population density map 	<ul style="list-style-type: none"> ▶ Electromagnetic interference information ▶ Navigation coverage information ▶ Communication coverage information ▶ Collaborative interface with ATC
 U4	Full	<ul style="list-style-type: none"> ▶ Integrated interfaces with manned aviation ▶ Additional new servicesw 	

Figure 8: U-Space services

3.3 Drone Categories

The rapid increase in the number of civilian drones (for both leisure and commercial use) poses significant threats to the safety of the general public. The authorities' solution so far has been to impose several restrictions on the use of drones. International committees have been formed to discuss the normalization of the drones' operations (JARUS, Eurocae WG 105, GUTMA, EASA, Conseil Pour Le Drone Civil in France).

The European Aviation Safety Agency (EASA)⁹ developed and published a prototype regulation concerning the licensing and operation requirements for unmanned aircraft (UA) in August 2016. The term "Unmanned aircraft" and the abbreviation UA are used in the EASA document as opposed to e.g. "UAS" or "Drone" in the relevant Federal Aviation Administration (FAA) documents. This regulation shows the vision of the EU regarding the UAS legislation. The approach taken focuses on the risks associated with UAS operations to divide them into categories rather than quantifiable metrics (e.g. weight or size). The prototype regulation lays down:

- Rules for regulating an operation-centric concept for the operation of unmanned aircraft (UA), and more specifically in the "open" and "specific" categories (see descriptions below), within the single European sky airspace.
- Technical requirements and administrative procedures for the design, production and maintenance of UASs in the "open" and "specific" categories within the European Union, as applicable;

⁹<https://www.easa.europa.eu/domains/civil-drones-rpas/drones-regulatory-framework-background>

- Technical requirements and administrative procedures for the implementation of the concepts of registration, electronic identification, and geofencing;
- Requirements for subcategories in the "open" category;
- Conditions to issue a declaration or to obtain an authorization, as appropriate, in the "specific" category;
- Requirements for the introduction of a concept of standard scenarios in the "specific" category;
- Conditions to obtain an optional light UA operator certificate (LUC), with associated privileges;
- Conditions for the making available on the market of UASs intended to be used for operations in the "open" category, as well as requirements for market surveillance relating to the marketing of those UASs in the Union.

Following the publication of the Prototype regulation for the "open" and "specific" categories in August 2016, EASA drafted and published NPA 2017-05 on 4 May 2017.

In 2019 and 2020, three regulations were adopted:

- Regulation 2019/945 on UAS and third-country operators of UAS. This Regulation defines requirements for the design and manufacture of UAS.
- Regulation 2019/947 - rules and procedures for unmanned aircraft. This Regulation lays down detailed provisions for the operation of UAS as well as for personnel, including remote pilots and organizations involved.
- Regulation 2020/639 regarding standard scenarios for operations executed in or beyond the visual line of sight.

Current discussions lead towards requesting a safety assessment for the Specific and the certified categories, thus demanding that a few development and verification activities are enforced. Today, drones are classified based on operations risks concerns (see Figure 9). In the open category, where the level of risk is low, the required level of safety will be ensured through a set of requirements and functionalities. In the Specific category, the safety will be ensured through a standard risk assessment process. In the Certified category, the risk is similar to current manned aviation operations, and safety is ensured with traditional safety measures and processes (certification and licensing).



Figure 9: Drone Classification¹⁰

- "open" is a category of UA operation that, considering the risks involved, does not require a prior authorization by the competent authority before its operation;
- "specific" is a category of UA operation that, considering the risks involved, requires an authorization by the competent authority before the operation takes place and takes into account the mitigation measures identified in an operational risk assessment, except for certain standard scenarios where a declaration by the operator is sufficient;

¹⁰ Elmrabti, Amin, Valentin Brossard, Yannick Moy, Denis Gautherot, and Frédéric Pothon. "Safe and Secure Autopilot Software for Drones." ERTS 2018.

- "certified" is a category of UA operation that, considering the risks involved, requires the certification of the UA, a licensed remote pilot and an operator approved by the competent authority to ensure an appropriate level of safety.

Only the "open" and "specific" operations are covered by the prototype regulation. The "open" category is further divided into four subcategories, based on technical requirements, operational limitations and requirements for the remote pilot or operator. The subcategories are:

- subcategory A0: operation of UA posing a negligible risk of severe injury to people on the ground or damage to manned aircraft, and requiring neither specific remote pilot competence nor age limitations;
- subcategory A1: operation of UA complying with requirements ensuring that they pose a negligible risk of severe injury to people on the ground or damage to manned aircraft, and requiring neither specific remote pilot competence nor strict operational limitations;
- subcategory A2: operation of UA complying with requirements ensuring that they pose a limited risk of severe injury to people on the ground or damage to manned aircraft, operated by registered operators, and equipped with geofencing and electronic identification;
- subcategory A3: operation of UA complying with requirements imposing technical mitigations like geofencing and electronic identification, posing a higher risk of severe injuries to people on the ground or damage to manned aircraft and operated by registered operators with higher competence.

For operations in the "open" category, risks are to be mitigated through a combination of safety measures, e.g. requirements and limitations on the operation, the UA, and the personnel and organizations involved as well as other limitations to be defined by the competent authority for geofencing purposes or for particular airspace areas. For operations in the "specific" category, risks are to be mitigated through safety measures identified in an operational risk assessment or contained in a standard scenario published by EASA.

Principles for UA Operations

- The operator of a UA shall be responsible for its safe operation.
- The operator shall comply with the requirements laid down in the applicable regulations, in particular those related to security, privacy, data protection, liability, insurance and environmental protection.
- The operator of a UA shall register with the competent authority and display registration marks on all the UA it operates in order for them to be easily identifiable, when required.
- The operator shall ensure that UA are equipped with an electronic identification means, when required.
- The operator shall ensure that UA are equipped with a geofencing function, when required.
- The competent authorities may designate zones or airspace areas where UA operations are prohibited or restricted.

3.4 SORA: Specific Operational Risk Assessment

JARUS (Joint Authorities for Rule-making on Unmanned Systems) has developed the Specific Operational Risk Assessment (SORA)¹¹, which is a methodology for risk assessment in UAS operations within the Specific category. Basically, SORA is a step-by-step procedure to evaluate risks that outputs a Specific Assurance and Integrity Level (SAIL) determining the necessary mitigation actions to achieve an acceptable level of risk.

SORA is a method based on the principle of a holistic/total system safety risk-based assessment model used to evaluate the risks involved in the operation of a UAS. Thus, it is based on a Holistic Risk Model that provides a generic framework to identify possible hazards and threats, as well as relevant harm and

¹¹ JARUS, "JARUS guidelines on Specific Operations Risk Assessment (SORA)," JARUS publications, 2018.

threat barriers applicable to a UAS operation. Given a specific operation, each risk can be defined as the combination of its frequency (probability) of occurrence and its associated level of severity. There are multiple risks to consider in a UAS operation, but they all can be classified into ground and air risks in terms of safety. Ground risks are basically those involving third parties in the ground, whereas air risks are those involving third parties in the air.

In the end, SORA determines how confident one is, in a qualitative manner, about the fact that the UAS operation will remain safely in the Operational Volume. This Operational Volume consists of the flight geography and the containment area. As the UAS is inside the flight geography, it is considered to be in normal operation and under operational procedures. However, if the UAS enters the containment area, it gets into an abnormal situation, being necessary the application of contingency procedures (e.g., returning home, manual control, landing on a predetermined site, etc.). Last, if the UAS gets out of the containment area (i.e., out of the Operational Volume), emergency procedures must be executed, as the operation would be out of control.

The SORA procedure begins with a description of the so-called Concept of Operations (ConOps), which specifies details of the operation assessed, such as the airspace requirements, the population density of the area, etc. It also describes the level of involvement of the crew and autonomous systems during each phase of the flight. After that, SORA proposes a step-by-step evaluation of the ground and air risks. Last, a SAIL is determined for the operation. With this evaluation in mind, there is a table called Operational Safety Objectives (OSO), which defines the objectives to be met by the operation depending on the estimated SAIL. In summary, SORA provides a logical process to establish an adequate level of confidence to conduct the UAS operation with acceptable level of risk. Essentially, the SORA method is based on a number of steps, which are depicted in Figure 10¹².

¹² C. Capitán, J. Capitán, Á. R. Castaño and A. Ollero, "Risk Assessment based on SORA Methodology for a UAS Media Production Application," 2019 International Conference on Unmanned Aircraft Systems (ICUAS), Atlanta, GA, USA, 2019, pp. 451-459, doi: 10.1109/ICUAS.2019.8798211.

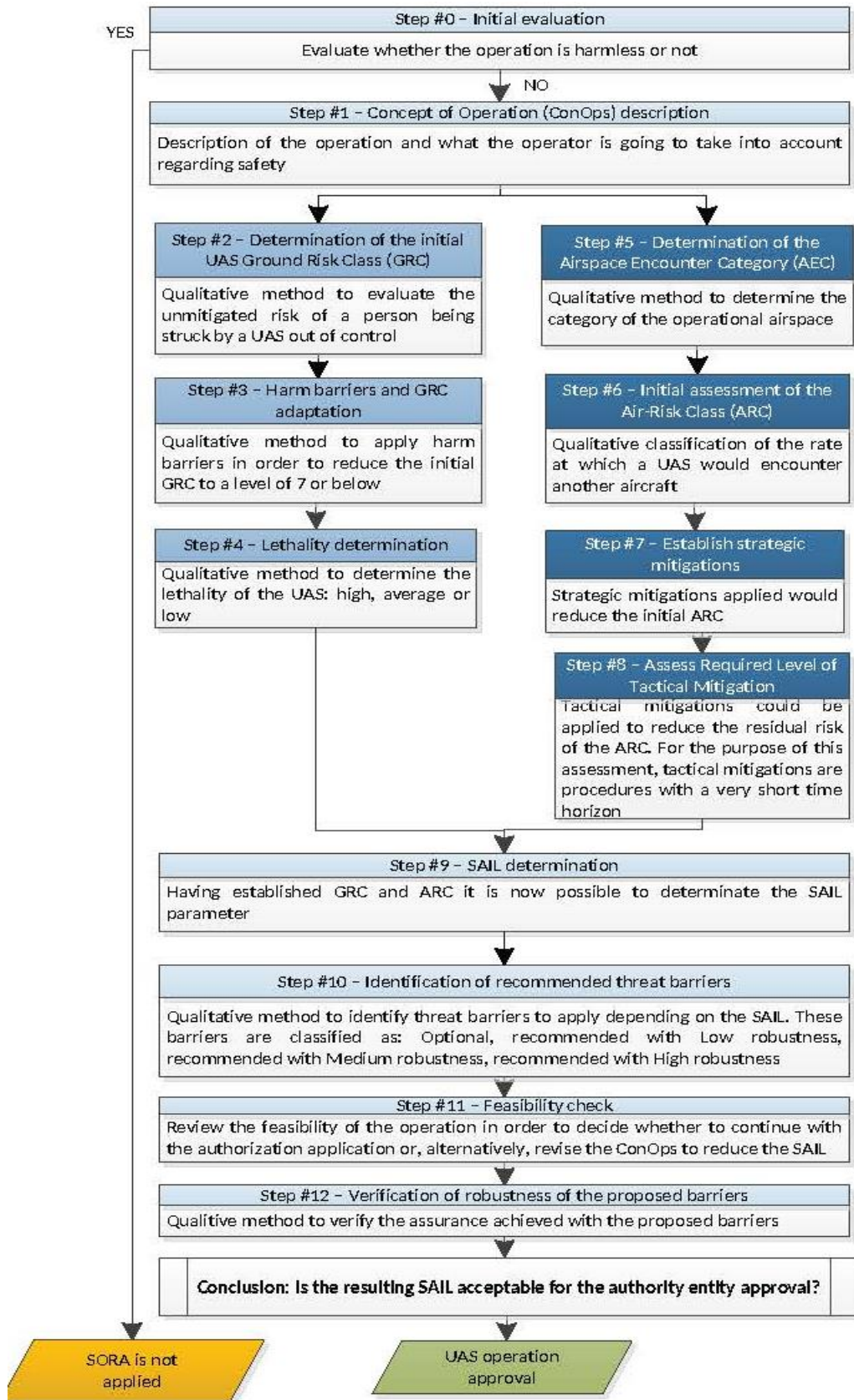


Figure 10: The SORA process

4 Needs for Drone Software Development

The drone system has different stakeholders. The stakeholders are classified into one of these categories: users, service providers, system integrators, and component providers (see the middle part of Figure 11). Based on each category, the requirements are defined from different viewpoint. First, in WP1, from the users' viewpoint, the requirements can be seen as a set of stories/scenarios they would like to achieve (e.g. traffic and incident monitoring). Second, in WP2, to achieve/support the user's stories, the service providers specify a number of system's features that should exist (e.g. geofence of incident area, flight plan, localization, video capture, etc.). Third, in WP3, these features are decomposed into a set of functional and non-functional requirements by the system integrators (e.g. the drone operator shall create a flight plan based on the incident, the system shall detect tactical conflicts due to proximity, etc.) Forth, in WP3, WP4, WP5, these requirements are then realized by the components' providers which are then integrated to form the sub-systems that constitute the system required by the end-users (see the top part of Figure 11). Finally, in WP6, the different tools that supports the different stakeholders in doing the different engineering tasks will be developed.

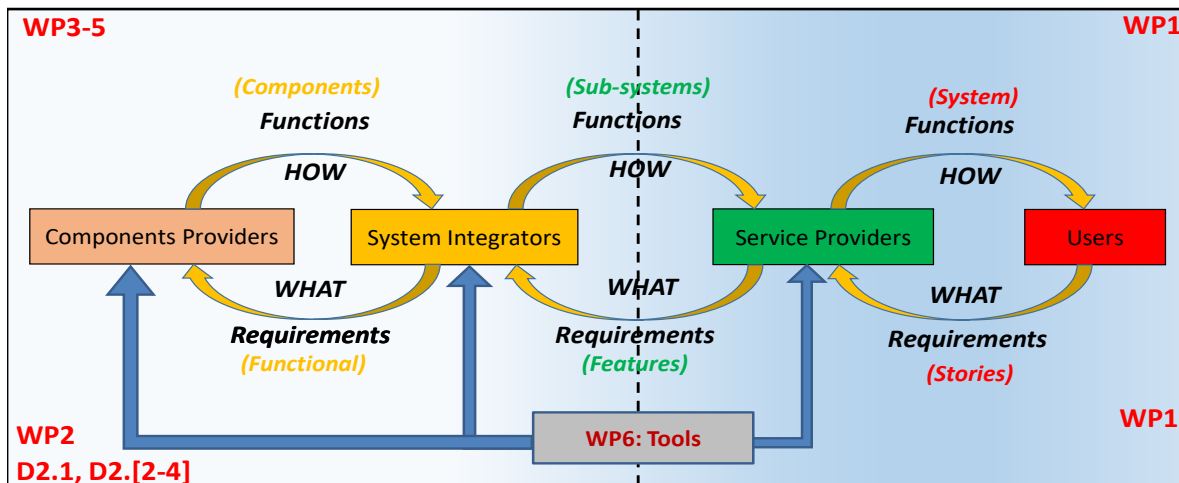


Figure 11: The different stakeholders and their viewpoints of the requirements

Following Figure 11, user needs/requirements have been collected in the deliverable D1.1, and Section 4.1 will describe end-user requirements that are raised from the analysis of **COMP4DRONES** demonstrators. Service providers and system integrator return of experiences (REX) are collected and presented in section 4.2.

Technology providers' solutions are developed in the relevant work packages (WP3-6). Thus, their respective specifications are documents in deliverables (D3.1, D4.1, D5.1, and D6.1), and Section 5.2.5 of this document will introduce high-level specifications related to the **COMP4DRONES** technologies from the integration perspective.

4.1 End-users Requirements

Based on the requirements of the project demonstrators, we have identified a unified list of drone usages. This list is divided into two categories flying stages and mission specific operations.

4.1.1 Flying Stages/Operations

A flight has several phases/stages. They are namely: taxiing, takeoff, climb, cruise, descent, and landing. If a drone completes all these stages of flight, then a flight cycle is completed¹³.

¹³ I. Astrov and A. Pedai, "Situational awareness based flight control of a drone," in IEEE International Systems Conference, Montreal, QC, 2011.

- Taxiing is the move of a drone on the ground using its own power autonomy or through a human.
- Take-off is the phase of flight in which a drone leaves the ground and becomes airborne.
- Climb is the operation to increase the altitude of a UAV.
- Cruise is a flight phase that happens between a climb to an altitude and before the drone starts to descend.
- A descent during a flight is any portion where a drone decreases its altitude.
- Landing is the last stage of a flight, where a drone returns to the ground.

