COMP4DRONES

DELIVERABLE

D2.3 – 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. Mauro Gil (INDRA)
Website	www.comp4drones.eu

ECSEL Joint Undertaking
Electronic Components and Systems for European Leadership

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826610

Document file	
Authors:	See Page 4
Internal reviewers:	Eugenio Villar – UNICAN Fedor Ester – DEMCON
Work Package:	WP2
Task:	T2.1 and T2.2
Nature:	R
Dissemination:	PU

Document History

Version	Date	Contributor(s)	Description
V1.0	02/07/2021	See Page 4	Complete draft of the deliverable
V1.1	09/07/2021	Mahmoud Hussein, Reda Nouacer	Deliverable ready for the internal review process
V1.1	20/07/2021	Eugenio Villar, Fedor Ester	Internal review process completed
V1.2	27/07/2021	Mahmoud Hussein, Reda Nouacer, Carlo Tieri	Final version of the deliverable
V1.4	27/01/2022	Réda Nouacer Ansgar Radermacher	Consideration of recommendation PR2-R3 - Scope and positioning of deliverable D2.3 - Introduction section: adding clarifications about the relation between the project work packages, by rephrasing some paragraphs and better explaining the figure 2. - C4D Workflow section: addition of figure 5 with comments to clarify the vision of the project on how the multiple achievement of the project are combined to enable the design of UAV system.
V1.5	01/02/2021	Réda Nouacer Ansgar Radermacher	Final review

Keywords:	Methodology, Workflow, Framework, Agile Development, Architecture, Safe Decision, Trusted Communication, Tools
Abstract (few lines):	This deliverable provides an updated version of the **projects' framework and methodology** (an initial version is described in the deliverables D2.1 and D2.2). In this deliverable, we first give an overview of the **COMP4DRONES** project procedure for developing drone systems. The main driver of this procedure is the system's **concept of operations** which include the system functions, its environment, and regulations concerning the system operations. Second, the different **regulations** that need to be taken into account during the system development are discussed. Third, the project's **framework** of the key enabling technologies is described. It includes the **key enabling technologies** for drones, and the different components that are being developed in the project related to these key enabling technologies. Fourth, **guidelines** to support the development of the drone systems, and the project methodology to develop such systems are discussed. Finally, the **tools** that are developed in the project to support the methodology/workflow are presented.

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 Nº 826610

D2.3 Authors:

Partner Name	Contributors
CEA	Mahmoud Hussein, Reda Nouacer
SKYA	Philipp Knopf
TEKNE	Carlo Tieri
TNL	Maurits de Graaf
SIEMENS	Federico Cappuzzo
UNIVAQ	Vittoriano Muttillo
ANYWI	Ozren Cecelja, Morten Larsen
ALM	Ludo Stellingwerff
ALTRAN	Quentin Ruiz, Shivaraj Kumar Velayudhan
ABI	Tiziana Fanni
AI	Stefano Delucchi
SM	Jiri Bartak, Petr Bartak
ACORDE	Fernando Herrera, David Abia
IMEC-NL	Federico Corradi
ENSMA	Emmanuel Grolleau
HIB	Elena Muelas
IFAT	Rainer Matischek
MODIS	Daniela Parletta
UNISANNIO	Valerio Mariani, Luigi Iannelli

WP2 Contributors:

SCALIAN, ALTRAN, INDRA, CEA, ABI, TEKNE, ROT, TNL, HIB, IFAT, ENAC, ANYWI, TUE, AI, ALM, TOPVIEW, SM, MODIS, SIEMENS, ENSMA, ATE, SHERPA, AIK, BUT, UWB, UNIMORE, UNISANNIO, UNISS, UNIVAQ, IMCS, LMT, IMEC-BG, SOBEN, IMEC-NL, SkyA, ACORDE

Table of Contents

Table of Figures

Table of Tables

Definitions, Acronyms and Abbreviations

Acronym	Title
ARC	Air Risk Class
BVLOS	Beyond Visual Line of Sight
C2 link	Command and Control Link
C3 link	Command, Control and Communication
CNN	Conventional Neural Network
ConOps	Concept of Operations
COTS	Commercial off-the-shell
DAA	Detect and Avoid
DFG	Difference Frequency Generation
DTN	Disruption Tolerant Networking
EASA	European Aviation Safety Agency
FAA	Federal Aviation Administration
FMS	Flight Management System
FPGAs	Field Programmable Gate Arrays
GCS	Ground Control Station
GNC	Guidance, Navigation, and Control
GNSS	Global Navigation Satellite System
GRC	Ground Risk Class
HSI	Hyperspectral Imaging
HSM	Hardware Security Module
IDS	Introduction Detection System
IMM	Intelligent Mission Management
IOLC	Intelligent Outer-Loop Control
IPS	Indoor Positioning System
ISHM	Intelligent System Health Management
IVVQ	Integration Verification Validation Qualification
Lipo	Lithium Polymer battery
LPWAN	Low Power Wide area Network
NN	Nearest Neighbor classifier
OTH	Over the Horizon Communications
OSO	Operational Safety Objectives
SAIL	Specific Assurance and Integrity Level
SORA	Specific Operational Risk Assessment Methodology
SLAM	Simultaneous Localization and Mapping
SOI	System-Of-Interest
SPL	Software Product Line
RLC	Rapid Learning Cycles
ROS	Robot Operating System
RTOS	Real-time operating systems
TMPR	Tactical Mitigation Performance Requirements
UAV	Unmanned Aerial Vehicle
UAS	Unmanned Aerial System
UGV	Unmanned Ground Vehicles
UTM	Unmanned Traffic Management system
UML	Unified Modeling Language
UWB	Ultra Wide Band
V2I	Vehicle to Infrastructure communication

V2V	Vehicle to Vehicle communication
VCA	Video Content Analysis
VLOS	Visual Line of Sight
WCET	Worst Case Execution Time
WP	Work Package

Executive Summary

The **COMP4DRONES** project aims to provide a framework of **key enabling technologies** for safe and autonomous drones. It leverages composability and modularity for developing customizable and trusted drones for civilian services. To support the **COMP4DRONES** framework development, the project also provides an **engineering methodology**. The methodology is based on **reuse** and **agility** to speed the system development and its qualification.

In this deliverable, first, we describe the **generic procedure** we follow in the project to develop a drone system. This procedure starts with the specification of the system's concept of operations (i.e., system functions, system's environment, and regulation constraining/affecting the system). Then, a number of key technologies are identified based on the concept of operations. Finally, the identified technologies with a set of guidelines are used for developing the system following the engineering methodology described in this deliverable.

Second, we present the three **drone categories** identified by EASA (i.e., open, specific, and certified). We also discuss the existing **regulation** requirements that affect the drone system development, and the specific operational risk assessment methodology (**SORA**).

Third, the key enabling technologies for drones are described (i.e., the **COMP4DRONES framework**). These technologies are categorized in four groups: drone capabilities for supporting U-space services, system functions (i.e., the core functions to enable the drone flight from one location to another), payload technologies (i.e., payloads added to the drone to perform a mission such as camera, package picker, etc.), and tools that support the system development. We also present the technologies that are being developed in the project. These technologies include generic components to support the reference architecture, components to enable safe and autonomous drone flight, and technologies that enable the trusted communication.

Fourth, to ease the development of drone systems, a number of **guidelines/recommendations** are provided. These guidelines are for (a) the development processes in general (e.g., rapid lifecycles and process for developing machine learning-based components), (b) enabling the development of safe drone by taking into account different failures (e.g., battery, controllers, and communication link failures), (c) re-use of the existing platform technologies, (d) considering the mixed-critically aspects in the system development, (e) architecture evaluation and performance optimization, (f) hardware-based security, and (g) development of specific system features (e.g., communication infrastructure and video analysis).

Fifth, the **system engineering approach** for drone system development is introduced. This approach is composition-based, where a composition structure is the main driver of the development. In the context of **COMP4DRONES**, this composition structure is the **reference architecture** (described in the deliverables D3.1/2). The different phases of the engineering approach include concept of operations' specification, selecting the technologies based on the concept of operations, system's design in relation to the reference architecture and the selected technologies, system's implementation and technologies integration, and system's validation and verification.

Finally, tools that support the different phases of the engineering approach are presented. The tools are divided into three categories: system modelling and code generation, system validation and verification, and system analysis and optimization.

Scope and positioning of deliverable

The initial version of the **COMP4DRONES** framework and engineering methodology were described in the deliverables D2.1 and D2.2. The aim of this deliverable (D2.3) is to provide a consolidated version of D2.1 and D2.2 based on the ongoing integration and evaluation efforts realized in WP1, and technological work-packages (WP3, WP4, WP5 and WP6). Later on (i.e., at M30), a final version of the project framework and methodology will be presented in the deliverable D2.4 (see Figure 1).

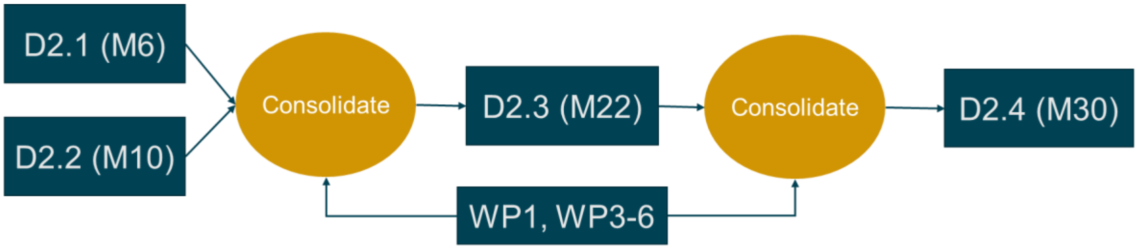


Figure 1: Relationships between WP2 deliverables

In the **deliverable D2.1**, an initial specification for the **COMP4DRONES framework** is provided. First, it introduced a set of key concepts that are needed to define the framework such as U-Space and SORA (Specific Operations Risk Assessment). Second, it discussed the current state of the drone systems which include the drone itself, the ground control station, and the communication between them. It also described some of the drone sub-systems such as navigation, positioning, autonomic management, etc. Third, a brief summary of the project demonstrators is presented and used to identify the common usages of drones which are classified into: flying stages (e.g., take-off, cruise, etc.) and mission specific operations (e.g., survey land, check crop health, inspect offshore infrastructures, etc.). Fourth, based on the common drone usages, the key enabling technologies are identified (i.e., the **COMP4DRONES** framework). These technologies include U-space capabilities (e.g., geofencing, security, telemetry, etc.), system functions (e.g., flight control, positioning, coordination, etc.), payloads (e.g., camera, LIDAR, etc.), and tools (e.g., system design, data analytics, mission planning, etc.).