4.1.2 Mission Specific Operations

Based on the project use cases, we have also identified the specific missions that can be performed by drones. These missions are classified into four categories: inspection, aerial surveillance, indoor missions, and logistics. In the following, we list these missions.

A) Inspection

- Checking the quality of air in and around structures and buildings.
- Assessing water quality where water is important to human survival, and then water quality need to be monitored and protected.
- Inspecting off-shore infrastructure by drone to decide whether a human is needed for doing a detailed follow-up inspection.
- Checking the health of plants to effectively manage plants, soil, fertilization, and irrigation in order to treat shortages, diseases, or pests in a timely manner.
- Gathering geological data for helping geophysicists in identifying and better approximating the location and presence of natural gas, minerals, and oil.

B) Aerial Surveillance

- Performing measures over places that are hazardous or difficult to be reached on foot.
- Surveying a land is a technique to determine the three-dimensional extension or terrestrial positions of objects of interest and compute the distances/angles between them.
- Gathering video and images for spotting an activity that can be missed by standard cameras with narrow view.
- Managing traffic operation based on (a) video streaming in real time from an accident location to the transport control center, (b) video analysis based on artificial vision, and embedded vehicle tracking capabilities on the drone.
- Digitalizing state of a constructive process by capturing data from areas under construction, highways, etc.
- Measuring crops height for providing timely and reliable spatial information to farmers and decision-makers employed in precision agriculture.
- Communicating with delivered seismic sensors to verify the good landing and coupling of the sensors.

C) Indoor Missions

- Analyzing underground constructions status by capturing a georeferenced scanner of the progress in the construction of a tunnel.
- Accessing GPS denied environment autonomously to (1) explore the unknown space and create a common reference model of the environment; (2) identify points of interest (e.g. human victims) and calculate routes for humans to extract those victims using this reference model.

D) Logistics

- Delivering seismic sensors to unlocks many hard-to-reach places where it is not worth going to shoot seismic in a traditional ground-based way.

- Delivering parcel through the coordination between a drone and a droid to fly between two buildings, and to reach places inside these buildings.
- Spraying crops to save a great amount of fertilizers and human effort by providing the right amount of fertilizer to every plant automatically.
- Watering plants by UAVs equipped with sensors that can identify parts of a crop that need more water. In this way, it is possible to improve the efficiency of water use, in the right places at the right time and in the right quantity.

4.2 Drone System Needs

Based on the project use-cases and service providers and system integrator return of experiences, a number of common needs for the drone system has been identified. The requirements/needs are divided into the following groups: drones' customization, communication, autonomous flying, safety, development effort, and execution platform.

4.2.1 Drone Customization

- The software deployed on the drones needs to be **extensible**, in order to make developers able to add modules for customization to facilitate adaptation to possible specification changes.
- Components shall provide a **configuration interface** to facilitate the configuration process (e.g. programmatic interface or setting file). In addition, **configuration management process** is needed to have full control over any changes made to the software and its parameterization.
- Better interactions between the system layers are needed to **facilitate integration** and resource sharing (mainly computational resources).
- Software modules need to maintain the **lowest coupling** possible, so that modules can be developed or modified independently.
- The communication between software modules need to be **loosely coupled**, so that modules do not need to be aware of other modules within the system.
- The ability to seamlessly **pick** and **replace** navigation algorithms like sensor filtering, path planning, and control systems, so that different algorithms can be tested quickly and efficiently.
- The possibility to have one integrated software application **capable to embed additional software modules and components**, in order to address both strategic (pre-flight), tactical (in-flight) and post-flight procedures, is perceived by UAS operators as compulsory need.

4.2.2 Autonomous Flying

- The drone must **autonomously navigate** with high position accuracy (a) along the tunnel, (b) when communication to ground station is interrupted based on geo-positioning data, (c) without using GNSS localization, and (d) during landing.
- The drone must be able to dynamically **detect** and **avoid** collaboratively or alone other aircrafts on the area and obstacles inside or outside a tunnel.
- Runtime monitoring techniques are needed to enable **safe decision making**.
- An **indoor positioning** system will provide geo-references position to drone.
- The trajectory of the drone is compared to the pre-recorded sensors trajectory done by the drone to **localize its own position** with more accuracy.
- A drone agent shall use a dedicated **external positioning system** giving it its position relative to a dronepad.
- An UAV agent shall use a computer vision to **land precisely**.
- The **SLAM** algorithms need to have the following features:
 - Reliable positioning based on multimodal sensing and geomagnetic measurements.
 - Probabilistic position estimates to model uncertainty.
 - Self-improving performance as mission carries on.

4.2.3 Drone Safety

- Drones need to be equipped with a safety module that can **prevent it from damaging itself or others** and to perform the operations that are requested correctly. The module using drone's sensors will evaluate the risk of the situation and if the safety condition is compromised will send actuation to drones system in order to maintain or restore a safety situation.
- The drone's safety Module shall keep the **operator possible to bring the drone from a danger area** in case of a system failure, example: propulsion failure from drone over a motorway: operator must have the possibility to bring the drone to a failsafe point.
- Drone safety module shall provide a consistent positioning system equipped with GPS failure detecting capabilities and SLAM algorithms to **enhance GPS precision**.
- The drone should have flotation device onboard to make sure that the drone can easily be **recovered after a crash** through robust and real-time system reconfiguration
- Definition of **fallback strategies** when components fail or are unable to work (e.g. lack of GPS signal preventing certain algorithms/components). Geo-information service (e.g. Ground obstacles map provided as a dedicated UTM service in strategic / tactical stage) can be considered as a mitigation strategy, in case of DDA sensors failure.
- The system must be able to **continue to operate even in abnormal conditions**.
- Perform as many tests as possible on the final system in order to **ensure that there are no errors**.

4.2.4 Drone Communication

- The drone must **communicate with the GCS or other drones** to inform about its landing position and to be able to monitor the mission operation.
- A **redundant/robust communication** that is capable of deploying secondary (alternative) communications channel, automatically reconfigured without the need for human intervention whatsoever.
- A **well-defined communication protocol** is needed for the data that needs to be communicated between software modules so that modules can be developed independently.
- The communication method shall **fit Europe radio transmission** requirements regarding frequency and signal power. The drone must also communicate using **4G and 5G network**.
- Drones have to provide an encryption module in order to exchange sensible data and to guarantee **data confidentiality**.
- A **secure communication** (i.e. detection of unauthorized access to the network, and detect both common and unknown cyber-security attacks in real-time and provide list of actions to perform to mitigate a detected attack).
- An Intrusion Detection System (IDS) will help the drones to **find out if someone is trying to steal or modify information** belonging to the system.
- IDS applied to GPS signal for **detecting spoofing attacks**.
- **Secure transmission** of important safety or mission data (i.e. RTCM for RTK navigation, C2link over IP, dynamic flight zones restrictions, etc.) through external UTM service providers.

4.2.5 Minimizing Drone System Development Effort

- Allow **components to be integrated in HIL (Hardware In the Loop) and SIL (Software In the Loop) platforms** to optimize project development times, avoiding planning days of "sterile" in-flight tests as well as minimizing risks that may result in poor programming in the drone flight
- **Reduce effort to obtain qualification (certification)** through automatic testing and verification of the components.
- Software modules need to **run seamlessly on a distributed computing platform with minimal configuration**, so that the end user does not have to spend time configuring each module when process distribution changes.

- The ability to **create independent software modules** in different programming languages so that development is faster and easier for the developers.
- The ability to add a new sensor or actuator to the system without having to change state estimation, world modeling, or control algorithms and code base so that the **development time in adding a new device is reduced**.
- The **common features of a software module need to be defined and abstracted**, so that developers do not spend time re-implementing already implemented functionality.
- Modelling tasks of the embedded software (using, for example, AADL) should allow a **better analysis regarding real-time verification and validation**.
- A domain specific language (DSL) might help **instantiating drone missions from graphical and, hence, more intuitive languages**.

4.2.6 Drone Platform and Execution

- In addition to its autopilot, a UAV may have a **companion computer to allow some components to run separately**. However, the two must have a strong link to share information and commands.
- The **software needs to be processor and computer architecture independent**, so that the system can be ported to a new computer without additional development.
- The **use of the minimum resources** and in terms of timing the execution of operations.
- Components shall define a **strategy for logging information** so that it is possible to filter the parts dedicated to manufacturers and those for the mission operators. The logging format must be suitable to allow to replay part of the mission.
- Components shall define a **strategy for sharing the data** on the network. The system shall be able to determine what to share from those strategies.
- Components of a UAV shall **identify themselves and detect undesired components**.

4.3 Regulations Constraints (SORA, EASA, U-Space)

The regulations' requirements on UAS design stem in different domains: general and multipurpose regulations for electronic devices, European regulations ranging from drones, drone operations, airspace considerations, manned airspace considerations, drone national regulations (before EU regulations become effective and regulations with regard to national security), and standards (CE marking, telecom, development process, safety assessment process, etc.).

As this is a highly multi-dimensional situation, there is a strong need for a usable concept of operations (CONOPS). The concept describes the drones, their operations, the technologies, environment, and the airspace assessment. Both regulations and CONOPS are moving targets, and the general regulations requirements will be tracked for the duration of the project. For each use case demonstration, national regulations will have to be taken into consideration on top of the following requirements.

The use of "shall" and "should", within the document, shall observe the following rules:

- The word "SHALL" in the text denotes a mandatory requirement imposed by EU regulation, or coming from standards applied in the **COMP4DRONES** project framework.
- The word "SHOULD" in the text denotes a recommendation expected to be followed unless good reasons are stated for not doing so.

RQ1: U-space requirements: UAS shall be designed to meet all requirements defined in U-space foundation services (U1), which are already mandatory in many member states: electronic registration, electronic identification, aeronautical information management, and geo-awareness. On top of those, we believe that some U-space initial services (U2) could affect both the UAS and the UTM design requirement as well: tracking, surveillance data exchange, geo-fencing, technologies allowing

incident/accident reporting, traffic information. Finally, additional requirements for initial services (U2) and enhanced services (U3) may be considered as advantages:

- **RQ1.1:** In order to support the remote identification, all UAS operated in the specific category shall be equipped with a remote identification system (Specs as per Annex Part 6 of commission delegated regulations (EU) 945/2019).
- **RQ1.2:** (CORUS CONOPS) UAV traffic display at ground control station shall have the capability to show minimum vertical and horizontal resolution, while in a multi UAV operation; the resolution will depend on the operation.
- **RQ1.3:** (U1 geo-awareness) Traffic display system shall have the capability to display restricted, prohibited, non-fly zones and permitted operational areas for each UAV.
- **RQ1.4:** (U1 geo-awareness) Each type of zones displayed in the system shall be represented differently.
- **RQ1.5:** (U2 dynamic geo-fencing) UAS shall be able to give priority to other emergency operation in case a command is initiated.
- **RQ1.6:** (Tracking and GCS) All UAVs during flight shall provide every operator that can control its trajectory with a clear and concise information on the geographical position of the UA, its speed and its height above the surface or take-off point.
- **RQ1.7:** (Geo-awareness/geo-caging) All UAVs shall provide means to prevent the UA from breaching the horizontal and vertical limits of a programmable operational volume.

RQ2. Common Altitude Reference System for manned and unmanned aviation: UAS should have GNSS capabilities and systems to convert between different altitude systems such as GNSS, barometric altitude, etc. For practical and cost reasons, small drones may use altitudes based on GNSS. It is assumed that U-space will generally use GNSS altitude and true north while accommodating other systems, but the discussion is still going on. In particular a parallel U-space study (ICARUS project¹⁴) has recently started by SESAR JU and EuroControl to address the Common Altitude Reference System¹⁵ issues for small drones and general aviation in Class G / X, Y, Z_u airspace volumes

RQ3: General telecommunication: No subsystems used in UAS shall emit unwanted interference to manned aviation systems, and abide by the ITU regulations (bandwidth, power, etc.):

- **RQ3.1:** (Telecommunication, SORA) C2 link, either terrestrial C2 link system or satellite C2 link system, shall be strictly complied with frequency allocated and technical requirements mentioned in ICAO Annex 10, Vol V and Annex 10 Vol VI.
- **RQ3.2:** (General Telecommunication) Mainly in B-VLOS operation, the C2 link chosen for UAS control shall have coverage to complete operational area of UAV operation.
- **RQ3.3:** (General telecommunication) Mainly in B-VLOS operation, the C2 link chosen for UAS control shall comply with the national regulations and safety assessment process in place.

RQ4: General requirements:

- (Commission implementing regulations (EU) 2020/649) In order to identify UAV in air separately from manned aircraft, green flashing light shall blink in night flight.

RQ5: Design requirements:

- **RQ5.1** (Design process, Article 11 of Commission implementing regulations (EU) 947/2019 mandate operational risk assessment for each types of operation). Based on risk assessment, design of UAV may vary and hence the capabilities of each UAV may differ. Chosen UAS design and architecture should be sufficient to demonstrate Operational Safety Objectives (OSOs) required for types of operation.

¹⁴<https://www.u-spaceicarus.eu>

¹⁵<https://www.eurocontrol.int/publication/uas-atm-common-altitude-reference-system-cars>

- **RQ5.2.** (Design process, SORA) UAS shall be designed to limit the effect of environmental conditions following SORA Annex E, Operational Safety Objectives (OSO) number 24.

RQ6: Additional requirements:

- **RQ6.1** (CE5 and CE6) All UAVs shall provide means for the operator in charge of flight safety, except in autonomous operations, to terminate the flight of the UA, which shall:
 - **RQ6.1.1** be reliable, predictable, and independent from the automatic flight control and guidance system
 - **RQ6.1.2** independent from the means to prevent the UA from breaching the horizontal and vertical limits as required.
- **RQ6.2** (CE5 and CE6) UAS shall provide the remote pilot with means to continuously monitor the quality of the command and control link.
- **RQ6.2** (CE5 and CE6) UAS shall provide the remote pilot with means to continuously receive an alert when it is likely that the link is going to be lost or degraded to the extent of compromising the safe conduct of the operation.
- **RQ6.2** (CE5 and CE6) UAS shall provide the remote pilot with means to continuously receive an alert, when the link is lost.
- **RQ6.3** (SORA) UAS flight control system shall incorporate automatic protection of flight envelope to ensure the UA remains within flight envelope in case of pilot error following SORA Annex E, OSO#18 and OSO#19.

5 State of the Art Systems Engineering Approaches in Avionics Domain

One of the most relevant domains to drones are the avionics domain. Therefore, in this section, we review the existing system engineering methodologies in this domain to use it as the bases for the **COMP4DRONES** methodology described in the next section.

5.1 System Engineering Process

The Systems Engineering Process is a comprehensive, iterative and recursive problem-solving process, applied sequentially top-down by integrated teams (see Figure 12). It transforms the needs and requirements into a set of system product and process descriptions, generates information for decision makers, and provides input for the next level of development. The process is applied sequentially, one level at a time, adding additional detail and definition with each level of development.

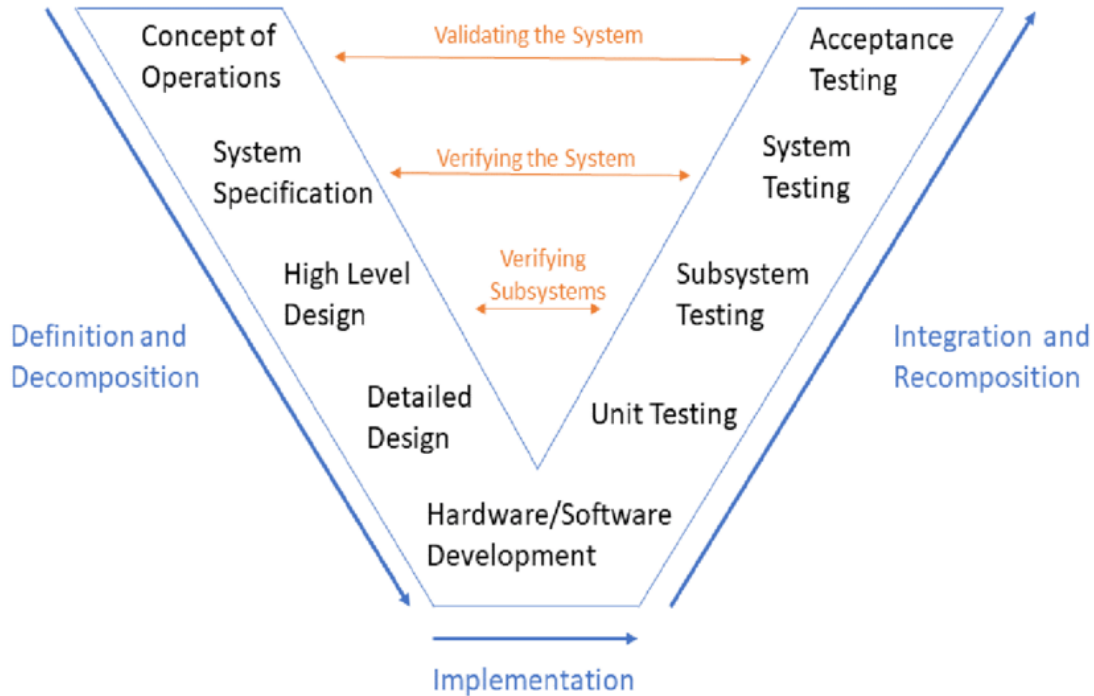


Figure 12: V-Model of a Conventional, Large-System Development Process

5.1.1 Standards for Avionics System Engineering Process

A drone system intended to be used by multiple customers or programs, and then it needs to be designed according to a process with all of the design assurance and artifacts that go with that process to enable lower cost of certification, lower risks and decreased time to market. This raises an important question: how can this be done, knowing that the system design process is a top down model (see Figure 13), while the hazard analysis and system safety assessment for a single system may be available, how to deal with multiple systems? The best practices that guide the development of safety-certifiable system is shown in Figure 13.

The ARP 4761 is the “how-to guide” for conducting the functional hazard analysis and system safety assessments, whereas the ARP 4754A document is the process to take the outcome of the analysis and flows the results down into requirements. The process flows down the first function design assurance levels (FDALs) and then item design assurance levels (IDALs) to the items in the system. These IDALs determine the DAL level required to be met by the item performing the function, and the design of that item falls under the design assurance guidance of RTCA DO-178C and RTCA DO-254. The first design assurance guideline from Radio Technical Commission for Aeronautics (RTCA) is DO-178C, Software Considerations in Airborne Systems and Equipment Certification. The U.S. Department of Transportation Federal Aviation Administration’s AC 20-115 made this guidance document an acceptable means, but not the only means, of showing compliance with the applicable airworthiness regulations for software aspects of airborne systems, which is the focus of this deliverable, and equipment certification. The second guideline is DO-254, Hardware Considerations in Airborne Systems and Equipment Certification, which is formally recognized by the FAA as an acceptable means of compliance for the design of electronic hardware in airborne systems.

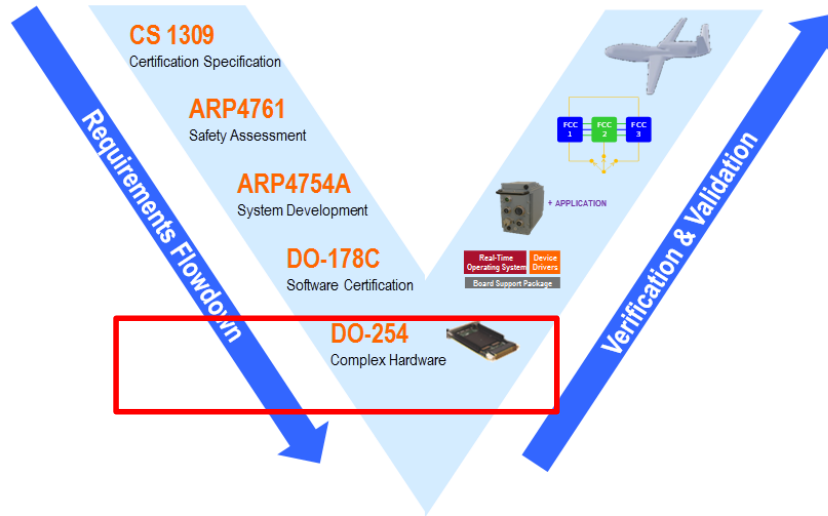


Figure 13: Standards for Avionics System Engineering Process

The **COMP4DRONES** project is focusing on the software aspect of the drone system. Therefore, the project will follow the DO-178C assurance guidance (see Figure 13), with less interest on the DO-254 which need to be followed by the semiconductor providers to provide certifiable hardware. In the following section, we describe a development life cycle that comply with the DO-178C.

5.1.2 DO-178C Software Development Lifecycle

DO-178C is the international and de facto standard for certifying all aviation safety-critical software. The need to comply with DO-178C can add significant cost to programs under development at a time when cost is becoming an increasingly critical factor in complex product development.

To develop aviation safety-critical software and to enable its certification, a process that can be followed is the development lifecycle described in the DO-178 standard which is show in Figure 14. The process recommends a waterfall model where each phase of the development process is finished completely before proceeding to the next phase. In addition, there are four stages of involvement for the certification liaison process (see Figure 14). In addition to these stages of involvement, there is a software development review after each phase of the development process.

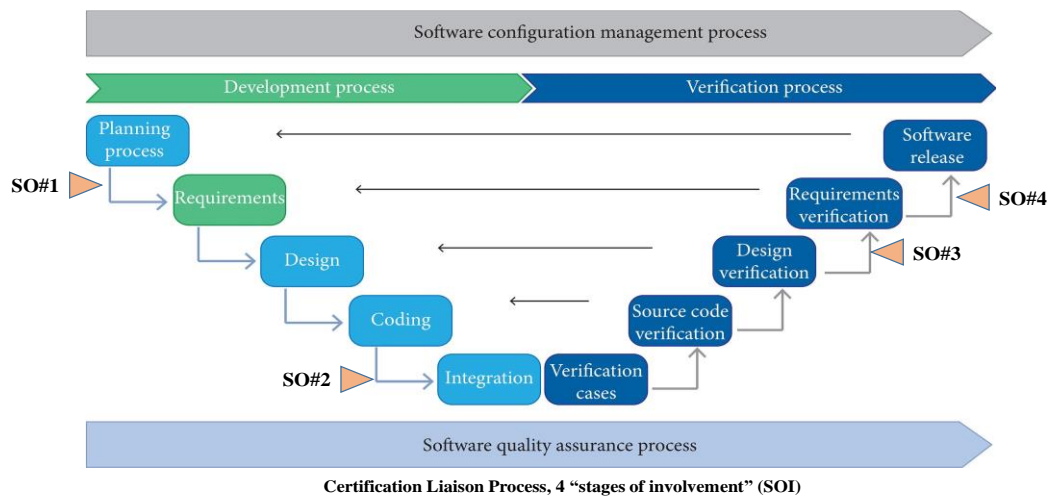


Figure 14: DO-178C software development lifecycle¹⁶

¹⁶RTCA / EUROCAE. —Software Considerations in Airborne Systems and Equipment Certificationll, DO178C/ED-12C, (2011).

The purpose of DO-178C is to provide guidance for developing airborne software systems to ensure that it performs its intended function with a level of confidence commiserate with its airworthiness requirement. DO-178C is objective-driven and companies may use a variety of means to achieve compliance as long as they meet the objective(s) in question. To comply with DO-178C, companies must provide multiple supporting documents and records surrounding their development processes.

First, planning requires associated output documentation, including the following: Plan for software aspects of certification (PSAC), Software development plan (SDP), Software verification plan (SVP), Software configuration management plan (SCMP), Software quality assurance plan (SQAP), System requirements, Software requirements standard (SRS), Software design standard (SDS), and Software code standard (SCS).

Second, output documents associated with meeting DO-178C standards in the development process include software requirements data, software design descriptions, source code and executable object code. According to DO-178C stipulations, without verifiable, unambiguous, consistent and well-defined requirements, a problem report must be created and submitted the issue back to the input source to be clarified and corrected. Those system requirements that will be realized by high level software requirements to one or more low-level software requirements must be able to be traced, and a low-level requirement to one or more high-level software requirements. Plus, all derived requirements need to be provided to the system safety assessment process. In a nutshell, this means that all of the source code developed needs to be traceable, verifiable and consistent, and it needs to correctly fulfill the low-level software requirements.

Third, to help ensure that the software fulfills DO-178C requirements, a verification report must be submitted that shows the absence of errors. All lower-level artifacts need to be proved that they satisfy higher-level artifacts by accomplishing traceability between requirements and test cases via requirements-based coverage and traceability between code structure and test cases through a structural coverage analysis. Each requirement in the software development process must be traceable not only to the code that implements it but also to the review, test or analysis through which it has been verified. It is also required to ensure that implemented functionality can be traced back to requirements and that testing can prove this (i.e. eliminate any dead code or code that is not traceable to requirements). Output documentation associated with DO-178C includes the following: Software verification cases and procedures (SVCP), Software verification results (SVR), Review of all requirements, design and code, Testing of executable object code, and Code coverage analysis.

Finally, to support compliance with DO-178C elements surrounding configuration management, companies are required to do the following: Uniquely identify each configuration item, Protect baselines of configuration items from change, Trace a configuration item to the configuration item from, which it was derived (lineage and history), Trace baselines to the baselines from which they were derived, Reproduce builds (replicate executable object code), Provide evidence of change approvals, Produce output documentation for a software configuration index (SCI) and a software life-cycle environment configuration index (SECI). DO-178C also requires that companies implement a problem reporting system to document any change to the formal design baseline.

5.1.3 Challenges in Applying DO-178C

The key problem of the above process is that errors that are detected late in the development process leads to of much re-work that is costly and leads to many delays in delivery of the software. In addition, projects that need to comply with DO-178C standards could see cost increases anywhere from 25 percent to 40 percent compared to projects that do not require compliance. The sources of additional costs may include the following:

- Reduced developer productivity due to increases in process complexity
- Manual reporting and documentation processes that are not suited to the level of detail required to comply with DO-178C
- Qualification activities involved in compliance

5.2 Approaches to Speed-up Avionics Software Development

A critical issue for airborne software development teams and companies is shortening the development cycle and reducing the development cost of adding features to software. In the following, we describe some approaches that has been proposed to shorten and speed-up the development lifecycle of avionics software.

5.2.1 Airbus Agile Development Process¹⁷

Within Airbus, development follows the “Requirements-Based Engineering” paradigm: every system or software functionality is justified with a requirement. Every requirement must be validated, to ensure the right system is being built, and verified, to ensure the system is being built right: verification is the largest single cost factor. Early validation and verification of requirements can avoid expensive late detection of errors, or at least detect them early, when they are least costly to resolve. In addition to the certification Stage Of Involvement SOI#s, the Airbus has quality gate reviews such as Project Planning Review (PPR), Software Requirements Review (SRR), Software Design Review (SDR), Test Readiness Review (TRR), Software Qualification Review (SQR) and Software Certification Review (SCR) as shown in Figure 15.

Airbus has proposed the use of agile software development to shorten the time for software development through incremental, functional, or agile process as shown in Figure 15. In these processes, the development is performed rapidly in an incremental way.

Incremental Approach: In this approach, multiple releases may be foreseen, resulting in the incremental instantiation, with two or more releases, each containing functional increments. Whether SOI#s are repeated or only performed once is dependent on the contract and certification liaison.

Functional Approach: The second approach shows a further step towards agile development: releases are divided into functional packages. As soon as the requirements for a functional package are complete and reviewed, the software design starts for this function. For the first functions, design starts prior to the formal quality gate SRR, resulting in a residual risk that the requirements may be impacted by interaction with a later function, but this risk is acceptable, as the requirements can be updated prior to SRR and the design aligned before SDR. By implementing high-risk functions first, this overlap can be beneficial as design can reveal requirement flaws, which can still be corrected prior to SRR. Also, this parallel development of several functions, each in a different development process phase, results in a more balanced work distribution across disciplines within the development team. “Verification lag” (the time between defining a requirement and completing verification of the implementation of the requirement), is reduced, but restricted by the quality gates. DO-178C Process/Guidance defines objectives to be met by software process activities, without stipulating HOW the development is performed. This allows the applicant to decide the order in which the evidences are accumulated.

¹⁷John Marsden, André Windisch, Rob Mayo, Jürgen Grossi, Julien Villermin, et al.. ED-12C/DO-178C vs. Agile Manifesto – A Solution to Agile Development of Certifiable Avionics Systems. ERTS 2018, Jan 2018, Toulouse, France.

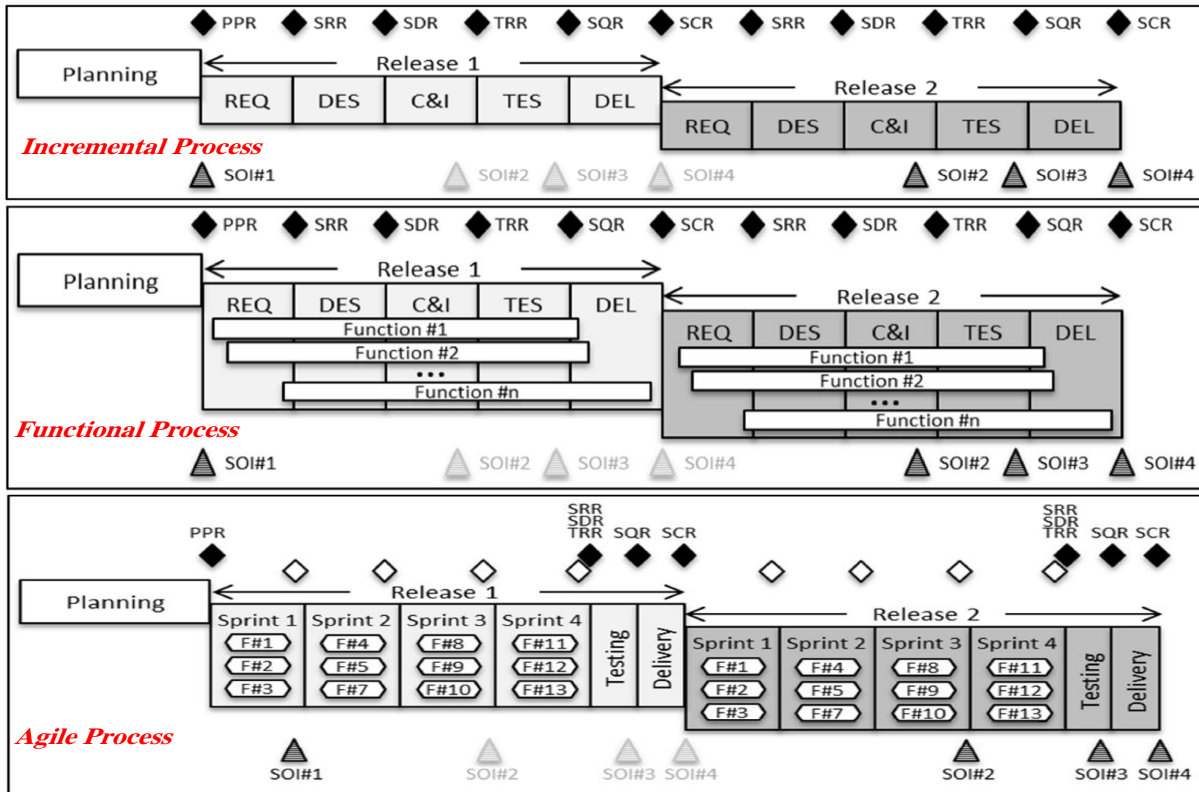


Figure 15: Airbus agile development process

Agile Approach: Further minimization of the verification lag can be achieved by reducing the granularity of the implemented functions or features further, instantiating the software life cycle once for each feature. However, this impacts SOI#s and quality gates: The quality gates SRR, SDR and TRR all occur simultaneously, as requirements, design and test procedures are only complete in the final sprint. This can be mitigated by “micro-reviews”, internal reviews held for every process transition of every function, documented with a checklist as quality assurance record. The quality gates then become a summary review of the accumulated evidences from the micro reviews. Further mitigation includes sprint reviews, at which the newly developed functions can be demonstrated to stakeholders, ideally including the customer: demonstration of the functions is a more tangible method of ensuring quality than reviewing documents! Number of implemented functions has proven a more reliable KPI than the number of requirements reviewed or the proportion of design completed. This approach can be described in a “Way of Working” section of the software plans, explaining the multiple instantiations of the software development processes and the micro-reviews. The duration of the planning process and timing of SOI#1 may be unchanged, but increased confidence in the plans may be obtained by performing initial sprints to validate and streamline the approach before submission of the plans to the certification authorities.

5.2.2 A Model-Based Agile Process for DO-178C Certification¹⁸

A new software development process that combines key advantages of both agile development processes and model-based engineering methodologies has been proposed to produce a Model-Based Agile (MBA) process capable of satisfying Federal Aviation Administration (FAA)-mandated process objectives for software of all Design Assurance Levels. A key element of the MBA process is the use of an Agile-style iterative and incremental approach to requirements elicitation, capture, and verification.

¹⁸ David J. Coe, Jeffrey H. Kulick (2013), A Model-Based Agile Process for DO-178C Certification, Proceedings of the International Conference on Software Engineering Research and Practice (SERP)

Provided that an appropriate modeling tool is used that admits executable specification models, one can start by developing use cases and iteratively refine them into executable models. These executable models will bring to bear the advantages of agile requirements elicitation while facilitating the complete capture of a set of requirements for a system before the detailed design and testing begin.