In the **deliverable D2.2**, an initial **COMP4DRONES methodology** and **workflow** is provided. First, as the project was in its early stages (i.e., the more focus is on use-cases and requirements collection), methodologies for requirements collection and for measuring the project success criteria are introduced. Second, it introduced 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 the deliverable D1.1, and from the perspectives of the service providers and system integrators. Fourth, 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 up the system development and its qualification.

In this deliverable, we consolidate the deliverables D2.1 and D2.2 by providing an updated version of the project's framework and methodology (that has been proposed during the first year of the project). Thus, we start by presenting the overall project workflow and methodology in Section 0, followed by the regulations that concern the drone system development in Section 0. In Section 4, we present the key enabling technologies identified in D.2.1, and the technologies being developed in the different work packages of the project. A set of guidelines for developing safe and autonomous systems are presented in Section 0. In Section 0, an updated version of the project methodology is presented. Finally, the different tools that support the proposed methodology are provided in Section 7.

1 Introduction

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

The aim of the **COMP4DRONES** project is to provide a framework of **key enabling technologies** for safe and autonomous drones. In particular, **COMP4DRONES** leverages composability and modularity for customizable and trusted autonomous drones for civilian services. The project takes 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 also consider the SESAR-JU studies concerning civilian drones, and adhere to the U-space approach and protocols. To support the **COMP4DRONES** framework development, the project also provides an **engineering methodology**. The methodology is based on **reuse** and **agility** to speed the system development and its qualification.

The focus of this deliverable is the method and workflow that will be introduced in the section 2. However, it is also interesting to state how the workflow has not been developed “on paper”, but has been influenced by the work in the different work-packages of the project, in particular the work on use cases and key enabling technologies. This influence of the **COMP4DRONES** *project* structure is shown in Figure 2. First, the different demonstrators have been specified (i.e., scenarios, features, and functional and non-functional requirements) and their requirements have been analysed to get a unified list of requirements (WP1). Second, the unified list of requirements was used to identify the key enabling technologies that are going to be developed during the project (WP2), including for instance precision landing or geo-fencing. 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). The experience during the work helped us to define a suitable workflow which is also reflected in the tools for design, verification, and performance analysis (WP6

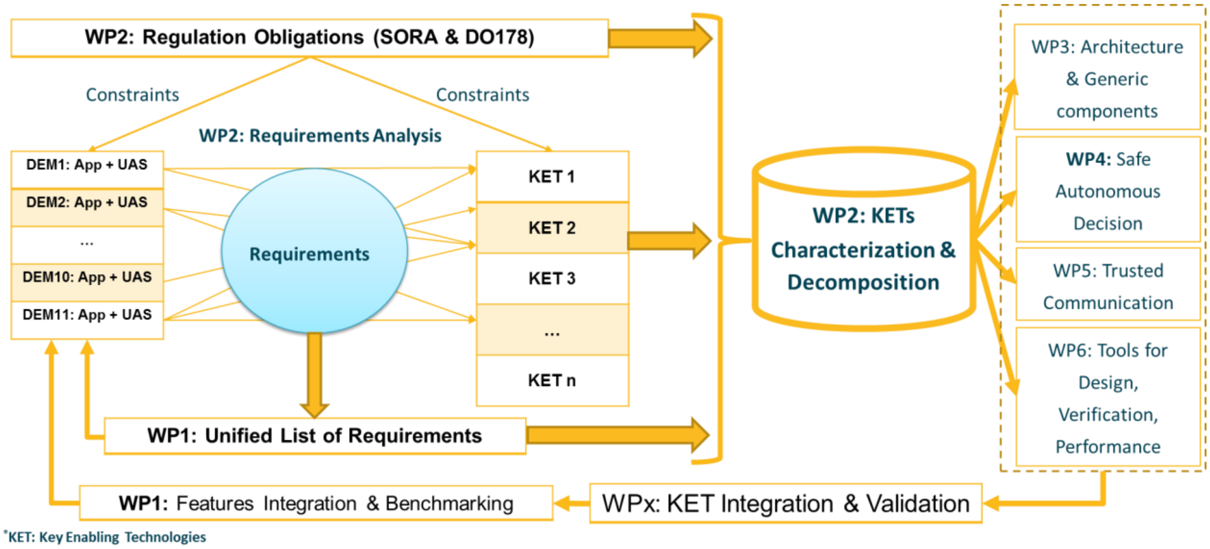

WP2: Regulation Obligations (SORA & DO178)
Constraints
Constraints
WP2: Requirements Analysis
DEM1: App + UAS
DEM2: App + UAS
...
DEM10: App + UAS
DEM11: App + UAS
Requirements
KET 1
KET 2
KET 3
...
KET n
WP2: KETs Characterization & Decomposition
WP3: Architecture & Generic components
WP4: Safe Autonomous Decision
WP5: Trusted Communication
WP6: Tools for Design, Verification, Performance
WP1: Unified List of Requirements
WP1: Features Integration & Benchmarking
WPx: KET Integration & Validation
*KET: Key Enabling Technologies

Figure 2: The overall work flow of the COMP4DRONES project

2 C4D Workflow

In this section, we describe the overall workflow of the project, and its methodology to support the development of the drone systems.

2.1 Generic Procedure for Developing UAS

When considering using 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 provider (see Figure 3). 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/her about what types of operations are allowed and what requirements must be fulfilled.

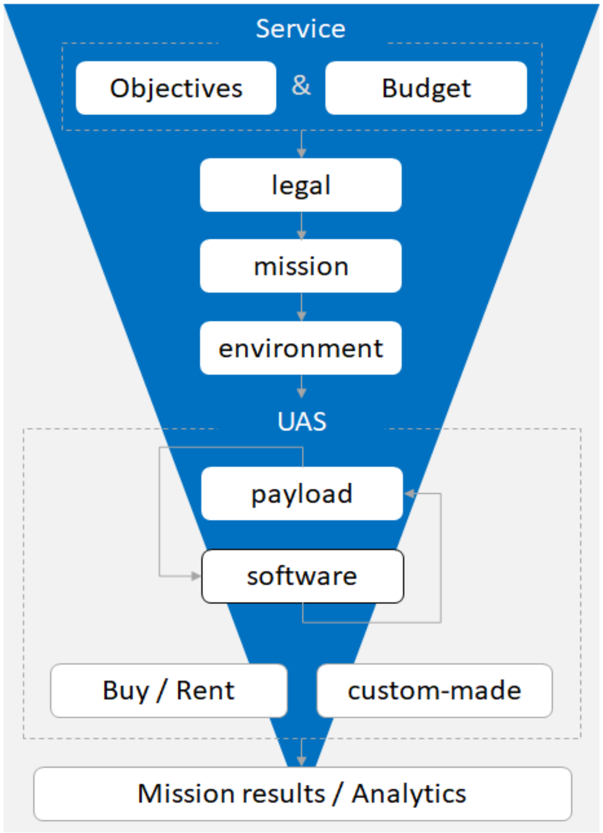


Figure 3: Step by step procedure for developing UAS

After that, a precise definition of the mission will define the UAS[1, 2]. 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 that the 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.

[1] https://www.easa.europa.eu/sites/default/files/dfu/SC-VTOL-01.pdf

[2] https://www.sesarju.eu/sites/default/files/documents/u-space/SESAR%20principles%20for%20U-space%20architecture.pdf

Finally, after listing all the specifications, the platform is defined. 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.

2.2 C4D Procedure for the Development of UAS

The system engineering design process is a common series of steps that engineers use in creating functional products. The process is highly iterative. Parts often need to be repeated many times before another can be entered. The part(s) that get iterated and the number of such cycles in any given project may vary. In the process of the system development, the following steps can be distinguished: solution orientation, design of solution, development of the solution, solution validation and the management of the value creation[3]. Below, we provide a brief overview of these steps.

Solution orientation. The objective of this step is the generation of knowledge, the reconciliation of stakeholder values and the building of a mutually agreed vision of the solution that will be proposed (i.e., legal, mission, and environment in Figure 3). Specific activities are: to understand the context and the needs of a solution leading to a shared vision of the goal and a description of the context in which the system will be used, and the development of an architecture and a development strategy.

Design the solution. The objective of this step is to formalize the requirements, finalize and agree on the specification and the chosen design that will be used to develop the solution (i.e., software and payload in Figure 3). This phase consists of the following steps: formalization of requirements leading to a functional baseline and specification, and design of a solution leading to a fully described solution.

Develop the solution. This step is aimed at lowering the level of the requirements down to a complete definition, and to consolidate definition up to the solution level (i.e., buy/rent vs. custom made step in Figure 3). It consists of a preliminary version of solution definition, which is typically reviewed in an iterative series of steps until the last step where a final solution definition is reached. Then, the solution is actually implemented, and all implementation components are integrated and verified. This involves: the evaluation of the design, behaviour interactions and performance of the solution or solution element, and to confirm by evidence that the requirements against which it has been designed are fulfilled.