A team works closely with the customer to identify requirements and to develop acceptance tests that will be used to verify the correctness of the requirements as captured in the specification model. The acceptance tests are executed on the specification model, and the results used to verify correctness, completeness, and consistency of the specifications. It should be understood that the test cases used to exercise and test the requirements model would most likely not be directly applicable to any subsequent design model without substantial refinement due to their lack of detail. For example, messages might only contain message types for exercising the specification model while design model messages will require detailed values in the message body for exercising any design model.

For the MBA process, they have proposed the use of a Unified Modeling Language (UML) tool such as IBM's Rhapsody to capture these requirements as they emerge from face-to-face interactions with the customer. Requirements capture via UML has been shown to be an effective means of communicating requirements information among stakeholders. Moreover, the Rhapsody tool allows the construction of executable UML models using non-synthesizable components of UML such as sequence diagrams. Commercial UML tools such as Rhapsody also provide interfaces to industry standard textual requirements management tools such as IBM's DOORS. This allows the developers to maintain traceability from the top-level textual requirements to the specification model as mandated by the FAA.

In their MBA process, the initial iterations are focused only on eliciting, capturing, and refining requirements for input into the safety analysis process. Once the customer and the team are satisfied that all requirements have been identified and verified, the safety analysis is performed to determine the DAL and identify any safety requirements. The safety requirements are integrated into the UML specification model and verified during the next iteration. A test coverage analysis on the model can be performed to ensure adequate testing of the specification model to the required DAL. Since only design models can be used for code generation, future iterations will focus primarily on implementation of the captured requirements.

As with the previous modifications to the requirements elicitation process to enable agile development of safety critical systems, the design process is similarly modified to accommodate construction of the optional design model, if desired. The forced separation of specification models and design models permits the use a different modeling tool for the design model including one that is more amenable to code synthesis and formal verification, if desired. Once the textual requirements and specification model are completed, then the implementation of the requirements can proceed piecemeal with testing and verification of each unit, component, subsystem, and system in turn.

5.2.3 Towards a DO-178-Aligned Agile Approach¹⁹

Scrum²⁰ phases being applied to the software development and software verification processes of DO-178, as depicted in Figure 16, allows the mapping of agile methods to these processes.

During the Preparation phase, planning and architecture activities are performed. Scrum's concept of planning is somewhat broader than that of DO-178. Scrum includes the definition of the next software release based on the currently known backlog, analysis of system requirements, and development of user stories. The architecture activities establish (or update) the software structure. During the Development phase, the functionality of a new release is developed as well as tests for new or changed code. The software is designed, and source code is implemented, integrated, and tested during a

¹⁹ Hanssen G.K., Wedzinga G., Stuij M. (2017) An Assessment of Avionics Software Development Practice: Justifications for an Agile Development Process. In: Baumeister H., Lichter H., Riebisch M. (eds) Agile Processes in Software Engineering and Extreme Programming. XP 2017.

²⁰ [https://en.wikipedia.org/wiki/Scrum_\(software_development\)](https://en.wikipedia.org/wiki/Scrum_(software_development))

sequence of sprints. In the Closure phase, the software release is prepared, including system testing, final documentation, and release. The sequence of Preparation, Development, and Closure is repeated until the final software release has been completed. The activities in each phase are described in more detail below.

During the Preparation phase, the allocated system requirements, or a subset thereof, are taken and high-level requirements (HLRs) are produced in the form of features that are further divided into user stories. A software architecture is established (or refined), which, together with the prioritized HLRs, as part of the product backlog, is provided to the Development phase. As required by DO-178, outputs of all processes are verified, e.g., by means of review or analysis. The planning process of DO-178 is kept outside the agile process. It is responsible for establishing and updating all plans, including the Software Development Plan, the Configuration Management Plan, and the Plan for Software Aspects of Certification. The latter document is used for communication with the authorities.

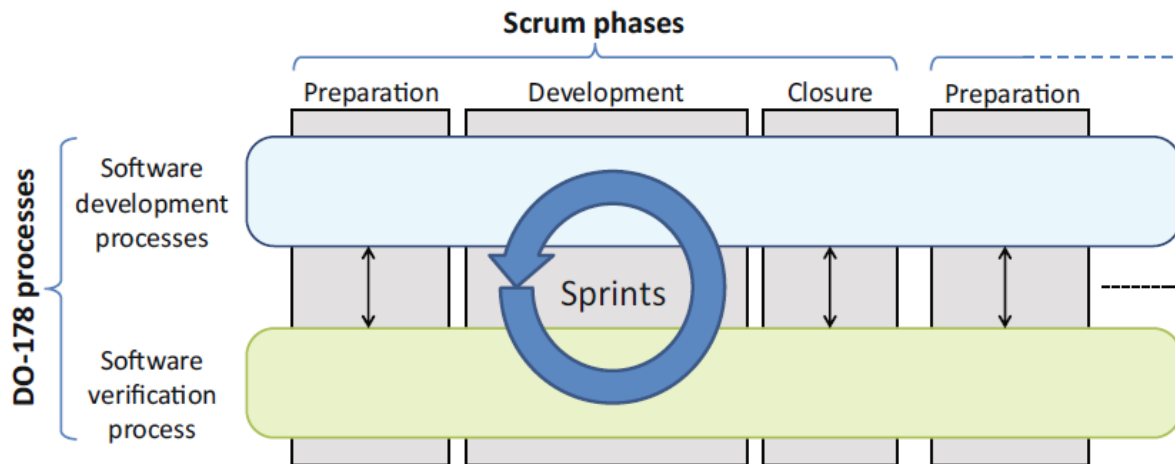


Figure 16: Application of Scrum phases to DO-178 processes

The Development phase consists of a sequence of sprints, all with preferably the same fixed duration (from 1 to 4 weeks). The number of sprints is not fixed. The result of a sprint is a set of implemented and tested user stories that are integrated into a working application. In addition, a sprint produces information for the assessor (the data items). The application can be demonstrated to stakeholders, but not all features may be complete and hence it is not releasable. Agile development promotes the Test-Driven Development (TDD) technique. A cyclic process is performed whereby first low-level requirements (LLRs) are established together with their test cases. Next, test code is produced and all tests are executed to verify that they fail. Then, source code is produced that just passes the tests. Finally, the code is re-factored and tests are re-executed. This cycle repeats until all LLRs have been implemented. In practice, the TDD technique implies that software development activities will be performed in conjunction with software verification activities.

Upon start of the Closure phase, a sufficient number of features should be completed to warrant release of the application. During Closure, all data items that already exist in some form are brought up to date. The remaining data items required for compliance with DO-178 are produced by other processes than software development and software verification. For example, the software configuration process produces the Software Configuration Index and the certification liaison process produces the Software Accomplishment Summary.

5.2.4 Agile for Aerospace²¹

A possible agile DO-178 development process is shown in Figure 17. In this process, the development plans and standards are developed, reviewed, and approved in the same manner as the traditional approach. After the FAA approves the process documents (as described in the Plan for Software Aspects of Certification (PSAC)), the development process begins and the differences from the traditional method become apparent.

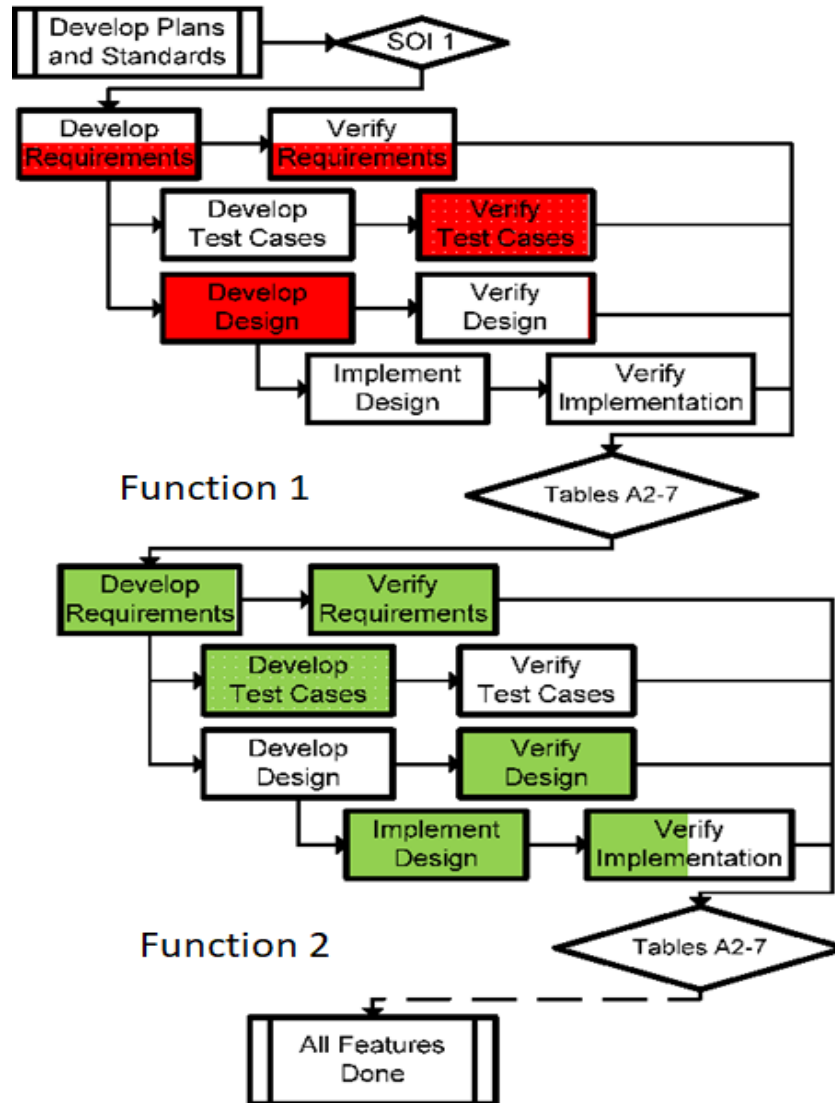


Figure 17: Agile DO-178

In an agile DO-178B software development process, all artifacts for a specific feature are created or updated during an iteration. After a predetermined number of iterations, the FAA (via the Designated Engineering Representative (DER)) performs an intermediate combined SOI audit covering all the objective tables (but only for the features developed since the last audit). After each of these audits, the software becomes ready for the final certification. After the project team has developed all necessary feature areas (one iteration at a time), the DER performs the final certification review on the artifacts.

Although this results in more SOI audits, each of the intermediate iterative audits would be much less time-consuming than a traditional SOI due to the smaller scope. The added benefit of more frequent

²¹ S. H. VanderLeest and A. Buter, "Escape the waterfall: Agile for aerospace," 2009 IEEE/AIAA 28th Digital Avionics Systems Conference, Orlando, FL, 2009, pp. 6.D.3-1-6.D.3-16, doi: 10.1109/DASC.2009.5347438

audits is that any issue identified during initial audits could then be mitigated on features that were yet to be implemented, therefore reducing risk and costly rework as the program progresses. This approach also brings the FAA into the program more directly through the full process, which reduces the risks of unforeseen issues arising during the final certification review. In a way, running the SOI audit on a feature is a dry run of the process – just as a dry run of tests can provide confidence that the formal test run will go smoothly, early intermediate SOIs can provide confidence that the process approach is acceptable to the FAA. Of course, this may potentially increase the workload for the DER.

5.2.5 Gains from Agile Development Methodology

Since a traditional software process suffers from its complexity, the effort needed to add functionality increases as the project progresses. This defines the Boehm curve that is already more than 30 years old²². Agile processes aim for an ideal, flattened curve, allowing a constant development pace (see Figure 18). At the beginning of a project, certification driven software development follows these curves. We call this the software phase of the project. A first divergence can be seen in the figure when deployment tests begin: the software is prepared to get tested in the field. Here, the process slows down because of hardware dependencies and (partly automated) acceptance testing. These issues are common in embedded software development. Hence, we call this the embedded phase. An even more significant slowdown is encountered when the software is ready to be certified. In this stadium, that we call certification phase, the software is presumed bug-free, but much documentation and manual testing is needed to provide the artifacts that are necessary for certification.

In the following, the most important agile opportunities for every phase are discussed. Together with the risks and weaknesses, they define a new curve for an agile DO-178B driven process.

Software Phase: Software development in this phase is not yet affected by other issues or constraints - the software is developed independently. In this phase, all agile practices may be used. Requirements changes - even hardware changes - are welcomed. User and acceptance testing can be fully automated within the context of the software.

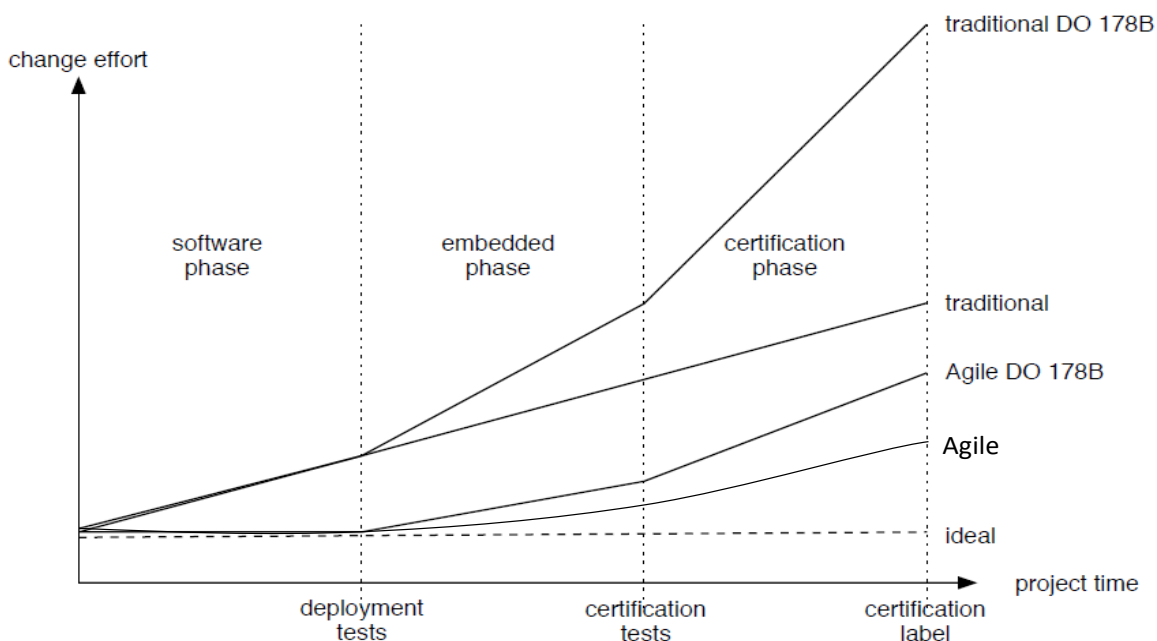


Figure 18: Software processes compared²³

²² Boehm, B.W.: Software Engineering Economics. Prentice-Hall Advances in Computing Science & Technology Series (1981)

²³ Wils A., Van Baelen S., Holvoet T., De Vlaminc K. (2006) Agility in the Avionics Software World. In: Abrahamsson P., Marchesi M., Succi G. (eds) Extreme Programming and Agile Processes in Software Engineering. XP 2006.

Embedded Phase: In this phase, the software must be deployed and tested on the target hardware. There are no fundamental reasons to abandon agility. However, there may be some repercussions on activities that depend on the developed software as input. Example activities include continuously installing the software on the target hardware, retesting the hardware, environmental tests and generating documentation. The opportunities in this phase mainly consist of automating these tasks. Also, feedback and communication become more important in this phase, in order to cope with the dependencies and coordination of software and other activities. Agile practices to be considered for facilitating these tasks are daily standup meetings and post-iteration workshops.

Certification Phase: This phase brings with it many additional tasks that need to be executed upon each software change. Code coverage and non-functional requirements (such as maintenance) need to be analyzed. Traceability needs to be established and manual testing and reviewing is required. The evidence of all these activities needs to be collected and reported. A logical measure here is to limit the amount of changes. First, to keep the requirements changes to a minimum, the customer can write their own acceptance tests. To further minimize the effort related to changes, one can utilize model-based traceability capabilities to help identify from text to code documents (including intermediate models) the specific impact of one change on the already accomplished activities and produced artefacts. There is also an opportunity to handle and manage documents more as source code, so that agile code-centric practices can also be applied to them. In particular, one can apply the following practices to reduce the time spent on creating, managing and reviewing documents:

- Auto-generate not only code, but as much documents as possible;
- Include all documents in a version control system;
- Manage their dependencies, so that it is immediately clear what document parts are affected by an artifact change.