Validate the solution. The solution validation (i.e., mission results/analytics step in Figure 3 is performed (a) to demonstrate with evidence that the solution or solution element fulfils its intended use when placed in its intended environment as defined in the contract (or any other formal agreement), (b) to support the certification (if required) and, (c) from the industrial point of view, to confirm that all IVV (Integration, Verification and Validation) activities are completed and that the product's data package is ready for release to production and to support.

Following the generic step by step procedure for developing UAS shown in Figure 3 and the system engineering process described above, we have proposed a procedure for developing a UAS in the **COMP4DRONES** project.

To develop a drone system, we suggest a system engineering approach (guided by the procedure shown in Figure 4). In the following, we describe the system engineering approach and how it is adapted to the UAS development. This workflow will be stabilized and documented with tool illustrations in the last iteration of the project via deliverable D2.4.

[3] Rephrasing, based on: https://en.wikipedia.org/wiki/Engineering_design_process

All steps in this procedure are supported by tools as shown in Figure 4. The different tools have access to the repository of reusable components which are classified according to the lightweight ontology developed in deliverable 3.2. An initial version of the architecture is obtained by instantiating the reference architecture template with specific components from the repository which are either from the market or developed in the context of **COMP4DRONES** project. The initial version is then *iteratively refined* based on the feedback from analysis, simulation or real experiments on a target platform. The latter are based on code generation and deployment tools. This system is then validated and verified to ensure its correctness and it meets the users' needs (i.e. solution validation step).

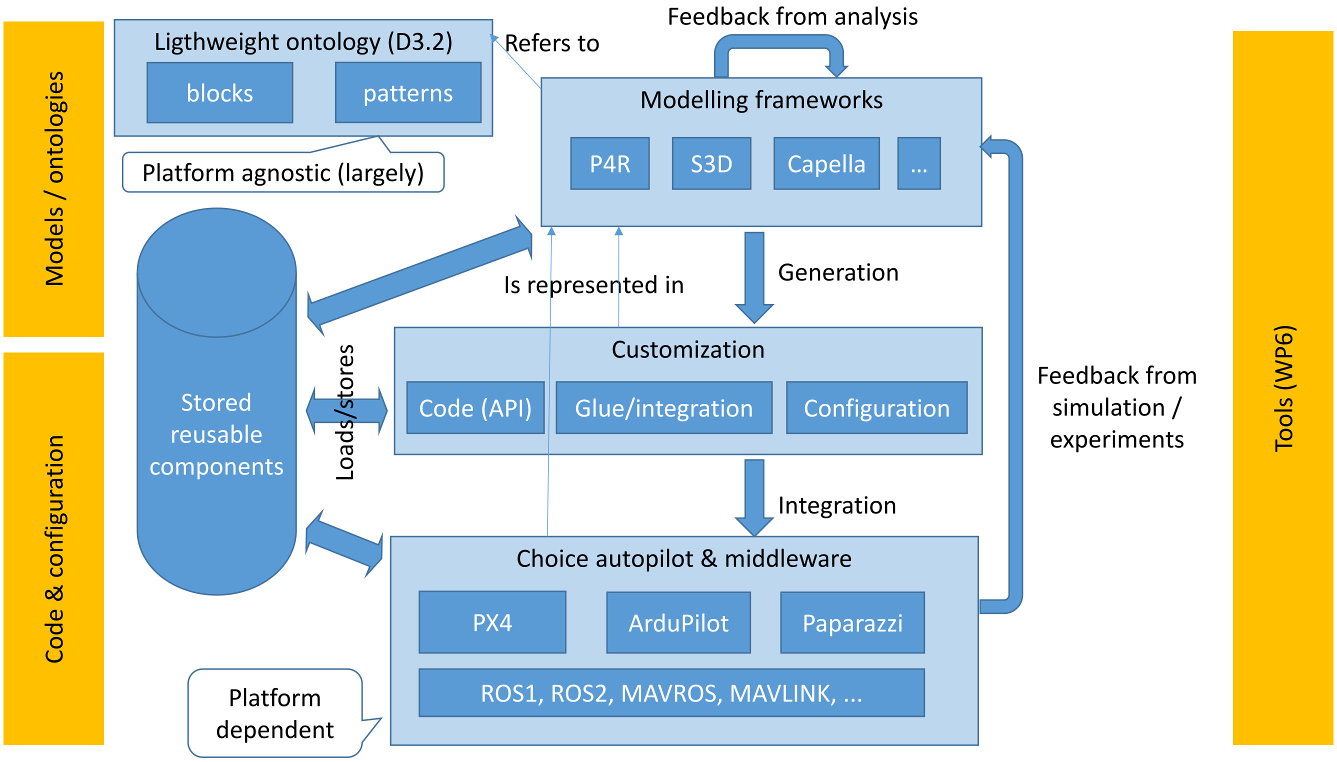


Figure 4: C4D general UAS development procedure

3 Regulations for Drone Systems

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, and Conseil Pour Le Drone Civil in France*).

In this section, we describe the three drone categories defined by EASA (i.e., open, specific, and certified), a set of regulation requirements that need to be taken into account in developing drone systems, and a step-by-step procedure to evaluate risks of a drone system under consideration (i.e., SORA methodology). These concepts help in specifying the concept of operations of a drone mission.

3.1 Drone Categories

The European Aviation Safety Agency (EASA)[4] 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 "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 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;
- 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.

[4] https://www.easa.europa.eu/domains/civil-drones-rpas/drones-regulatory-framework-background

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 5). 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).

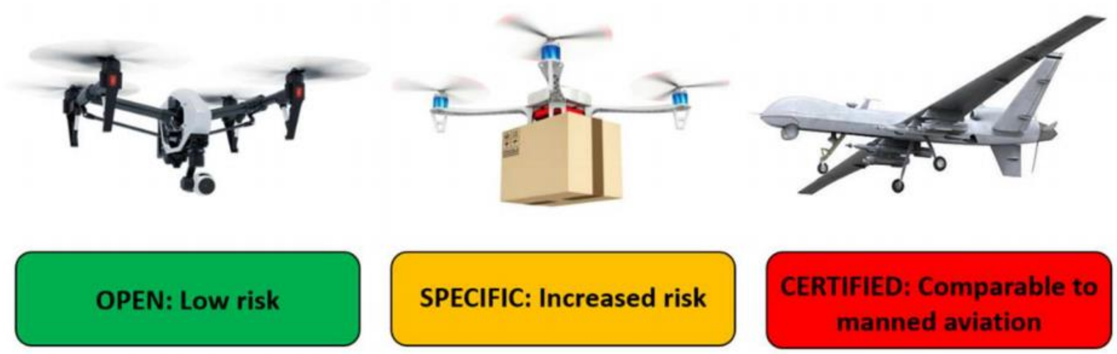


Figure 5: Drone Classification[5]

- "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;
- "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

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

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. The following are the 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 UAs it operates in order for them to be easily identifiable, when required.
- The operator shall ensure that UAs are equipped with an electronic identification means, when required.
- The operator shall ensure that UAs are equipped with a geofencing function, when required.
- The competent authorities may designate zones or airspace areas where UA operations are prohibited or restricted.

3.2 Regulations Requirements on UAS Design

The regulations' 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). This 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 be taken into consideration on top of the following requirements.

The use of "shall" and "should", 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 the U-space foundation services (U1), which are already mandatory in many member states such as 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 (e.g., tracking, surveillance data exchange, geo-fencing, technologies allowing incident/accident reporting, and 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 operates 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 vehicle 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 project[6]) has recently started by SESAR JU and EuroControl to address the Common Altitude Reference System[7] 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 type 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.
- **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.

[6] https://www.u-spaceicarus.eu

[7] https://www.eurocontrol.int/publication/uas-atm-common-altitude-reference-system-cars

- **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.

3.3 SORA: Specific Operational Risk Assessment

JARUS (Joint Authorities for Rule-making on Unmanned Systems) has developed the Specific Operational Risk Assessment (SORA)[8], 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 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.

At 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.). Finally, 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. Finally, 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 6[9].

The SORA methodology consists of ten systematic steps:

Step #1: ConOps Description

[8] JARUS, “JARUS guidelines on Specific Operations Risk Assessment (SORA),” JARUS publications, 2018.

[9] 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.

The ConOps contain all the relevant technical, operational, and system information needed to assess the risk associated with the intended operation. It includes such things as the flight path, airspace, air and ground density maps, Air Navigation Service Provider (ANSP) interface, and other information related to the intended use of the UAS.

Step #2 and Step #3: Determination of Ground Risk Class (GRC)

- Step#2: The Intrinsic Ground Risk Class (scaled from 1 to 10) is first determined, depending on the UAS weight and physical dimensions, (with indication of typical expected kinetic energy released upon ground) as well as the intended operation.
- Step#3: The Final Ground Risk Class (that may be higher or lower than the intrinsic Ground Risk Class) is determined considering design aspects which may have a significant effect on the lethality of the drone and three mitigation measures:

  1. Strategic mitigations based upon ground risk buffer and overflown population density.

  2. Mitigations intended to reduce the effect of a ground impact.

  3. An emergency response plan to address and limit the effect of an operation out of control.

Step #4 and #5: Determination of the Air Risk Class (ARC)

Both the initial and the residual risk after mitigations are applied.

- Step #4: The Initial ARC is assessed based on the airspace requested in the ConOps. The parameters that define the airspace class are: a typical (e.g., segregated) versus typical airspace, altitude, controlled by air traffic versus uncontrolled, airport environment versus non-airport, and airspace over urban versus rural environments.
- Step #5: The Residual ARC is the residual air risk after applying strategic mitigation measures. Two types of strategic mitigations measures exist in the SORA. Air risk mitigations are either operational restrictions (e.g., boundaries, time of operation) controlled by the UA operators or by structure of the airspace and the associated rules controlled by the relevant authorities. Strategic mitigations are applied before flight. Determination of ARC requires full coordination with an agreement by the ANSP for the given operation.

Step #6: Tactical Mitigation Performance Requirement (TMPR) and Robustness Levels

Tactical mitigations are applied during the conduct of the operation, and are used to mitigate any residual risk of a mid-air collision that may remain after the strategic mitigations have been applied.

Tactical Mitigation Performance Requirements (TMPR) address the functions of Detect, Decide, Command, Execute and Feedback Loop, for each Air Risk Class. These mitigations range from simple, for example relying on UTM infrastructure, to more complex TSO (Technical Standard Order) DAA equipment that addresses the risk of non-cooperative air traffic (those without transponders) and cooperative air traffic.

Step #7: SAIL determination

A SAIL (scaled from I to VI) is then determined using the proposed ConOps, and the consolidation of the final GRC and residual ARC.

Step #8: Identification of Operational Safety Objectives (OSO)

For the assigned SAIL, the operator will be required to show compliance with each of the 24 OSOs, although some may be optional for lower SAILs. Each OSO shall be met with a required Level of robustness (High, Medium or Low), depending on the SAIL. OSOs cover the following areas:

- UAS technical issue
- Deterioration of external systems
- Human error

- Adverse environmental conditions

Step # 9: Adjacent Area/Airspace Considerations

Compliance with safety requirements associated with technical containment design features required to stay within the operational volume regardless of the SAIL. This addresses the risk posed by an operational loss of control that would possibly infringe on areas adjacent to the operational volume whether they be on the ground or in the air.

Step #10: Comprehensive Safety Portfolio

A comprehensive Safety Portfolio is the SORA safety case submitted to the competent authority and the ANSP prior to final authorization. The Safety Portfolio contains the following information:

- Mitigations used to modify the intrinsic GRC
- Strategic mitigations for the Initial ARC
- Tactical mitigations for the Residual ARC
- Adjacent Area/Airspace Considerations
- Operational Safety Objectives

If compliance with the required safety objectives is not achieved for the given SAIL, additional mitigation measures may be needed to further reduce the GRC or/and ARC or a change to the operational volume and ConOps may be required.

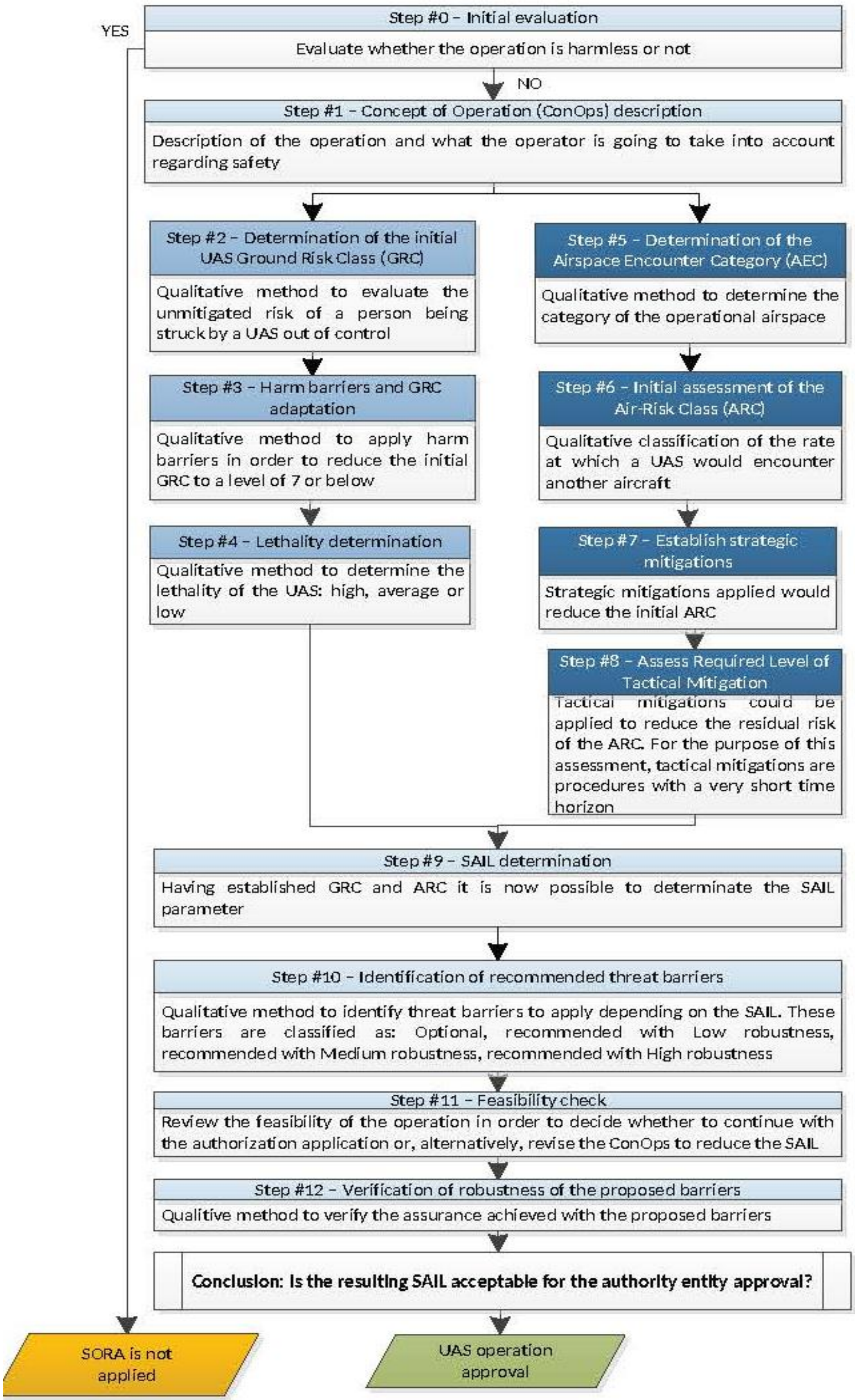

Step #0 – Initial evaluation
Evaluate whether the operation is harmless or not
YES
NO
Step #1 – Concept of Operation (ConOps) description
Description of the operation and what the operator is going to take into account regarding safety
Step #2 – Determination of the initial UAS Ground Risk Class (GRC)
Qualitative method to evaluate the unmitigated risk of a person being struck by a UAS out of control
Step #3 – Harm barriers and GRC adaptation
Qualitative method to apply harm barriers in order to reduce the initial GRC to a level of 7 or below
Step #4 – Lethality determination
Qualitative method to determine the lethality of the UAS: high, average or low
Step #5 – Determination of the Airspace Encounter Category (AEC)
Qualitative method to determine the category of the operational airspace
Step #6 – Initial assessment of the Air-Risk Class (ARC)
Qualitative classification of the rate at which a UAS would encounter another aircraft
Step #7 – Establish strategic mitigations
Strategic mitigations applied would reduce the initial ARC
Step #8 – Assess Required Level of Tactical Mitigation
Tactical mitigations could be applied to reduce the residual risk of the ARC. For the purpose of this assessment, tactical mitigations are procedures with a very short time horizon
Step #9 – SAIL determination
Having established GRC and ARC it is now possible to determinate the SAIL parameter
Step #10 – Identification of recommended threat barriers
Qualitative method to identify threat barriers to apply depending on the SAIL. These barriers are classified as: Optional, recommended with Low robustness, recommended with Medium robustness, recommended with High robustness
Step #11 – Feasibility check
Review the feasibility of the operation in order to decide whether to continue with the authorization application or, alternatively, revise the ConOps to reduce the SAIL
Step #12 – Verification of robustness of the proposed barriers
Qualitive method to verify the assurance achieved with the proposed barriers
Conclusion: Is the resulting SAIL acceptable for the authority entity approval?
SORA is not applied
UAS operation approval

Figure 6: The SORA process

4 Key Enabling Technologies

After the definition of the concept of operations of a drone system (while taking into account the existing regulations as discussed in Section 0), the different technologies that enable the system development need to be identified. Thus, in this section, we describe the key enabling technologies in general (i.e., the **COMP4DRONEs** framework - more details are in D2.1), and the technologies being developed in the context of **COMP4DRONES** (more details can be found in the different work packages WP3-5).

4.1 Drone Key Enabling Technologies

In the following, we give a brief description of the **different components/technologies** that are required to have a fully functioning drone system. These components are divided into four groups: u-space capabilities, system functions, payload, and tools. The system functions are the common/shared elements that can be reused across different drone systems. The payloads are specific to certain application/mission, but also can be used in more than one application.

4.1.1 Drone Capabilities for U-space

The capabilities expected for enabling U-space services are shown in Figure 7. These capabilities are divided into three groups to support different type of services: foundation, initial, and advanced. First, capabilities for the foundation services include: geofencing, security, telemetry, operation management, e-identification, communication, command and control, surveillance, and navigation. Second, the initial services capabilities include tracking and emergency recovery. Third, the capabilities for the advanced services are: detect and avoid, vehicle-to-infrastructure communication, and vehicle-to-vehicle communication (see Figure 7).

4.1.1.1 U1: U-space Foundation Services

- *E-Identification* is the ability for identifying the drone and its operator in the U-space.
- *Geofencing* is the drone ability to be compliant with time, geographical, and altitude restrictions defined by the geo-fencing service.
- *Security* is making the drone able to protect itself and its data (i.e., interaction with infrastructure and other vehicles) from attacks.
- *Telemetry* is the ability for transmitting measurement data from a drone to another drone or to a service provider for meeting the demands of relevant services.
- *Communication, navigation, and surveillance* is making the drone able to meet performance requirements of the communication, navigation and surveillance in the specific environment (in which it will operate). This capability consists of on-board sensors and equipment (e.g., voice radio relay, data link, etc.) as means to achieve the required performance.
- *Command and control* are a drone’s ability to communicate with a ground control station for conducting the flight (normally through a specific data link).
- *Operations management* is the ability for planning and managing the drone missions. This involves accessing and using of all relevant information for planning, notifying, and operating a mission.