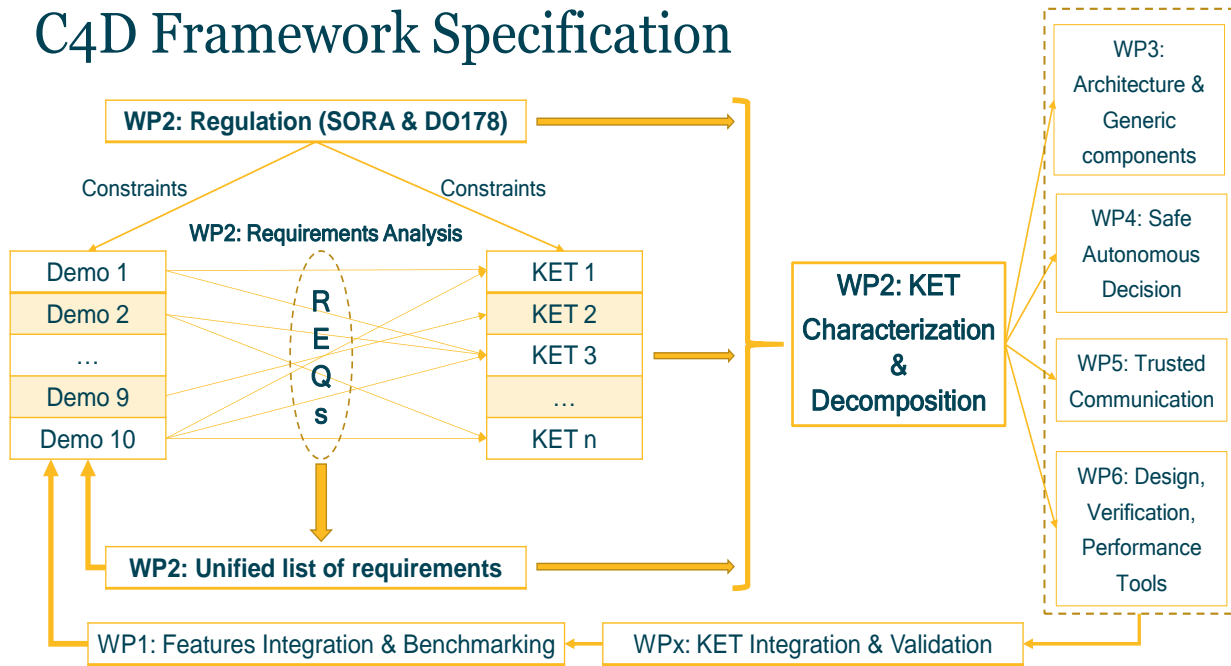
6 COMP4DRONES Methodology and Workflow

The potential applications for drones, especially those in manned areas or into non-segregated airspace, are currently not possible without the **development and validation of certain key enabling technologies**. The development and integration of these key enabling technologies require the drone to be equipped with sophisticated sensors to have precise knowledge of the environment (perception), trusted communication capabilities (identification, availability and cyber-security) and the ability to make intelligent decisions autonomously in real time to react to unforeseen situations (detect & avoid, safe coordination, contingency).

The **COMP4DRONES** framework specifications (see deliverable D2.1 for more details), as shown in Figure 19, define what the framework should provide (i.e. components, tools, methodology and workflow) to be key enabling technologies (KETs) for drone. To limit the scope and better target the domains considered in the project, the use cases specifications are used as inputs. The final specifications should also consider the extensibility and usage of the framework in related domains of application.

Figure 19 shows the overall workflow during the **COMP4DRONES** project. First, the different demonstrators are specified (i.e. scenarios, features, and functional and non-functional requirements). The demos' requirements are then analyzed to get a unified list of requirements (WP1). Second, the unified list of requirements is used to identify the key enabling technologies that are going to be developed during the project (WP2). Third, the identified key technologies are characterized and decomposed into the technical working packages: the architecture and its generic components (WP3), technologies for safe autonomous decision (WP4), trusted communication technologies (WP5), and tools for design, verification, performance analysis, etc. (WP6). Fourth, the developed technologies are integrated and validated. Finally, the key technologies are evaluated by using them for the development of different demos (WP1).

C4D Framework Specification



*KET: Key Enabling Technologies

Figure 19: The overall work flow of the COMP4DRONES project

In this section, based on the state of the art methodologies described in Section 5, we describe the initial **COMP4DRONES** methodology (which going to be detailed in the deliverable D2.3 and D2.4) in sub-section 6.1. We also describe the challenges and intended solutions to provide drone architecture (sub-section 6.2) and tools (sub-section 6.3) for supporting the proposed methodology. Finally, challenges and expected contributions to provide key technologies in the domains of safe-autonomous decision (sub-section 6.4) and trusted communication (sub-section 6.5) are presented. These technologies are going to be developed following the proposed methodology with its supporting architecture and tools.

6.1 C4D Reuse-based Agile Development Process Workflow

In the current economic environment, it is important for companies to minimize the additional costs related to DO-178C development. In

Table 1, we describe reasons for the additional cost and a number of suggestions to reduce such costs. This leads us to propose a reuse-based agile process for cost minimization. In this section, we start by describing the conflicts between the agile methods and DO-178C and how to resolve them. Then, we describe our proposed process.

Table 1: DO-178C Problems and Suggested solutions

Problems	Solutions
Reduced developer productivity	Introduce more automation through increased usage of qualified tools as a support (e.g. code generator, certified compiler, automated testing tool, etc.)
Establish overall traceability during product development	Model-based Approach (SysML, Simulink, AADL, etc.) and tool interoperability
Manual reporting and documentation	Model-based document generation
Qualification activities involved in compliance	Incremental certification through increasing of reusability of qualified components (sub-systems)
Running manual SOI to communicate to authorities	Workflow engines to enable standard compliant engineering and generation of certification reports

6.1.1 Conflicts Between Agile and DO-178C and their Resolution²⁴

The potential conflicts between agile and DO-178C that must be addressed in the planning process are as follow:

- The less-valued items on the right-hand side of the Agile Manifesto²⁵ (processes & tools, documentation, contract and plan) are crucial to software certification, so clearly deploying agile in the avionic software domain must clarify the apparently conflicting priorities with the higher-value left-hand side items (individuals /interactions, working software, customer collaboration and responding to change).
- Embracing change (planning to re-plan) not only endangers the project schedule and budget, but may invalidate SOI#1, resulting in additional certification liaison and probably the need to repeat some work, especially expensive verification activities. Furthermore, allowing change appears to absolve the customer of the responsibility to be clear about his requirements at the start of development: a recipe for disaster!
- Continuous improvement implies that teams will change HOW they develop software during the development, contradicting the certification mantra of executing the plans.
- Agile frameworks favor “light” requirements in the form of user stories, which may not be complete, resulting in incomplete testing and verification gaps: the objectives of DO-178C cannot be fulfilled!
- The agile concept of evolving architectures, deliberately not considering anticipated future changes to prevent duplication of effort through rework can result in an under-dimensioned architecture: this is less likely if all design is performed before coding is started.
- Many excellent but specialized engineers can struggle with the flexibility required in agile development: whilst they are extremely productive in their area of specialty; in a cross functional/cross-discipline team, they are expected to perform other tasks in which they are less effective, slowing development.
- Where does agile stop? There are potential conflicts wherever in the enterprise agile teams meet non-agile: if the agile team has detected the need for a change, but the non-agile team is not willing to adapt, how is this resolved?

Key aspects of the engineering process to avoid the above conflicts are:

- Promotion of an agile mind-set, planning to re-plan and commitment to continuous improvement: training and coaching must be provided. One may put effort to improve continuously the practices (people, process, technology), e.g. by adopting best practices from CMMI (Capability

²⁴ Hanssen G.K., Wedzinga G., Stuij M. (2017) An Assessment of Avionics Software Development Practice: Justifications for an Agile Development Process. In: Baumeister H., Lichter H., Riebisch M. (eds) Agile Processes in Software Engineering and Extreme Programming. XP 2017.

²⁵<https://agilemanifesto.org/>

Maturity Model Integration)²⁶ or SPICE (Software Process Improvement and Capability determination) referential²⁷.

- Highly-Automated Continuous Integration and Verification (CIV) to minimize repetition of manual verification activities and to immediately detect negative impacts of increments on the baseline.
- Parallel continuous improvement of the tool chain: both manual and automated verification approaches are described in the Tool Qualification Plan & PSAC. Whether to perform the tool qualification can be decided during development when the re-verification effort becomes clear: if it is not performed, SOI#1 is not endangered, as the manual verification approach has also been agreed.
- User-oriented requirements (user stories), including a description of how the customer will use the function, from which test cases / acceptance criteria can be derived. This leverages the practice of Test-Driven Development, and could be termed verification-driven development or Acceptance-Driven development: requirements with a high-level of abstraction are constrained by acceptance criteria ensuring verifiability whilst remaining design-agnostic.
- Optimized configuration and change management, supporting non-bureaucratic exhaustive tracing/linking between requirements, design and software, at a fine level-of-granularity, enabling automated change impact analysis of requirement increments, continuous baselining and generation of configuration records
- Small, cross-functional, cross-discipline, co-located teams.
- Quality Assurance (QA) records generated in micro-reviews for every process transition of every function.
- Sprint reviews should involve the customer, or at least the in-house customer representative: changes to the customer requirements are permitted, on the understanding that it is a trade-off: the backlog is reprioritized and lower priority features may have to be removed from the release.
- At the Release Planning event all development teams perform rough-order-of- magnitude effort and conservative performance estimates, enabling dimensioning of the architecture.
- The answer to the question “where does agile stop?” is: when the organization no longer sees additional benefits, which, in view of accelerating technology development and increasingly competitive markets, is probably never: either the entire organization becomes agile, or the organization will probably not survive.

6.1.2 C4D Development Process

Based on the Airbus agile process, in the **COMP4DRONES** project, a reuse-based agile development process is going to be followed as shown in Figure 20. In this process, after the planning phase and requirements identification, a repository that contains hardware and software components is checked to identify COTS (Commercial off-the-shell) components that exist and can be used to satisfy the requirements by the use of assumption-guarantee contracts approach. In case of such a COTS component exists, the development process starts from the integration phase. Otherwise, the full development cycle needs to be followed from design to delivery (see Figure 20). The main idea of this process is to speed up the development process through reusing the existing components that supports the identified requirements. This reuse-based strategy is also adequate to smooth the certification process if used with the concept of dependability certificate²⁸—that contains all information on the dependability attached to the reusable components.

Following this reuse-based agile process, the workflow for developing a drone system can be divided into two main phases: development and integration as shown in Figure 21. In this workflow, the drone system is decomposed into sub-systems which are later divided into components. These components are either reused or fully developed from scratch. After having all required components developed or

²⁶ www.sei.cmu.edu/cmmi

²⁷ I. O. f. Standardization, ISO/IEC International standard 15504 (all parts), Information technology – Process assessment, 2004.

²⁸ D. Schneider, M. Trapp, Y. Papadopoulos, E. Armengaud, M. Zeller and K. Höfig, WAP: Digital dependability identities, IEEE 26th International Symposium on Software Reliability Engineering (ISSRE), pp. pp. 324-329, 2015.

made ready for integration, the integration phase starts where the components are integrated together to form a sub-system. These sub-systems are then integrated to have a fully functioning system (see sub-section 6.1.3).

The reuse-based agile process will be constructed following a model-driven engineering approach. Indeed, interest in using model-based system engineering/design (MBSE/D) has been steadily increasing in the system engineering community. The INCOSE defines MBSE as “the formalized application of modeling to support system requirements, design, analysis, verification, and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phase²⁹. MBSE relies upon system level models and offers convenient frameworks to integrate different dedicated analysis views within a global modelling environment. Hence, to be successfully implemented, an MBSE approach necessitates the following tree elements: 1) a modeling language, 2) a modelling methodology and 3) a modelling framework that implements the modeling languages, preferably customized to support the development methodology.

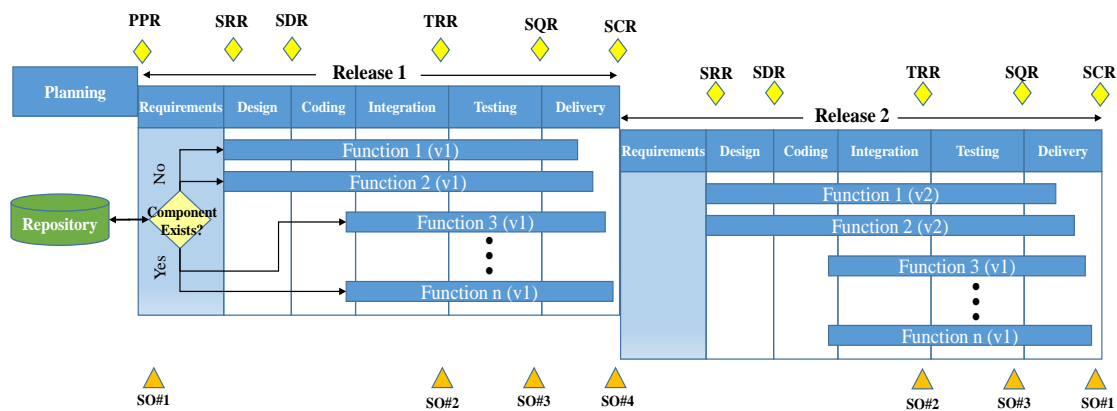


Figure 20: Reuse-based agile development process

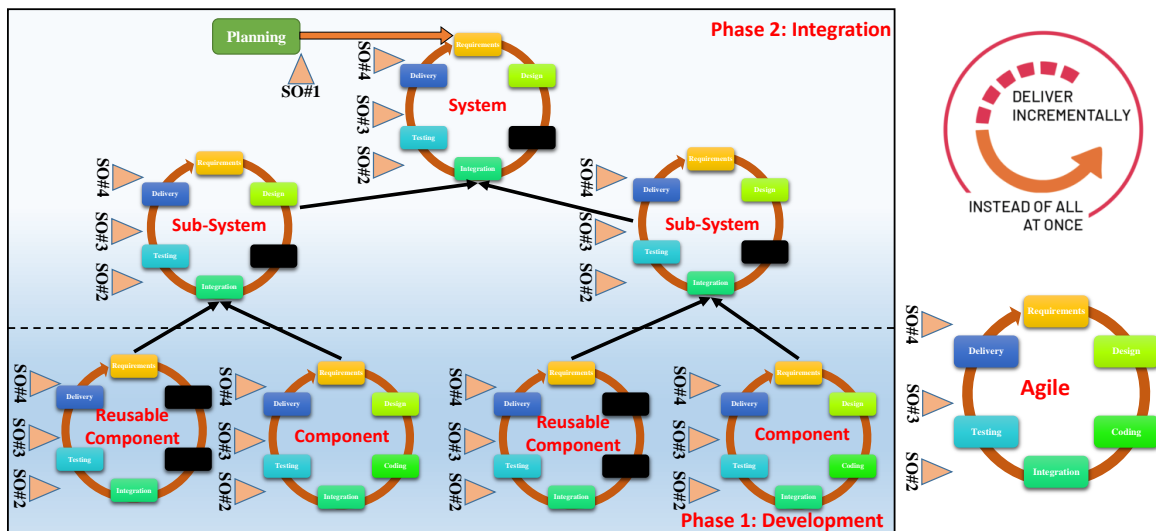


Figure 21: Reuse-based agile development process workflow

The modeling language defines the notation (visual representation) and the semantics (meaning) used to construct the model. Recent studies place the Unified Modeling Language (UML) as the most used,

²⁹ Incose, Incose Systems Engineering Handbook V4, John Wiley and Sons, 2015.

tool supported and disseminated modeling language³⁰. UML provides a standard extension mechanism called profile, which allows enriching the language with domain specific concepts. This is the purpose of SysML. SysML is such an UML profile that specialized UML concepts for system engineering. SysML is designed to provide simple but powerful constructs for modeling a wide range of system engineering problems. It specifically addresses the areas of requirements, structure, behavior and allocations/constraints to support various engineering analyses. SysML can be itself further specialized using profiles to cope with a specific methodology depending of the domain or application.

6.1.3 A System Composition Approach

During last years the European Commission has funded many basic and industrial research projects to leverage to Europe a framework of key enabling technologies in the Robotics domain. One of these projects is RobMoSys³¹. This project has developed a System Engineering Approach to enable the design of safe and efficient robots based on reuse and composability of qualified components. In the context of the RobMoSys project, two complementary development environments (i.e. Papyrus4Robotics³² and SmartMDSD³³) have been produced. These environments follow the proposed engineering approach and the standard frameworks in the robotics domain such as RTC³⁴ and SmartSoft³⁵ (see the details in the deliverable D3.1). In the following, we describe the RobMoSys approach, and next deliverables will introduce the **COMP4DRONES** composition approach adopting and extending this approach.