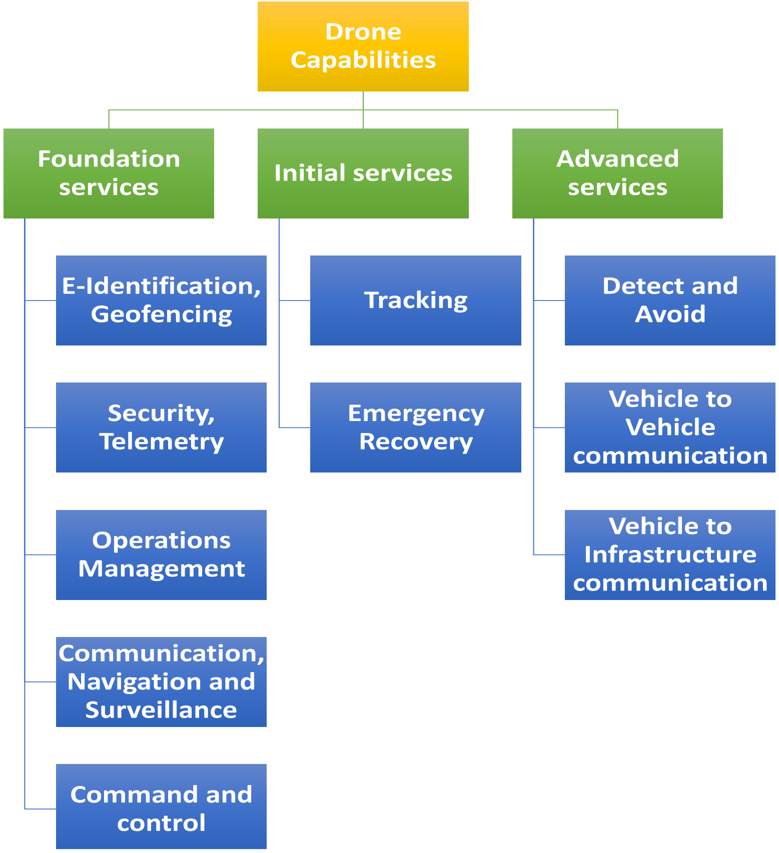


Figure 7: Drone capabilities for U-space

4.1.1.2 U2: U-space Initial Services

- *Tracking* is the drone's ability to provide flight parameters that include at least its position and altitude.
- *Emergency recovery* is the ability of drones to take into account failure modes such as link failure, command and control (C2) failure. It also takes measures for ensuring the safety of the vehicle itself, other vehicles, and property and people on ground.

4.1.1.3 U3: U-space Advanced Services

- *Vehicle to infrastructure communication (V2I)* is the drones' ability for sharing information with infrastructure components.
- *Vehicle to vehicle communication (V2V)* is making drones able to communicate information to each other. The nature of the exchanged information and its performance depend on the application.
- *Detect and avoid* is ability of drones to detect hazards, or cooperative and non-cooperative conflicting traffic, and to take the appropriate actions to comply with the applicable rules of flight.

4.1.2 System Functions

The drone system functions are the core functions required for the drone to perform its flying stages in safe and efficient manner. The different system functions are shown in Figure 5 and are described in the following sub-sections.

4.1.2.1 Flight Control

- *Intelligent Mission Management (IMM)* is onboard and/or ground technologies for providing a desired mixture of human-directed and autonomous drone operations. It enables the shift of the human role in conducting drone missions from vehicle operators to be users and requesters for drone applications. It increases the autonomy level in performing complex UAV operations.
- *Intelligent Outer-Loop Control (IOLC)* is an on-board capability to enable autonomous and semi-autonomous operations. Traditional control systems such as autopilots or flight management systems (FMS) achieve navigation and guidance goals defined by human through controlling the vehicle flight surfaces. IOLC achieves high-level mission goals through extending the traditional approach. For example, a flight management system can be tasked to make the aircraft follows a specified route, and an IOLC can be tasked with a broader goal such as monitoring a set of ground targets for events of interest to alert users whenever such events occur. For meeting such goals, the system should be able to control not only vehicle surfaces, but also its communications, sensor payload, and other sub-systems.

4.1.2.2 Flight Navigation

- *Flight planning and scheduling* are general technologies that take higher-level goals, constraints, and objectives and then turns these into detailed plans and schedules. It can be performed by humans or drones. The difference between planning and scheduling is that: (a) planning includes more choices about what objectives to achieve, and the different actions needed to achieve them; (b) scheduling involves activities that are given, and the main decisions is to order these activities, and assign resources to them. Both scheduling and planning are cross cutting technologies and have a wide application in many areas of intelligent systems.
- *Fail-safe Mission* is the ability of a UAV system to adapt to software or hardware failures for having an acceptable level of safety. This function is among the most critical functions in drones. Such technology is generic to any drone application for providing high reliability and it is one of the important features needed to access the air space. Reports regarding drones indicated that they are looking for “reliability comparable to a piloted aircraft”.
- *Contingency Management* is an on-board capability for reacting to unforeseen events. It is particularly needed to minimize the likelihood of property damage and human casualties. It also maximizes the likelihood of drone and payload survival. In general, it includes a number of techniques that are designed to increase robustness of a drone in response to uncertainties. These uncertainties have many forms such as a failure or a degradation of hardware components (actuators, sensors, etc.), missing precise information about the drone environment (wind, visibility, cloud cover, etc.), or unknown events (volcanic eruptions, fires, etc.). In the face of these uncertainties, techniques for contingency management are useful to improve mission safety and productivity.
- *Deconfliction* is a function that is used to resolve potential conflicts that can occur between drones’ trajectories in the phase of planning strategic trajectories. This function aims to reduce the workload of the air traffic controller in conflict resolution through designing efficient trajectories with minimal number of potential conflicts. Thus, once the drone(s) is(are) cleared for flying, the workload of controller will be more monitoring with less conflict prediction and resolution (i.e., more flights can be accommodated by the controller at a given time).
- *Detect and Avoid (DAA)* is a basic requirement for drones to safely operate. The collision avoidance process involves capturing the surrounding environment, and assessing

potential of colliding with hazards that are detected. It takes corrective actions to avoid the hazards when a collision is upcoming. The hazards that are of concern in collision avoidance are: ground (earth surface), drones (other vehicles in the space), weather, and obstacles (e.g., power lines, towers, ground equipment, etc.). The pilot's eyes can be used to visually detect and track the hazards. However, when the operator of the drone is remotely located, an automatic DAA system is foreseen as an important feature to allow the drone operation with an equivalent level of safety of a piloted drone.

4.1.2.3 Positioning

- An *indoor positioning* system (IPS) is a set of devices used for locating people or objects when GPS and satellite technologies fail or lack precision. This usually occurs in multistory buildings and underground locations. A set of techniques and devices can be used for providing indoor positioning ranging from (a) reconfigured devices such as Bluetooth antennas and Wi-Fi, smart phones, digital cameras, and clocks to (b) purpose-built installations with beacons and relays placed throughout a defined space. IPS has many applications in military, commercial, and inventory tracking industries.
- *Geofencing* is a virtual barrier that can be created by combining GPS network and LRFID (Local Radio Frequency Identifier) connections (e.g., Wi-Fi, Bluetooth, etc.). This boundary is forced by the drone during its flight. Such technology is available since many years with an early adaption to monitor cattle with the help of GPS for providing alerts when livestock left its predefined boundaries. There is also other use such as monitoring of fleet vehicles to provide early warning when anything abnormal occurs.
- *Georeferencing* is the task of assigning real-world coordinates to the pixels of a raster. Such coordinates are obtained through performing field surveys (i.e., collect coordinates by a GPS device for easily identifiable features in the map or image). In some situation, when looking for digitizing scanned maps, the coordinates can be obtained from the markings on the map image itself. Using such sample coordinates, the image can be warped and fitted within a chosen coordinate system.
- *Simultaneous Localization and Mapping (SLAM)* is the process of recording environment and location awareness in a map of an autonomous vehicle. SLAM is an important component in self-driving cars and other autonomous robots to enabling awareness of their location and best routes to their destination. Through the creation of its own maps, SLAM provides a quicker, a more autonomous and an adaptable response than the pre-defined routes.

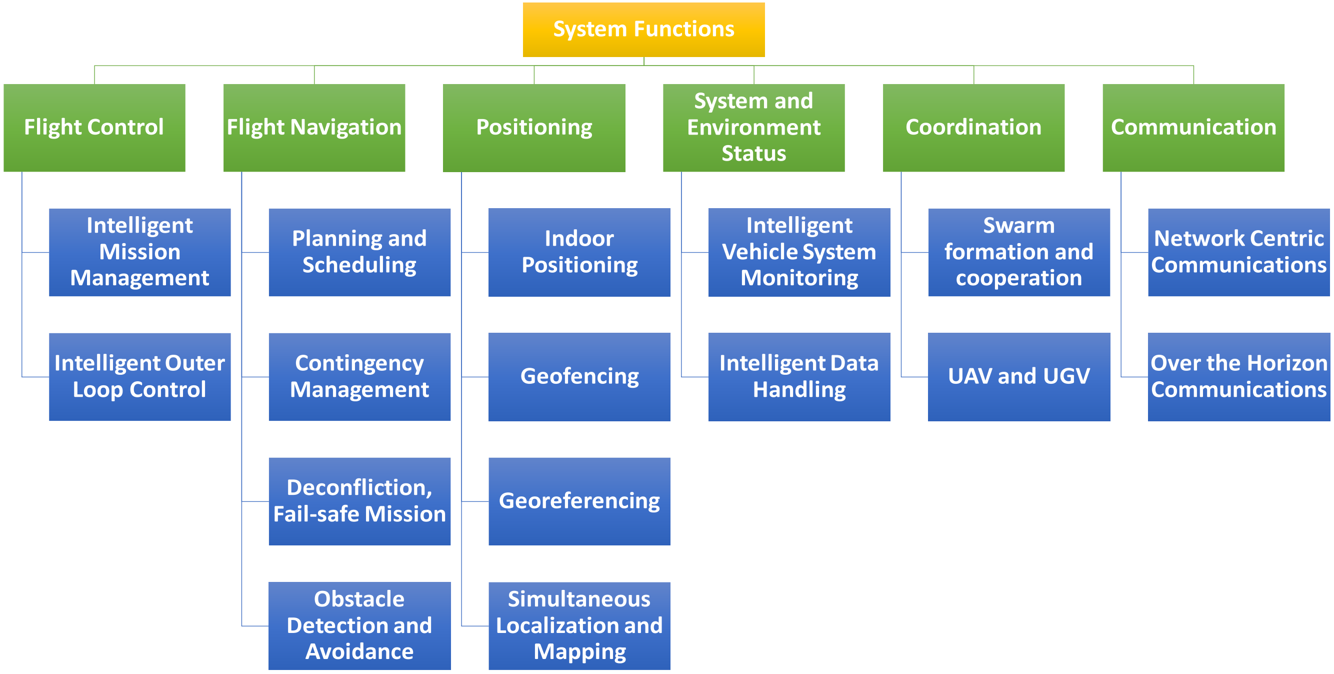

System Functions
Flight Control
Flight Navigation
Positioning
System and Environment Status
Coordination
Communication
Intelligent Mission Management
Intelligent Outer Loop Control
Planning and Scheduling
Contingency Management
Deconfliction, Fail-safe Mission
Obstacle Detection and Avoidance
Indoor Positioning
Geofencing
Georeferencing
Simultaneous Localization and Mapping
Intelligent Vehicle System Monitoring
Intelligent Data Handling
Swarm formation and cooperation
UAV and UGV
Network Centric Communications
Over the Horizon Communications

Figure 8: Drone system functions

4.1.2.4 System and Environment Status

- *Data fusion* is the task of integrating many data sources to produce consistent, useful, and accurate information that cannot be provided by any individual source of data. The drone telemetry system has a limited bandwidth. This bandwidth must be allocated between the flight control function and the payload elements. Thus, it is not possible to transmit all data out of drone, and then data processing is needed for reducing the volume of transmitted data. This enables real-time data analyses, as well as ensuring some level of data capture in the event the loss of the platform.
- *Intelligent System Health Management (ISHM)* is a technology designed for assessing a system's health and recommending/performing actions that ensure it will remain healthy in future. This technology contributes to drone safe operations in several ways such as recovering from faults, and recommending actions in the presence of other faults. ISHM techniques can be either on the drone or on the ground.

4.1.2.5 Coordination

- The necessity of *coordination* between unmanned aerial vehicles (UAV) and unmanned ground vehicles (UGV) is particularly evident to do missions in remote areas, where human may be exposed to dangerous situations. In these situations, monitoring and exploration missions could be performed safely by robots where they can easily gather information from the environment safely. Multiple UAVs and UGVs are certainly able to play preprogramed missions by moving around in certain scenarios. However, the most interesting challenge is providing them with a decisional autonomy and opportunity for cooperation and adaptation according to real-time situations and with little human intervention.
- *Swarm Formation and Cooperation* is the reasoning and making decision entity that is responsible for the use of mission requirements, observations (by the UAV itself and other UAVs in the fleet), and system constraints to have a specific organization of the UAVs. In brief, it needs to compute trajectories of the different UAVs and make decisions on how tasks are allocated for achieving a good team behaviour. Coordination means

achieving and sustaining good formations or task distribution between drones in a self-organizing manner. The coordination can be done at a global or a local level depending on the mission specification and drones' capabilities.

4.1.2.6 Communication

- *Net-centric communications* is a concept of operation that uses advanced technology for shifting to a data-centric paradigm from an application-centric paradigm. It allows the users to access applications and services by web services. This concept increases situational awareness and robustness of missions via networking sensors, decision making, and faster command and control.
- *Over the Horizon Communications (OTH)*, which is commonly referred to BVLOS (Beyond Visual Line of Sight) communications, is a basic function that is required for UAVs to operate in the global airspace. OTH is needed for Command and Control (C2Link), status and health of the vehicle, situational awareness, and real (near real) time vehicle position (longitude, latitude, and elevation above the surface of the earth at a given time) using the GPS or the UAV's on-board navigation system. There is also a need to have OTH with UAV payload to receive real-time data, snap shots, or determine status of on-board data recorders.

4.1.3 Payload Technologies

The drone system includes a set of payload technologies to support the drone mission specific operations. The payload technologies include optical sensors, microwave sensors, in-situ sensors, and external sensors as shown in Figure 9.

4.1.3.1 Optical Sensors

- *Active optical* remote sensors (i.e., lidar) use optical source such as a laser for sensing targets. The targets can be either hard objects (e.g., other vehicles, terrain, and obstacles) or the atmosphere through scattering light from molecules and aerosols. Measurements of hard target are useful for geographical information systems, and for payload delivery. On the other hand, atmospheric parameters can be measured such as gas concentration, aerosol density, wind, and cloud cover. The advantage of lidar-based approaches is that spatial and temporal resolution is much higher than the other sensory systems.
- *Passive optical* sensors are the major imaging devices that are found on aircraft and satellites. The devices essentially capture infrared radiation emissions, and direct or reflected solar energy. Then, they project them to photosensitive detectors through an imaging optics system. However, there also non-imaging sensors that are used to collect radiometric and/or spectral data from a single point. Such sensors are typically used for measuring radiations from the earth and the sun, and are used to characterize the intervening atmosphere. Both types of the sensors are appropriate within UAVs systems.

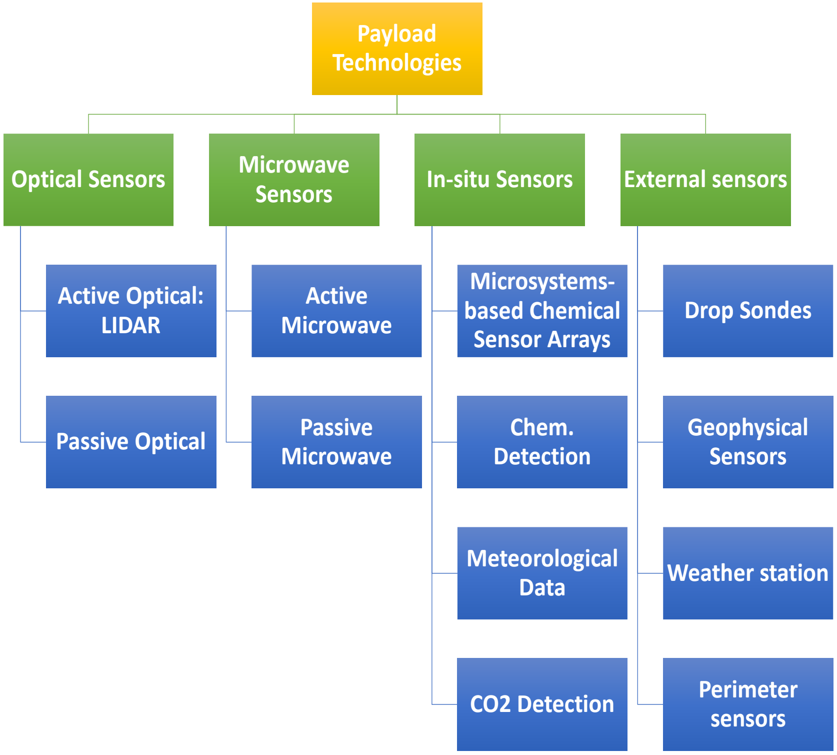


Figure 9: Payload Technologies

4.1.3.2 Microwave Sensors

- *Active Microwave* sensor is an imaging radar. It emits microwave radiation. The sensor then records the echoes returned from the scene to be observed. This system contains wavelengths that varies from less than a centimetre to three meters that depend on the application. To achieve a good resolution, the system transmits a chirp waveform with a bandwidth that depends on the desired resolution, then many pulses are collected and combined by signal processing approaches to achieve the desired resolution.
- *Passive Microwave* sensors are used for both surface imaging and atmospheric measurements. They gather data through detecting light, vibrations, radiation, etc. A main challenge for such sensors is the required spatial resolution. For example, Cold Land Processes may need spatial resolutions of a hundred meter at microwave frequencies, while spatial resolution of one km is needed for Soil Moisture measurements which is very challenging.

4.1.3.3 In-situ Sensors

- *Chemical Sensor Arrays* technology allows measurements for a range of chemical species in the UAV’s surrounding environment. Such type of micro-sensors is small and consuming less power in comparison with the standard instrumentation. Thus, they can be easily integrated with drone hardware and software systems.
- *Meteorological Data* such as air density, temperature, and wind affecting UAV operations need to be measured. For example, both pressure and temperature need to be measured for determining the air speed and its accuracy.
- *Difference frequency generation (DFG) lasers* are used as advanced in-situ detectors for tracing gases and their composition. Also, they can be combined with enhanced absorption spectroscopy. DFG-based sensors that allow measurements of gas traces

are small and non-cryogenic. Such measurements are an important element of meteorological or atmospheric research missions.

- The *CO_2 Detection* sensor measures CO_2 using a quantum cascade laser spectrometry in the flight configuration. The CO_2 detector uses a laser spectrometer in a flight configuration for measuring CO_2 with long term precision of 0.05 ppm and with 0.1 ppm as an absolute accuracy (i.e., matching the requirement for most of atmospheric-based drone mission).