In RobMoSys, system composition requires a structure. This structure has requirements originating from three perspectives (see Figure 22): composability as the ability of building blocks to be combined and recombined into different compositions. Since composability is a cross-cutting concern, it needs consideration through the whole composition workflow that involves all steps, stakeholders and elements. Finally, the workflow must be applied by stakeholders who need proper support via tooling.

Composability is the ability behind system composition that enables to put together parts in a meaningful way. It comes with composability as the property of parts that makes them become “building blocks”. Composability puts a focus on the new whole (system) that is created from existing parts. It is not just about making the individual parts work together just by uniting pieces that then become inseparable. Composability is the capability to select and assemble simulation components in various combinations into valid simulation systems to satisfy specific user requirements³⁶.

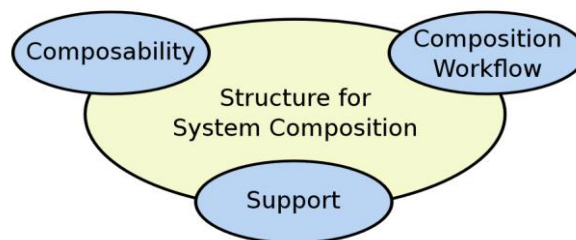


Figure 22: A structure for system composition has requirements originating from composability, composition workflow, and support via tooling³⁷

³⁰ I. Malavolta, P. Lago, H. Muccini, P. Pelliccione and A. Tang, What industry needs from architectural languages: A survey, TSE Journal, p. pp. 869–891, 2013.

³¹ Deliverable D2.6 of RobMoSys (H2020-EU.2.1.1. - INDUSTRIAL LEADERSHIP project under grant agreement number 732410)

³² https://robmosys.eu/wiki/baseline:environment_tools:papyrus4robotics

³³ https://robmosys.eu/wiki/baseline:environment_tools:smartssoft:smartmdsd-toolchain:start

³⁴ https://openrtm.org/openrtm/en/doc/aboutopenrtm/rtc_architecture

³⁵ <http://smart-robotics.sourceforge.net/>

³⁶ Mikel D. Petty and Eric W. Weisel. “A Composability Lexicon”. In: Proc. Spring 2003 Simulation Interoperability Workshop. 03S-SIW-023. Orlando, USA, Mar.2003

³⁷ https://robmosys.eu/wp-content/uploads/2019/10/D2.6_Final.pdf

With respect to system composition, composability must be addressed on three axes (see Figure 23): between different components (A), between alternatives of components (B), and between components and the application needs (C). The relations on all three axes need to be satisfied with respect to (I) syntax and semantic plus (II) application and technical level perspectives to enable composability for system composition.

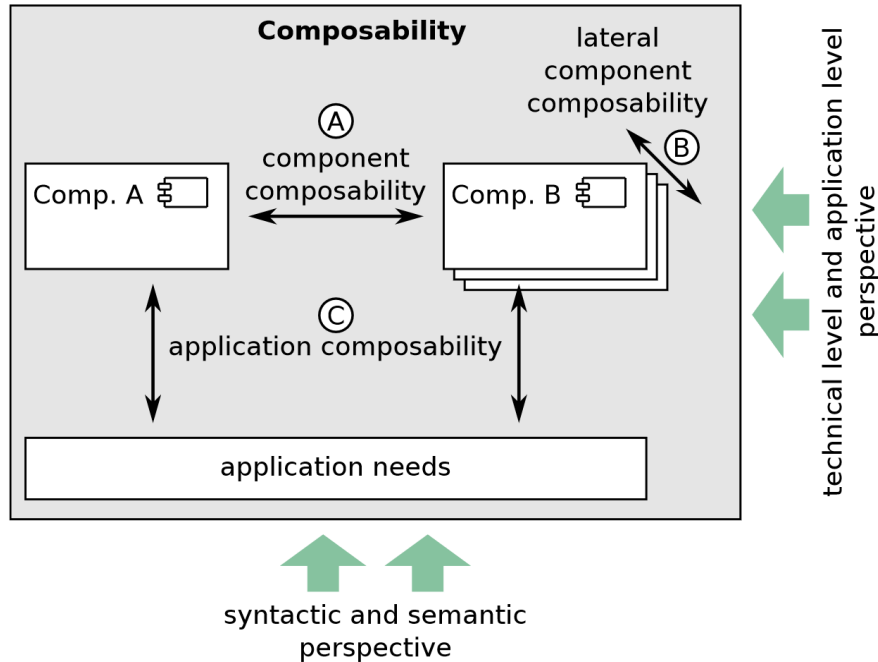


Figure 23: Occurrence of composability³⁸

The *composition workflow* is the activity of putting together building blocks. The workflow defines the steps and order to bring together all participants. It has to address their individual needs for system composition (see Figure 24). Stakeholders supply and use artifacts, e.g. provide or use components for composition. This requires prior alignment of what is provided and what is expected: functional boundaries, interfaces, and other necessary information. Collaboration within the workflow includes handover of these artifacts between stakeholders and workflow steps while ensuring, managing, and maintaining composability during the workflow.

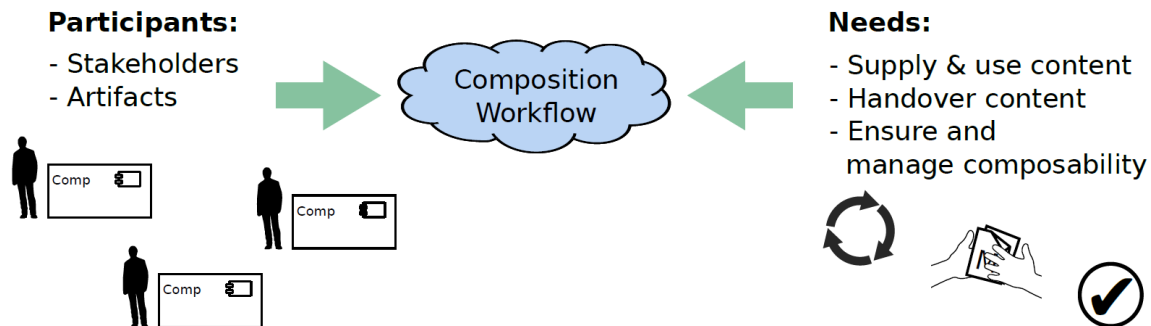


Figure 24: The composition workflow³⁸

The objective of the workflow is to allow to create and to use a structure for system composition. The workflow defines the roles and artifacts and the according steps for composition. Individual software development processes (such as e.g. Scrum, Unified Process) or other methodologies (e.g. Software Product Line (SPL)) can be applied within these steps to develop building blocks.

³⁸ <https://mediatum.ub.tum.de/doc/1399658/document.pdf>

Finding a workflow that defines and uses a structure requires to understand its stakeholders. The two main stakeholders in the ecosystem (see Figure 25) are content suppliers that provide building blocks and system builders that use them to compose new applications (systems). Even though there is a connection between them, they do not necessarily work together as a team or even know each other. Structural drivers shape or define the structure of the ecosystem. They provide guidance for the contribution of content. Within such a framework, all suppliers and system builders can rely on stable structures. They can work within clear boundaries and their building blocks can connect through the defined interfaces.

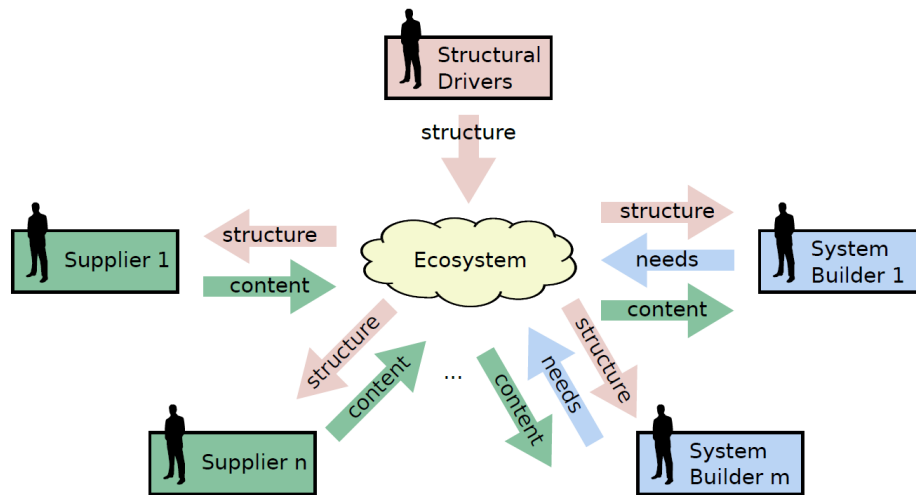


Figure 25: Stakeholders collaborating and interacting in an ecosystem³⁸

Support can have many forms. Adequate support in terms of *tools* for participants is critical towards system composition in an ecosystem as illustrated in Figure 26. Tools support in accessing and using the ecosystem by ensuring that parts adhere to its structure. Tools will realize the underlying structures of the approach and utilize them to prevent errors and provide automation, thus speeding up the development. Without adequate support by tools, participants of the ecosystem have a hard time “accessing” the methods and concepts. These concepts thus remain unused or are used in the wrong way, causing less acceptance and even leading to decreasing consistency and assets that cannot be composed. Tools play an important role in applying freedom from choice. Tools lower the effort, realize the handover, and realize the link between the different steps and participating roles of the composition workflow.

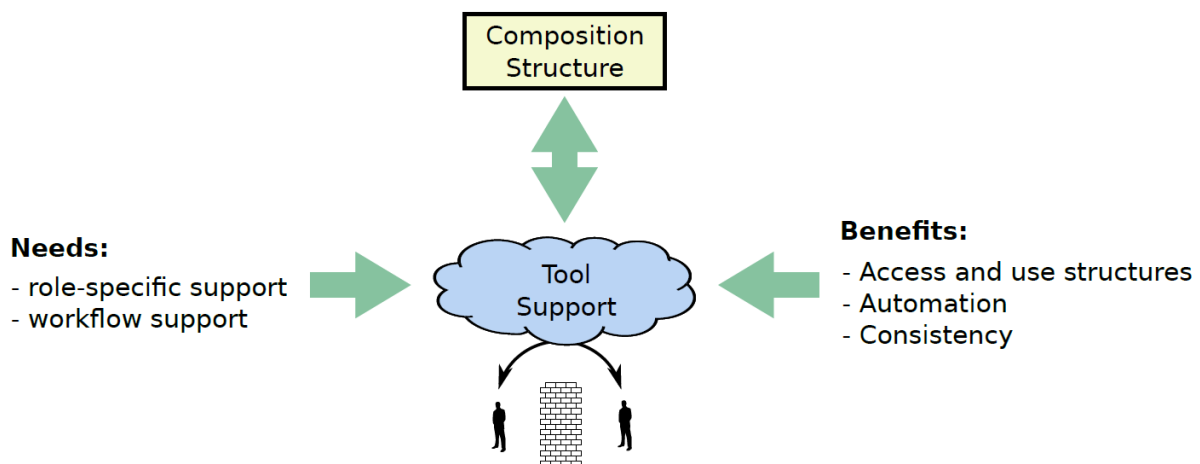


Figure 26: Adequate support via tooling for participants is critical towards system composition³⁸

6.2 Drone Embedded Software

The main objective of this R&D challenge is easing the integration and customization of qualified systems embedded in drones. The goal is to provide reference architecture of a flexible embedded platform inspired from the avionic practice (i.e., ARINC-653). This reference architecture will be based on modularity and separation of concerns. In addition, the focus will be on setting-up a common repository of generic qualified components maintained by the drone's community. The availability of such an integrated modular architecture will be a step forward with respect to the current state-of-the-art.

The reference architecture will not be proposed from scratch. The baseline of the reference architecture will be inspired from the PX4 stack and DIMA (IMA2G) avionics, currently adopted in the manned aircraft industries. The PX4 will be studied and compared to other existing autopilot architecture before proposing a new reference architecture. Also, a state-of-the-art analysis will be carried out in what concerns integrated modular architectures and components for mission-critical support as well as trusted communications. Importantly, robotics frameworks, such as ROS2 will be analyzed and if possible integrated in the architecture in order to increase the modularity and software composability.

6.2.1 Challenges in Existing Approaches/Practices

Current drone embedded architectures are organized in loosely coupled monolithic boards, each composed of processor, memory and communication resources (e.g. flight control, planning and vision boards). This separation ensures that the subsystems operate almost independently from each other and avoids interference coming from the other parts. However, this approach does not support the continuous development of drone applications such as the ever-increasing demand on autonomy. Drone embedded platforms must meet this demand, while still providing safety, robustness and a small footprint in physical size and power consumption. Another drawback is that the huge number of different resources has significantly increased the integration, customization, and maintenance costs in terms of component provisioning and handling.

In existing open autopilots, even if a modular object approach is employed, to the best of our knowledge, there is no component-based autopilot approach. However, this would be a very important contribution to raise the diversity of sensors and actuators that can be used for piloting a drone, and especially in the scalability of drones in their different versions and improvements.

6.2.2 Specification/Guidelines

The **COMP4DRONES** project will face this challenge by developing a compositional and integrated drone embedded reference architecture following the IMA principles, adapted to, and still considering, the drone resource constraints. Figure 27 summarizes these principles.

6.2.2.1 Processing Partitioning

The main idea is to place functional modules (applications) on processing modules partitioned with respect to space (resource partitioning) and time (temporal partitioning). Resource partitioning allocates memory and I/O resources in a static manner via configuration tables (contracts), which guarantee protection against any modification from other partitions. Temporal partitioning defines a periodic sequence of slots, statically organized in a time frame, so that functional modules are periodically executed at fixed times.

6.2.2.2 Communication Partitioning

Communication between function modules (applications) will be placed on shared networks. Like in processing partitioning, communication is divided into virtual links, with a bounded bandwidth, dedicated to the traffic from single applications. This guarantees that no function can use the network beyond its contract.

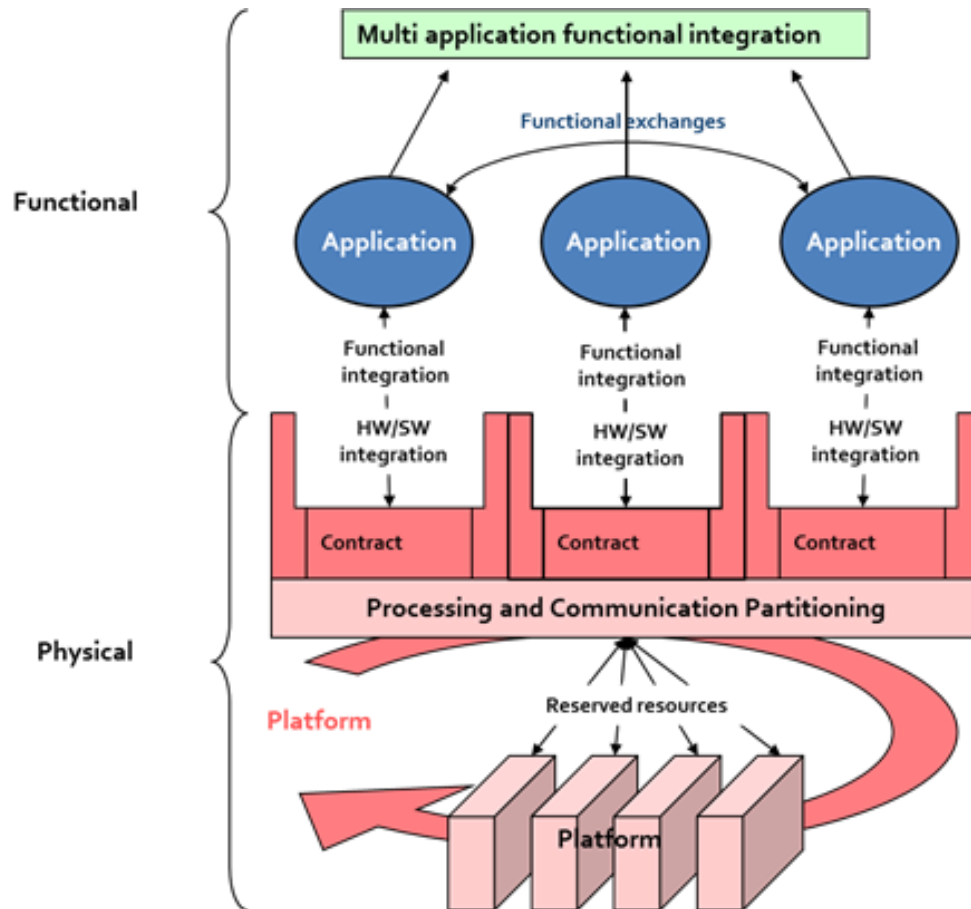


Figure 27: Modular drone Embedded COMP4DRONES Reference Architecture

6.2.2.3 Functional Independence of Applications

Applications implement services hosted on the platform partitions. The platform offers basic services such as scheduling, communications, health monitoring, among others. In this way, applications can be potentially moved to new platforms with minimal changes to the code. Similarly, a specific platform can be used to support multiple different applications with slight changes. This approach requires of course an important change in the design paradigm for all engineering parties to switch from a pure “top-down” approach (with upper-level design decisions) to a different approach in which applications must use existing platform services. This is why engineering tools are needed.

6.2.2.4 Standardized Interfaces

In order to enable this level of portability, it is necessary to harmonize the platform definition and the API to integrate applications in the partitions of this platform. Standardized APIs enable technology transparency and avoid any impact on an application either during development or execution.

6.2.2.5 Expected Benefits

By following the above principles, **COMP4DRONES** pursues the following specific objectives:

- **Incremental assurance.** Applications sharing the same platform can be qualified independently and changes to one will not impact the assurance achieved by others, as long as the platform resources allocation is not modified. The scope of re-verification is well-defined and bounded in case the platform resource allocation and contract are modified. The same applies in case the platform design is modified (additional services, improved performances, obsolescence management).

- **Reduced physical resources.** The number of processing, communication, and auxiliary resources (cables) is reduced as well as the vehicle weight leading to better power consumption efficiency and then to greater endurance. This also leads to cost reductions on platform development, configuration and maintenance.
- **Better disaggregation of the drone supply chain.** Keeping the independence between module providers (both application and platform modules) facilitate the evolution of the market supported on generic and reusable components. It also enables indirect savings through common development environments, separate platform / applications life cycles.

Moreover, we plan to unify the existing software bricks for drones, by creating a repository of software components. In fact, libraries must be proposed in order to first unify the current software components (e.g., Paparazzi UAV project) and to enhance the reuse and interoperability. These shared components must be well defined and characterized, to check their non-functional properties and validate their use regarding a given context of new missions by taking into consideration accuracy and safety constraints.

6.3 Minimizing the Design and Verification Effort

One of the main objectives of COMP4DRONE is minimizing the design and verification efforts for complex drone applications, ensuring that new services based on drones can be devised and offered by companies of any kind, especially SMEs³⁹. As a result of the research and development activities to be carried out in the project towards this objective, **COMP4DRONES** will deliver an agile, system engineering framework providing the system engineer with all the resources required in modeling, composing, optimizing, verifying, and implementing complex services based on the use of fleets of drones. The availability of such an integrated design and verification framework will be a step forward with respect to the current state-of-the-art.

Drone systems are considered as open and distributed systems that can interact (coordinate) in an autonomous manner to achieve certain specific goals⁴⁰. These devices operate in an open environment, where there is no complete knowledge of the surrounding world (i.e. it is not possible to have the complete state of the world - it is necessary to take into account the uncertain nature of the environment). Thus, new approaches (methodologies and technologies) for validation and verification are necessary to be able to deploy safe and secure system solutions.

In addition, these kinds of environments need a tight combination of the cyber capabilities (distributed application SW, communication protocols, cyber-security issues, etc.) of the corresponding devices with the physical world (different variety of sensors and actuators) requiring novel verification and validation challenges.

6.3.1 Challenges in Existing Approaches/Practices

The cost of drone software accounts for a large share of the overall cost of a typical drone project, including component development, system integration, verification, validation and customization. The current tooling landscape for engineering drone-embedded systems is quite fragmented. It is characterized by a series of isolated tools with different and sometimes complementary capabilities ranging from specialized coding towards sophisticated simulation environments. However, tools still need (a) harmonization to enable agility and interoperability, and (b) clear separation of concerns for efficient disaggregation of engineering roles at different abstraction levels.

These are the main challenges:

- Connectivity to combine both open and closed networks.
- Plethora of resource constrained devices.

³⁹ Drone, "Commeriale drone applications," 2018. [Online]. Available: <http://dronenodes.com/commercial-drone-applications/>

⁴⁰ Intel, "Commercial Drones," 2018. [Online]. Available: <https://www.intel.com/content/www/us/en/drones/drone-applications/commercial-drones.htm>

- Highly unpredictable non-controlled environments.
- Dynamic and heterogeneous environments, not only in terms of number of devices but also in terms of number of protocols, infrastructures and platforms.
- Non-functional requirements such as safety and security.
- Highly unpredictable behavior of autonomous devices/systems.

6.3.2 Specification/Guidelines

The availability of such holistic system-engineering framework for drone applications will be an innovative breakthrough over the design methods currently being used. The framework will simplify the design and verification effort of drone engineers and therefore, will enable the development of more advance, more complex drone projects at an affordable cost. This will open the way for a wide ecosystem of competitive European companies providing advanced services based on drones, most of them SMEs.

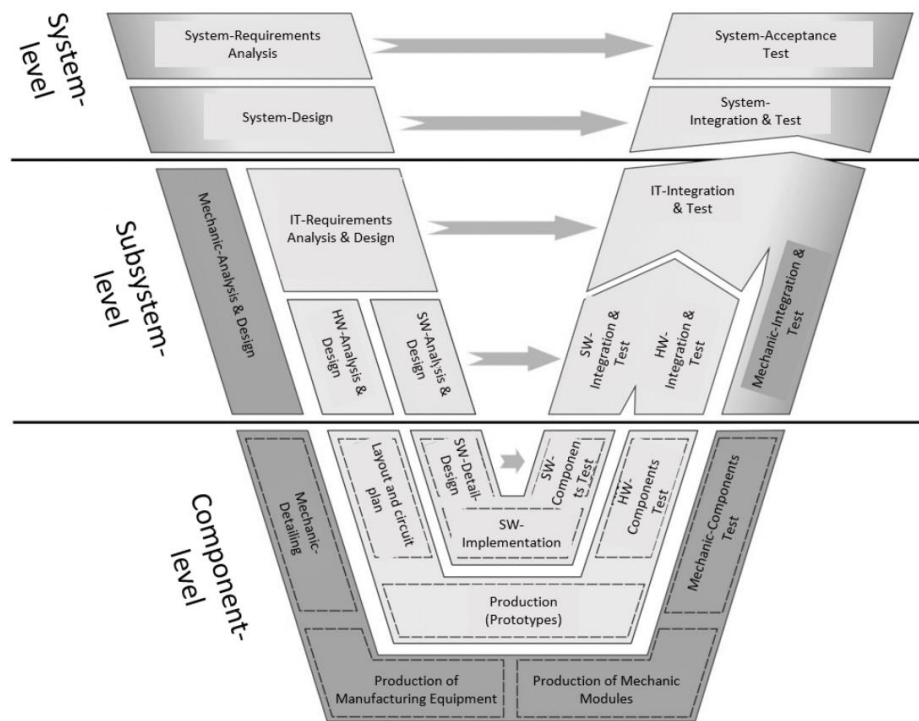


Figure 28: Agile and interoperable COMP4DRONES tool ecosystem (WP6)

Additionally, as the **COMP4DRONES** follows a modular and incremental approach with specific computation and communication mechanisms (e.g., partitioning, protocols, and contracts), tooling must be capable to be customized to manage configurations in a composable and agile way.

COMP4DRONES's vision is to have an open, sustainable and agile drone embedded development ecosystem. To achieve this vision, the project will deliver a model-based (rather than code-centric) engineering framework to specify, compose, integrate, verify and implement drone modules and systems. Figure 28 (presented in detail in D6.1) summarizes the main concepts behind the **COMP4DRONES** engineering framework.

6.3.2.1 Model-driven Approach

Current open drone embedded platforms (such as ArduPilot, Paparazzi or Dronecode) embrace communities of hundreds of stakeholders, which need to be preserved and strengthened. However, these communities are still rather fragmented, by representing specific technologies. To strengthen these platforms, and to enable interconnections between them and new ones (e.g. AI-based technologies), the **COMP4DRONES** project envisions a model-driven approach built on top of, or rather

around, the current code-centric platforms. The main principle behind this approach will be to agree on common modelling structures that share these platforms, formalizing the specification of the different platform patterns by using the proposed modelling structures, and integrating these modelling structures in the **COMP4DRONES** tools ecosystem. The modelling structures will be realized by an agreed meta-model. We are aware that agreeing on common modelling structures is a long process, but the project will build on existing modelling languages, such as RobMoSys approach (e.g., to enable composition and separation of concerns in the robotics domain), and MARTE/AADL (e.g. for timing and performance concerns).

6.3.2.2 *Compositional Approach*

COMP4DRONES follows a compositional/modular approach for facilitating distribution of problem solving, reuse, and easy customization. This approach distributes the whole drone system into a set of distinct modules or components that can be developed independently and then plug together. The **COMP4DRONES** tackles the problem of drone platform composition, by means of three concepts: components, interfaces and patterns. A component can be viewed as a black box, which is defined only in terms of its inputs, outputs and transfer characteristics. A black box encapsulates distinct separate functions and abstract away the internal component mechanics, so that it is only represented by interfaces in the form of “data sheets”. Data sheets provide information about functional aspects but also non-functional properties, either to other application components (for horizontal composition) or to platform components (for vertical composition). Patterns are needed to provide meaningful mechanisms to interconnect those components. These three concepts will be part of the **COMP4DRONES** meta-model.

6.3.2.3 *Separation of Concerns*

Thanks to a methodological separation of concerns and model abstractions, many problems can be addressed in isolation, with composite models at successive stages of drone system development and deployment. The **COMP4DRONES** modelling structures will be oriented to the management of the interfaces between different roles (drone domain expert, component supplier, system integrator, quality manager), at different levels of functional abstraction (e.g. from navigation and control to mission decision making), and different levels of platform abstraction (computation, coordination, configuration, communication, composition and transformation), in an efficient and systematic way. These composition structures will be organized in semantically related viewpoints, which provide common and harmonized modelling views to perform multiple development activities, such as architectural design, validation and verification, code generation and deployment. **COMP4DRONES** will enable separation of concerns as a mechanism to distribute the complexity of the development problem and to collaborate in a consistent way.

6.3.2.4 *Tool Interoperability*

Tools heterogeneity, in the scope of either a single company or in a full ecosystem, is one of the main barriers to enable agile engineering frameworks. The **COMP4DRONES** project will address interoperability through the definition of a syntactically and semantically well-defined meta-model. Tools will use this meta-model to directly target specific technological platforms (e.g., to automate code generation or simulation) or to perform model transformations when existing tools use different modelling structures.

6.3.2.5 *Expected Benefits*

This approach aims to bring the following benefits:

- **Correct-by-construction design.** Composition as addressed by the **COMP4DRONES** will foster preservation of component properties (namely, composability) but also predictability of the global system properties (compositionality). To make an example, a component supplier could have implemented a particular algorithm for Detect & Avoid in a past project and a system integrator would like to reuse this software in a new project as a basic building block to get full

navigation functionality. To make this reuse happen, the model of the Detect & Avoid algorithm must be composable with well-defined interfaces. Interfaces must be more than simple API, since through these interfaces the component defines its functional and non-functional perimeter (assumptions and guarantees) and might be tuned to a new application context or simply verified against different application-specific functional or non-functional requirements (e.g., different detection-reaction time).

- **Validation of design.** This includes for instance real-time scheduling and TCP path planning.
- **Automation of labor-intensive tasks.** This model-centric **COMP4DRONES** approach creates a great potential for automation of software production. It is worth to be noticed that models can be equally understood by domain experts and by machines (for execution and validation). **COMP4DRONES** will deliver a software engineering landscape formed of mature tools and associated tool-chains specifically targeting the autonomous nature of drones. The primary benefits of automating labor-intensive tasks are engineering cost reduction, productivity, and reliability.
- **Seamless integration.** **COMP4DRONES** engineering tools will provide a seamless interoperability between system-level activities and component-level activities (i.e., design, implementation, validation and verification), along with transversal quality activities (e.g., compliance management, safety assessment). This level of interoperability aims to lower the threshold of drone component and system development and assurance in face of rapidly changing product features and market needs.

6.4 Safe Autonomous Decisions

The market of drones is rapidly growing in both the consumer and business areas. Drones are ideal in many applications such as safety, surveillance, construction, agriculture, logistic and many more⁴¹. One of the main challenges is that current drone systems are too complex to be safely operated by humans. This can result in harms to people and property. Relying on humans is not only unsafe but is also inefficient in any of these applications.

6.4.1 Challenges in Existing Approaches/Practices

Drones are perfect for doing all sorts of important jobs that are considered dull, dangerous, dirty, distant or difficult such as for example: precision farming, coast guard search and rescue, mapping fire or natural disaster areas, and surveillance. In order to enable the targeted operations, there are some challenges to address first. Today, UASs are excluded by regulatory frameworks from many of these kinds of operation scenarios due to safety concerns. These concerns include making sure that the drones are safely and securely integrated into a mixed manned and unmanned aviation space and the Air Traffic Management (ATM) network. In addition, most drones already have autopilots, but these are generally not reconfigurable and do not have the ability to make high level decisions, e.g. deciding on what procedures to adopt due to changes in operational conditions such as bad weather, communication issues, or following ATM instructions.

6.4.2 Specification/Guidelines

For this reason, in the **COMP4DRONES** project we will take an integrated and transversal approach to the problem. We will start from developing Artificial Intelligent systems capable of maneuvering safely the drones as well as robust and communication systems for high reliability of the control connection (see Figure 29).

The **COMP4DRONES** project aims to design and develop reconfigurable drone algorithms and components with the capability to autopilot an aircraft system and make safe autonomous high-level

⁴¹ Goldmansachs, "Technology Driving Innovation Drones," [Online]. Available: <http://www.goldmansachs.com/our-thinking/technology-driving-innovation/drones/>

decisions concerning an individual or cooperative situation. The project will build on the following key concepts.

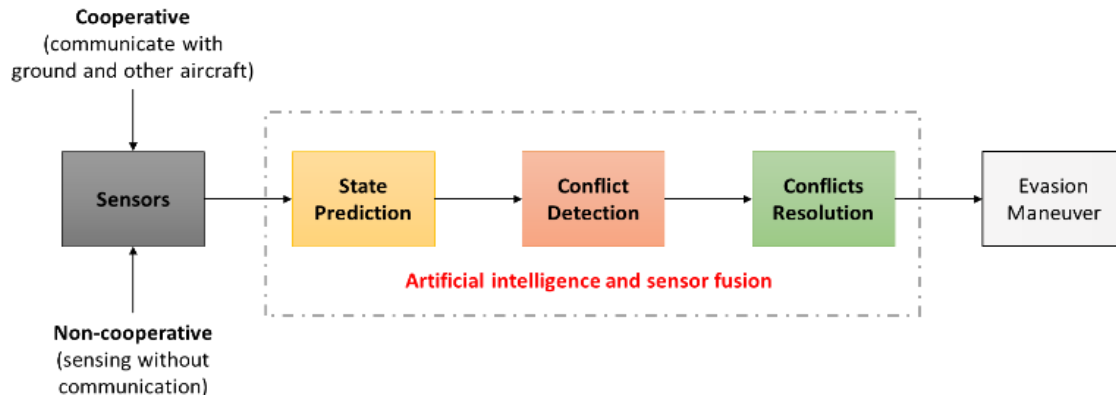


Figure 29: Safe Autonomous Decision

6.4.2.1 Advanced Navigation

Most navigation systems in drones use the space-based Global Navigation Satellite System (GNSS). Despite advances in this technology, GPS-based systems will not meet the demands of future autonomous vehicles for several reasons. First, GPS signals alone are extremely weak and unusable in certain environments. Second, GPS signals are susceptible to intentional and unintentional jamming and interference. Third, civilian GPS signals are unencrypted, unauthenticated, and specified in publicly available documents, making them spoofable (i.e., hackable). The **COMP4DRONES** project will rely its solutions not only on GPS, but on a suite of other sensor-based technologies such as cameras, lasers, and sonar. These technologies will be based on high-performance computing architectures (GPUs or CPUs) such as neuromorphic systems. **COMP4DRONES** will develop smart navigation systems that will exploit neuromorphic hardware together with safe ad-hoc software routines. This will provide neural network closed-loop perception and computational methods that can enable a new class of computing technologies for ultra-low power navigation and exploration in unknown environments.

During the project, we will develop smart navigation systems that will exploit neuromorphic hardware together with safe ad-hoc software routines. The objective of our work is to provide neural network closed-loop perception and computational methods that can enable a new class of computing technologies for ultra-low power navigation and exploration in unknown environments. We will improve the state of the art by developing efficient event-based neural network algorithms and on-line learning strategies in neuromorphic hardware for on-demand, massively parallel, and low-power computation. The tasks that we will explore are essential for autonomous navigation and include Simultaneous Localization and Mapping (SLAM), optic-flow computation, ego-perception and closed-loop motor control with a tight interconnection among perception and control. The output of the project will be novel data-driven neural network architectures validated in FPGA platforms. These computational architectures will target drones or other robotic platforms, enhancing security and performance of autonomous vehicles therefore stimulating the European industry with a new generation of adaptive architectures.