4.1.3.4 External Sensors

- *Dropsonde* is a weather reconnaissance device that is designed to be dropped from an aircraft. There are four basic measurements that are performed by the dropsonde: temperature, pressure, winds, and humidity. Measurement of temperature, humidity, and pressure are generally performed using a thin-film polymer and thermistor package, while winds are typically measured by GPS receivers (either true or codeless GPS).
- *Seismic (geophysical) sensors* are deployed on the ground to record soundwave velocities coming from an activated seismic source (like vibrator truck). Number and spatial sampling of these sensors will drive the quality and fidelity of the final image. In general, seismic sensors are cabled-based system that requires heavy ground logistic to put them in place. With the technology advances, seismic sensors can now be delivered from air using drone swarms.
- A *weather station* is very useful to monitor the changes in real time. It allows the UAV system to detect and act when important changes in the forecast happen. For example, the drone could be ordered to finish its task and return without any further take-off, or if the change comes quickly, it can be ordered to urgently return to the base.
- *Perimeter sensors* use light-detecting, passive infrared (PIR), and infrared (IR) sensors to monitor surroundings. When a movement is detected by the sensors, notifications are sent to the security system. Such sensors can be hidden among plants or placed in the open area to create an undetectable, invisible protective barrier.

4.1.4 Tools

To support the development of drone systems, a number of tools need to be developed. These tools are divided into two main groups: tools for service specification, and tools for system development (see Figure 10).

4.1.4.1 Service Specification

- *User Requirements* referred to as needs. They specify what the user wants from the system (i.e., what activities the system enables the users to do). These requirements are documented generally in a user requirement document as narrative text. The requirements are signed by the user. Then, they are used as the main input to create the system requirements.
- *Acceptance Testing* is a level of testing, where a system is tested for user acceptability. The purpose of this type of test is to evaluate compliance of the system with different business requirements and evaluate whether it can be delivered or not. It is also considered as a formal testing for the achievement of user requirements and needs. To determine whether a system satisfies acceptance criteria or not, a business processes is conducted. These processes enable the user/customers whether to accept the system or not.

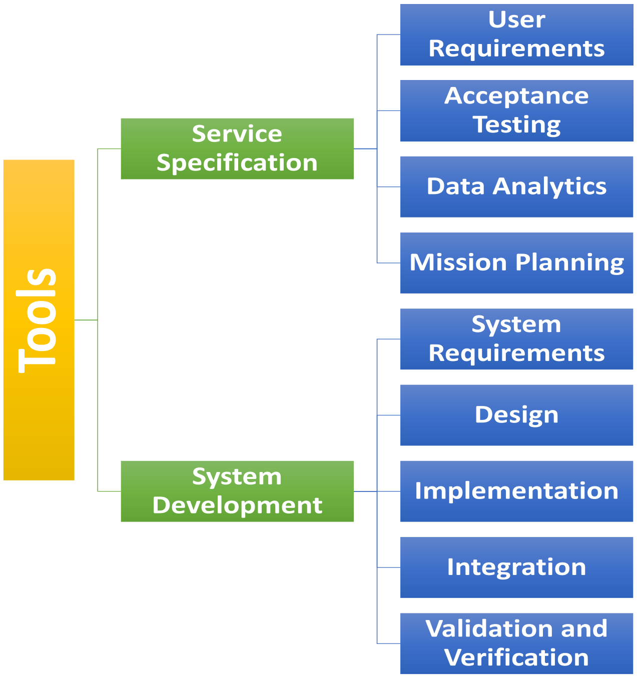


Figure 10: Tools for drone systems

- *Data Analytics* is a process of cleaning, inspecting, transforming, and modeling data with the aim to discover useful information and conclusions to support decision-making. Data analytics has many facets and diverse techniques, which used in different science and business domains. It plays an important role to make decisions more scientific and to help businesses to operate in an efficient way. Example data analytics technique is data mining that focuses on knowledge discovery and statistical modelling for predictive purposes. Another technique is the business intelligence that relies on aggregation with a main focus on business information.
- *Mission Planning* is the process to produce a flight plan that describes a proposed drone flight. It has two safety-critical aspects: compliance with air traffic requirements to avoid collisions, and fuel calculation to ensure that the drone can safely reach the desired destination. It also minimizes flight cost by choosing route, speed, and height that minimizes the necessary fuel. The produced flight plan is used by the Air Traffic Services for aircraft tracking, and finding a lost aircraft in search and rescue scenario.

4.1.4.2 HW/SW System Development Cycle

- *System requirements* are the main blocks that developers use for building the system. These requirements are statements that explain what the system should do. They are classified as either functional (i.e., specify something that is required by the users to perform their tasks) or non-functional (i.e., certain system qualities) requirements.
- *System Design* process provides the sufficient detailed information about the system and its sub-system to enable the system implementation. The result of the design process is views and models of the system architecture.
- *System Implementation* follows the structure created during the system design, and the system analysis results to construct system components. These components must meet stakeholders and system requirements specified in the early phases of the development life cycle. Implementation phase yields the lowest-level elements of the system. The

elements are bought, made, or reused. This phase also involves the processes for hardware fabrication and software realization.

- *System Integration* is the process to bring together the different component/sub-systems into one system. It also ensures that the sub-systems (i.e., different software and hardware components) work together as a system. The integrated sub-systems may include computer networks, business processes, and enterprise applications.
- *Verification and Validation (V&V)* is checking that a system meets its specifications and fulfils its intended purpose (goals). It also refers to the software quality control and software testers are responsible for this task. Simply, software validation is "does our software meet its intended goals", while software verification is "does the right software has been built".

4.2 C4D Enabling Technologies

In the context of the **COMP4DRONES** project, a number of technologies are being developed. These technologies are: generic components to support the reference architecture, components to enable safe and autonomous drone flight, and enabling technologies for trusted communication. In the following sub-sections, we describe these components/technologies briefly, and more details can be found in the technical work packages of the project (i.e., WP3-5).

4.2.1 Generic Components Supporting the Reference Architecture

A number of generic components are being developed in WP3. These components/technologies are grouped into five categories: hardware platforms, basic software, sensing, image/video processing, and trusted communication. In this section, we describe these groups in brief and more details can be found in the deliverables D3.1/2 ("Specification of Integrated and Modular Architecture for Drones").

4.2.1.1 Hardware Platforms

To support the execution of software components of drone systems, a number of hardware components are being developed in the **COMP4DRONES** project. Such hardware components are either to speed up the execution of system functions in general, or they are developed for specific system feature. First, the demand for onboard computational power of modern drones is exponentially growing due to the ever-increasing request for autonomous operation. To satisfy this request, more powerful (in terms of operational throughput) computing platforms must be embedded in the drones, while at the same time maintaining operational constraints related to the power envelope, and the interoperability and connectivity with standard drone software stacks. Thus, in the project, a number of components are developed such "Onboard Programmable and Reconfigurable Compute Platform Design Methodology", "Efficient Digital Implementation of Controller on FPGAs", and "Modular SoC-based Embedded Reference Architecture" as shown in Table 1. These components aim to speed up the execution of different functions of the drone system.

Second, SLAM (Simultaneous Localization and Mapping) techniques are considered to be a mature field. But the actual problem is the lack of a modular architecture (involving hardware and firmware configurations) that can deal with heterogeneous sensors' configurations and can provide an easy-to-use and easy-to-adapt, extensible and reusable base configuration package. Thus, "Highly Embedded Customizable Platform for SLAM technique" (described in Table 1) is being developed in the project to solve this problem. Finally, most of drone missions require image and video processing that is heavy based on the mission specification. Thus, "HW/SW System on Module for Object Detection and Positioning" is being developed in the project to enable faster execution of objects detection and their positioning (see Table 1).

Table 1: Hardware components to speed up the execution of different system functions

Component Name	*Component Description*
Onboard Programmable and Reconfigurable Compute Platform Design Methodology	This methodology is intended to enable the integration with the legacy software components, the programming interfaces, and the management of many heterogeneous components. It combines 1) an Open-Source FPGA overlay that enables plug-and-play deployment of Hardware Processing Elements (HWPE) in typical drone workloads, and 2) a methodology (MDC) to design Coarse-Grained Reconfigurable Co-processing Units to be used when defining and integrating those HWPEs into the overlay compute clusters.
Efficient Digital Implementation of Controller on FPGAs	The proposed methodology for the efficient implementation of controllers on FPGA is focused on re-timing and pipelining. The former is a transformation technique used to change the locations of the delay elements in a circuit without affecting the input/output characteristics of the circuit.
Modular SoC-based Embedded Reference Architecture	This component provides a potential solution for the limited drone's onboard computational capacity. It aims to utilize the modern heterogeneous systems that incorporate different computational paradigms (MPU, FPGA, and GPU), as they promise the advantages of better computational capabilities and power efficiency without introducing additional system complexity.
Highly Embedded Customizable Platform for SLAM technique	This platform aims to support a SLAM component through the deployment of a range of sensors to collect odometry and geo-magnetic field measurements, and assessing the distance from fixed points within the map.
HW/SW System on Module for Object Detection and Positioning	This component targets a heterogeneous-computing architecture, which includes general-purpose CPUs, DPSs, real-time capable CPUs, digital programmable logic and specialized cores for video encoding/decoding and communication means. It allows the implementation of complex algorithms and the integration of high date-rate sensors.

4.2.1.2 Basic Software

To enable the execution of a drone mission, a number of basic software components are needed. In the context of **COMP4DRONES**, generic components for mission control and power management are being developed. First, to enable safe autonomous operations in uncertain environment with unknown dynamic objects. The "Control Components that Implement Potential Barriers" supports navigation and mission planning by providing geo-awareness and avoiding any geo-fence violation (see Table 2). Second, many applications require multiple drones to work together in a cooperative manner in order to complete complex tasks quickly and efficiently. Thus, the "Multi-agent Swarm Control" and "Generic Mission Controller" described in Table 2 support fleets formation and management to enable such cooperative behaviour. Finally, power supply is an important aspect to make sure the drone flight is safe. Thus, the "Complex System for Autonomous Drone Battery Management" and "Smart and Predictive Energy Management System" are proposed to manage the battery and to select the trajectory with the lowest power consumption (see Table 2).