6.4.2.2 Autonomous Decision-Making for Planning.

To achieve reactive and fault-tolerant mission planning for drones, new planning algorithms will be developed and tested, where the mission plan can be varied autonomously, on-line and in real-time by each drone, in response to events which might affect the validity of the current mission planning (ATM interference, obstacle avoidance, on-board health management, drone malfunctions, communication faults, among others). The aim of this on-line re-planning is to achieve the assigned mission targets, as far as possible, and to keep the overall drone behavior coherent, even if one or more drones modify the planned mission without sacrificing the efficiency, flexibility, reliability and security of the drone system.

AI-based techniques will be investigated, where mission planning will consist in determining the optimal path for a drone to accomplish the goals of the mission in the smallest possible time. The algorithms to be developed in **COMP4DRONES** must be adaptive with respect to all possible faults and variations that may occur during a mission as well as to the drone processing power and communications availability.

Part of our ambition is the development of sensory fusion technologies for real-time applications, such as visual object recognition, attention, and multi-sensory integration. To do so, we plan to improve the state of the art by developing data-driven and event-based computational modules, that will perform on-demand processing and that will exploit the computational properties typical of neuro-biological systems, but embedded in novel neuromorphic architectures. This will result in an increase of the efficiency and robustness of computational architectures that will finally be able to be applied in autonomous vehicles with tight power constraints (as drones). The anticipation and the foresight of novel massively parallel and brain inspired IC architectures can greatly boost Information and Communication Technology (ICT) industry and it is one of the most relevant challenges when building autonomous cognitive systems.

6.4.2.3 Safety Monitoring

Drone safety will be addressed at runtime to face with the limited computation resources by analyzing the behavior of the algorithms. Certain failure scenarios have been anticipated, together with potential reconfigurations that assure that critical functionality remains assured. However, it is not possible to anticipate all failures, in particular not the combination of failures, as the required database would become too big (combinatorial explosion). Therefore, some failures have to be handled at runtime; the system should autonomously take a suitable action. **COMP4DRONES** will use safety monitors looking at past and current states, in order to verify correctness and validate the system; or focus on future states, with prediction algorithms and actively diminish risk by assessing threats. To ensure correct runtime functionality in a UAV component, its execution will be monitored according to predefined invariants that essentially specify a contract for the dynamic behavior of the component. In the case in which it is not possible to find a feasible solution before a decision must be made, a safety mechanism will need to take place.

6.4.2.4 Expected Benefits

By following the above principles, the **COMP4DRONES** will achieve the following specific objectives:

- **Reliable navigation.** The implementation of sensory fusion algorithms in dedicated hardware accelerators (such as neuromorphic systems) is beneficial for several reasons that include a) concurrency in processing (memory and computation are co-localized); b) event-based processing which enable to process information on demand; c) low-power consumption; d) compatibility with embedded systems; e) ability to learn and adapt to specific situations. The exploration of sensory fusion to elaborate sensor's data on the drone itself without the need of transmitting high-data rate to cloud services is appealing for increasing security and the overall system abilities.
- **Higher levels of autonomy.** Autonomy ranges from the simple motion on specific perimeters (geofencing), to the ability to be self-sufficient in a complex environment. The benefits of not having a human pilot are many. Human error is eliminated from the safety equation, the automated systems are not subject to fatigue, and are repeatable. Automated drone systems also have structured and auditable decision points backed up with data and facts. Response time and availability, which are critical to enterprises, are shortened significantly when using an automated system. Increasing levels of AI in "smart" sensory-motor loops allow robots to perform in increasingly dynamic uncertain complex environments with increasing degrees of autonomy.
- **Safer operations.** Providing smart autonomous drones with a runtime monitoring system allows for some kind of envelope of permissible behaviors, without compromising safety due to uncertain and complex operations. This solution enables adaptive navigation and planning that cater for self-diagnostic and self-correcting regulation of system performance from the point of

view of safety. This component would act as a runtime manager to detect abnormal robot behavior, triggering a different execution mode (e.g. a safe degraded mode), or re-plan the mission altogether. This could, for example, make drones avoid situations or behaviors that are expected to be harmful to people, their property or other vehicles.

6.5 Trusted Communication

Regarding the security related aspects, the innovation that will arise by means of the present project will be a trusted communications component. This component will be in charge of dealing with the identification of cyber-security threats, their risk and scope evaluation and the deployment of the decision and/or actions to mitigate or protect against those attacks. This will be reflected on the following innovation aspects that will try to advance on the current state of the art.

6.5.1 Challenges in Existing Approaches/Practices

Communication environment can be qualified as trusted as long as it satisfies the three classical security requirements (confidentiality, integrity, availability), to which are added the performance and safety requirements⁴². The safety requirement obviously cannot be entirely fulfilled by means of communication primitives; yet it has to be addressed in the design of communication and cybersecurity components in order to ensure a smooth joint operation of safety-capable communications. The requirement of performance can straightforwardly be linked to that of availability: both aim at ensuring a satisfying communication quality irrespective of the communication environment (non-malicious and malicious factors).

The drone mobility pattern introduces a large variability in the perceived communication environment and the quality of available communication technologies: depending on the drone coordinates and altitude, some technologies may be available or unavailable; some of them, excellent in one position, may become unusable in another e.g. due to poor throughput, latency or jitter. At the same time, cybersecurity requirements will require that the drone-to-drone and drone-to-infrastructure communication links be protected against cyberattacks. This protection will have to offer an excellent quality especially with respect to the integrity and availability security properties. Finally, an orthogonal challenge with respect to trustworthy (reliable *and* secure) communications consists in the fact that drone embedded trustworthiness enforcement modules (e.g. communication stack, intrusion detection system, etc.) will have to accommodate the inherent drone constraints, e.g. in terms of memory and computing power.

6.5.2 Specification/Guidelines

Figure 30 proposes a repartition of **COMP4DRONES** key features beyond the state-of-the-art based on this requirements categorization. These features belong to four key principles (described below).

6.5.2.1 Lightweight Communications

In order to achieve a satisfying communication performance level, the drone must rely on a finely tuned communication framework. This latter comprises efficient radio technologies, lightweight radio stacks and middleware, as well as fulfilment of the dependability property. “Lightweight communications” natively means lightweight impact in terms of required processing power (and accordingly, energy consumption) and memory consumption; the idea is that the communication system can answer its strong reliability requirements without hindering other subsystems. To that aim, specific low-footprint high-performance communication stacks have to be built. Lightweight will also be meant literally, with the design of lightweight (small-sized) powerful communication hardware. Lightweight character will also come from the communications system modularity: the communication stack will be able to exploit the available hardware communication technologies in order to optimally communicate. For example, (lighter) Wi-Fi-only drones will be able to seamlessly share the cellular connectivity of heavier drones or

⁴² Eray Yağdereli, Cemal Gemci, A Ziya Aktaş, "A study on cyber-security of autonomous and unmanned vehicles," The Journal of Defense Modeling and Simulation, Vols. Vol 12, Issue 4, p. 369 – 38, 2015.

ground stations. Eventually, the overall system will exploit this composability property to allow drones to mutualize their communication resources. In particular, the lightweight communications stack should include facilities for multi-link and robust communications.

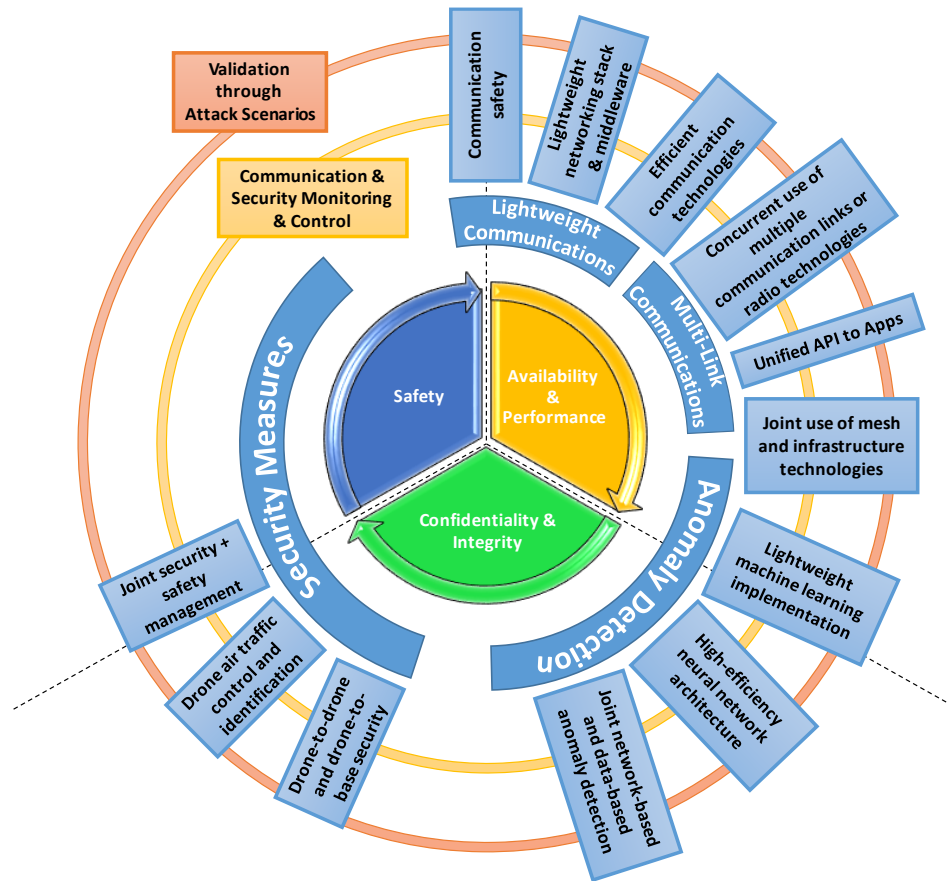


Figure 30: COMP4DRONES trusted communication framework: key properties

6.5.2.2 Multi-Link and Robust Communications

This principle denotes the capability for the drone to communicate efficiently with other drones as well as with the infrastructure. The efficiency property to which it is referred here means both the fulfilment of applicative requirements and the capability to leverage on the neighboring communication environment, however diversified it may be (e.g. availability of multiple radio technologies and / or communications links). This global communication support must on turn be made available to drone-embedded applications, which means that a unified API hiding the communication means diversity will have to be developed. It has to be noted that the capability to use the best available communication technologies also allows to suit the traffic type safety requirement (e.g. dynamically give highest priority to traffic management, dynamically enable drone-to-drone when doable without hindering other communication types).

The ambition within **COMP4DRONES** is to develop an improved path manager for use in communication with (unmanned) aircraft, and enable it for multiple, heterogeneous communications technologies at OSI level 4. Specific technologies will be selected based on the concrete demonstrator among, for instance, BT-LE, IEEE 802.11, relevant subtype to be determined⁴³, LTE/5G with LAN capabilities, VSAT or sat

⁴³ IEEE Standards Association: Std 802.11-2016, IEEE Standard for Information technology, "Telecommunications and information exchange between systems Local and metropolitan area networks—Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," 2016. [Online]. Available: <http://standards.ieee.org/findstds/standard/802.11-2016.html>.

phone (if available), NB-IoT⁴⁴, etc. The path manager should select the best combination of technologies for the flight situation (phase of flight, position/altitude, speed, etc.) and the payload requirements (amount of data to transmit, time constraints (for instance, whether real time communication needed for the active mission)).

The purpose of a higher-reliability connectivity system is to ensure that important information reaches the vehicle (or the ground). For that purpose, it is important that applications on board the vehicle (whether related to the control of the vehicle or information to/from the payload) are able to adjust the use of the communication channel to the available bandwidth and latencies. To facilitate such prioritized communication, an API will be developed to communicate connection quality and availability as well as available resources (unused channels, if any) that can be employed if needed by the applications. The API will also allow applications to send requests for higher levels of connectivity if needed.

In some interesting use cases drones have to operate in difficult communication environments where there might not always be a (viable) direct connection between endpoints. Robust communication mechanisms can help alleviate this issue by offering Store-Carry-Forward message transfer, reducing the impact of interruptions in the end-to-end path from drone to ground stations. Such a mechanism is called Disruption Tolerant Networking (DTN) and will for this project be further developed for the specific communication patterns and small footprint of drones. A DTN instance is responsible for communication even if there is no direct link or path to the destination. In this case, the Store-Carry-Forward principle is applied where bundles are stored on the node itself until the destination becomes available again. The project will develop Software components for robust communications by means of store- and forwarding methods, using mechanisms from Disruption Tolerant Networking (DTN).

6.5.2.3 Anomaly Detection

Anomaly detection represents a strong component of reactive security, which affords the drone with the capability to detect and react to threats in real-time. Specific machine learning primitives will have to be conceived that will have to be lightweight enough to be supported by the drone while at the same time allowing it to detect unknown threats. In this particular field of security, "lightweight" means that the proposed primitives will not exaggeratedly drain computing power. For example, the AI-based intrusion detection system will be specifically tuned in order to consume very few energies in the detection phase, while offering a very good level of performance. Anomaly detection will have to reason on both drone-to-drone and drone-to-base links. It will also reason on network metadata (e.g. protocol anomalies) and parseable data (data plausibility) for increasing the range of possibly detected anomalies.

COMP4DRONES will design distributed lightweight machine learning primitive for carrying out anomaly-based intrusion detection based on network metadata and applicative data. Especially, a neural network architecture will be conceived that will be able to intelligently parse data relevant to multiple network or applicative data and temporally match anomaly detection results on them with one another in order to refine the overall detection quality. The neural network framework will be specifically built for considered traffic that does not only include hyper optimization but also specify the best detection methodology (classification, prediction, network behavior "coding" by means of an auto encoder) and the corresponding pre-processing steps (identification of the most relevant features and feature extraction). Special care will be taken in all steps of the neural network design to keep its surveillance operation affordable on the drone itself. The learning operation, on the other hand, will be designed in order to be remotely performed, since it would likely be too heavy for the drones.

6.5.2.4 Security Measures

COMP4DRONES mostly concerns design decisions related to preventive security countermeasures that make them highly relevant for the drone environment. These include especially the joint support of security and safety in the decision taking, the capability to operate on drone-to-drone and drone-to-base

⁴⁴ A. D. Zayas and P. Merino, "The 3GPP NB-IoT system architecture for the Internet of Things," 2017 IEEE International Conference on Communications Workshops (ICC Workshops), Paris, pp. 277-282, 2017

links, and the development of aspects of security related to drone air traffic control and identification (secure drone authentication scheme).

6.5.2.5 *Expected Benefits*

COMP4DRONES general objectives for trusted communications comprise the monitoring and management capability on the one hand, and the scenario-based validation on the other hand:

- **Monitoring and control** represent a general objective both for the communication environment and configuration and for the preventive and reactive security countermeasures. This objective especially means that the communication framework and its security subsystem will be uninterruptedly kept in line with the environmental (location, interactions with other drones, threats, safety status) requirements.
- **Scenario-based validation** refers to the validation of the **COMP4DRONES** key concepts regarding trusted communications by means of actual scenarios, especially involving actual operations that can be observed during the course of an attack (recon, weaponized, deliver, exploit, etc.)

7 Conclusion

In this deliverable, first, as the project still in its early stages, we provided methodologies for supporting the requirements collection, and for measuring the project success criteria.

Second, we have introduced a general procedure for developing drone systems, and a set of key concepts that are required to specify the project methodology and workflow. These concepts include U-Space, drone categories, and SORA (Specific Operations Risk Assessment): (a) U-Space is a set of new services and specific procedures designed to support safe, efficient, and secure access to airspace for large numbers of drones; (b) drones are classified into three categories based on their operations risks concerns: open, specific, and certified; (c) the Specific Operations Risk Assessment (SORA) provides guidance to both the competent authority and the applicant as to what is required for a national aviation authority authorization required to fly a UAS in a given operational environment.

Third, the drone system has different stakeholders with different needs/requirements. Therefore, the drone system requirements are described from the users' perspective (as a summary of D1.1), and from service providers and system integrators viewpoints.

Forth, an initial methodology for drone systems development has been presented based on existing state of the art system engineering approaches in the avionics domain. This methodology will be detailed and elaborated in the coming deliverables (D2.3 and D2.4).

Finally, the challenges and project contributions to improve the existing technologies to enable easy customization of drones and their safe operation have been described. The improvements are divided into four groups: integrated modular reference architecture, minimization of the system design and verification, safe autonomous decisions, and trusted communication.