Table 2: Basic software of drone system

Component Name	**Component Description**
Control Components that Implement Potential Barriers	This component is used to create the required potential barriers. It produces a potential field in which the drone acts as a point. If the distance between the drone and the restricted region becomes less than a certain threshold, it generates a repulsive force to fly away the drone and does not cross geo-fence boundary or collide with any other object.
Multi-agent Swarm Control	Multi-agent swarm control component is aimed to provide the functionality of multi-drone consensus and formation tracking in distributed manner, while taking into account various constraints which are associated with the practical scenarios.
Generic Mission Controller	The controller allows fleets of UAVs to perform variety of missions. Its genericity allows to use heterogeneous UAVs and to allocate them different missions. For instance, a fleet of 10 UAVs performs logistic operations by dropping sensors

	over a huge area collaboratively, or a fleet of 4 UAVs swipe scanned areas to build an orthophoto. This controller requires an updated status of the mission. Thus, a Knowledge Base (KB) is used to serve the purpose of sharing the information through a reliable link inside the fleet.
Complex System for Autonomous Drone Battery Management	The Droneport acts autonomously. It monitors the batteries state, controls charging and battery exchange process, and provides necessary navigation information for drone landing and broadcast status of remaining batteries.
Energy Management System	Energy management system main purpose is to identify excellent trajectories from an energetic point of view, knowing the initial and final point of the mission.

4.2.1.3 Sensing

To enable a mission execution, the drone should be equipped with a number of sensors to perceive its status and environment. Some of the sensing technologies supported by the **COMP4DRONES** project are presented in this section.

Table 3: Sensing technologies

Component Name	Component Description
Hyperspectral (HSI) Cameras	The HSI camera captures hyperspectral images, tags the images together with GPS coordinates, stores the images locally, processes the data to provide certain analytics, and sends the raw and classified/processed images to be further transmitted to ground controller.
Ultra-Wideband-based Indoor Positioning	The Ultra-Wideband based Indoor Positioning System/solution (IPS) enables the provision of real-time trustable 3D position and attitude of a drone in a long indoor infrastructure (tunnel) under construction.
Outdoor Position and Attitude Estimation	The outdoor geo-referenced position and attitude estimation system (GLAD+) enables the provision of real-time trustable position and attitude to a drone in an outdoor scenario. GLAD+ provides an optimized cost-performance (accuracy, precision, continuity, and integrity) solution by relying on state-of-the-art, low cost GNSS receivers and low-cost complementary sensing devices (IMU, barometer).
Simultaneous Localization and Mapping Algorithms	SLAM component provides positioning capabilities without relying on the GPS signal. Instead, the component relies on inertial measurements, geo-magnetic earth field, and the distances from fixed points within the map to estimate the position of the drone.

First, to capture high quality images, “Hyperspectral (HSI) Cameras” are used to capture, process, and store images in a good quality. Second, knowing the drone position is very critical for a mission execution and the position should be available during the whole flight in different environment (i.e., in both indoor and outdoor environments). Thus, the “Ultra-Wideband-based Indoor Positioning” and “Simultaneous Localization and Mapping Algorithms” developed in the project are used for localizing the drone in indoor situations, while the “Outdoor Position and Attitude Estimation” provides accurate position of the drone in outdoors (see Table 3).

4.2.1.4 Image/Video Processing and Analytics

Most of drone missions require data capturing, processing, and analysis. Thus, in the context of the project, a number of components are being developed to support image and video processing. Such processing is drone through conventional neural network (CNN) for object detection (Computer Vision Component for Drones”), high-dynamic-range imaging (HDR) tone mapping (“Video Data Processing Algorithms”), hyperspectral imaging (HSI) pipeline (“Hyperspectral Imaging (HSI) Processing Pipeline”), artificial intelligence algorithms (“AI Drone System Modules”), and video content analysis algorithms (“Video and Data Analysis Algorithms”) as described in Table 4.

Table 4: Video processing and analytics

Component Name	Component Description
Computer Vision Component for Drones	This is a post-processing computer vision system based on previously-trained CNN algorithms. It enables the auto-detection and geo-referencing of different objects from RGB images captured by the UAV's on-board camera.
Video Data Processing Algorithms	It covers all functionality: reading data from a camera sensor, merging multiple images with alternating exposures into HDR images/HDR video, and applying HDR tone mapping.
Hyperspectral Imaging (HSI) Processing Pipeline	The hyperspectral imaging (HSI) pipeline is a system that processes and analyses the hyperspectral data originating from the hyperspectral payload.
AI Drone System Modules	This component uses AI algorithms with camera imaging system to detect and identify parasite animals and to classify leaf diseases.
Video and Data Analysis Algorithms	This component is a software module that implements Video Content Analysis (VCA) algorithms. The algorithms are based on Deep Learning methodologies and their goal is to process different type of images (e.g., RGB, thermal, etc.) acquired by onboard cameras.

4.2.1.5 Hardware Security Module

C4D is working on different aspects of defining a standardized lower-level API for the easy and modular integration of a hardware security component into any modular drone architecture. To access the functionality of a Hardware Security Module (HSM), a multi-threaded architecture is designed. It is split into two main parts: the transport driver and the command library. The transport driver is communicating with the HSM, while the command library exposes the functionality to the higher-level user application.

4.2.2 Technologies for Safe Autonomous Decision

Command, Control, Communication, Computing, and Artificial Intelligence capabilities at the edge are required to enable autonomous navigation and commercially viable business opportunities. WP4 goal is to foster the industry growth with ease of integration and modularity.

WP4 focuses on systems and subsystems for enhancing mission-critical functions while increasing the autonomy of drones to enable safe and continuous operation. A modular approach is followed in which many function-specific components are being developed to serve the use-cases envisioned in the **COMP4DRONES** project (see D4.1/8). Since not all operations are required in all phases of the mission, in the few envisioned operational scenarios, a modular approach is beneficial. This approach ease of integration and personalization of the drone's platform for various use cases.

To enable autonomous (or semi-autonomous) operations, the standard Guidance, Navigation, and Control frameworks (GNC) is exploited. This framework is extended with newly designed systems and subsystems with task-specific responsibility. The modular approach defines expert systems and subsystems for functional execution. Each system comes with a detailed interface description, functionality, and application-specific programming interfaces.

4.2.2.1 Sensory Systems and Payload Technologies

Many of the **COMP4DRONES** technologies have the goal to enhance the perception of drones, where several novel sensing technologies are being developed (e.g., hyperspectral imaging, event-based sensing, position tracking technologies, and sensory fusion strategies as shown in Table 5).

Table 5: Sensory systems and payload technologies

Component Name	Component Description
Sensory Fusion	This component exploits sensory fusion between a dynamic vision sensor and a radar for robust real-time collision avoidance. By exploiting complementary sensors (vision and radar), this component can detect fixed and moving obstacles in low visibility conditions, low-light conditions, and in different weather conditions (fog, rain, and cluttered environment). It works by computing optic flow using monocular vision with an event-based silicon retina
Anchor and Tag Firmware of the Indoor Positioning System	The Indoor Positioning System (IPS) firmware comprises both anchor firmware and tag firmware. The IPS provides real-time, sufficiently accurate position information for enabling functional, safe navigation of a drone in an indoor structure. The solution relies on Ultra-Wide Band (UWB) technology.
Hyperspectral Inspection Payload	The hyperspectral payload shall collect georeferenced RGB & hyperspectral data and pre-process the data on the drone. The hyperspectral payload is composed of a novel hyperspectral camera image sensor, and an embedded GPU for image processing, and recordings.
Hyperspectral Inspection Processing Chain	The processing chain takes the collected hyperspectral data, restore the hyperspectral images, create multispectral cubes of the images, and indicate where corrosion is thought to be present in the image.
Integration of High Accurate GNSS Components in Actual RPAS Architecture	This is an architecture composed of one or more state-of-the-art GNSS receivers, antennas, communication links (if any) for precision agriculture applications. The architecture specifies the interface requirements considering common open protocols, and standards and new European GNSS differentiators if available. Position reporting to U-space service provider is also possible with this architecture.
Transponder for Drone-Rover Cooperation	This component provides the drone and the rover anti-collision and identification functionalities. It consists of Ultra-Wideband transceivers and the controlling and data processing embedded software. The component is capable of cooperative ranging (internodal distance measurement based on the propagation time of the radiofrequency signals) when multiple transceivers participate in the ranging procedure. It is capable of localization with respect to a relative frame. The drone is equipped with one transceiver, the rover can be equipped with two or more transceivers. Optional fixed beacons can be used according to the mission needs.
Shared Reference Frame	Shared reference frame is used for in-door, GPS-denied, cluttered, unknown environment. The base station needs to share its position estimate with all other drones during operation. Localization challenge, when beacons are moving in the map-frame.

4.2.2.2 Embedded System Function, Re-Configurability, and Machine-Learning Acceleration

Another large portion of the development in WP4 is devoted to enabling advanced functions on SoC technologies (see Table 6). These SoC technologies contain standard processors and microcontrollers (ARM/RISC-V) in conjunction with field-reprogrammable-gate arrays (FPGA). SoC platforms are industry-rated technologies that enable the acceleration of real-time data processing, sensory fusion, and advanced machine vision tasks (object detection, optic flow, obstacle avoidance, etc.). The flexibility provided by these technologies enables the change of electrical functionalities at runtime. Drones equipped with SoC technologies allows fast integration of advanced functionalities while enabling efficient acceleration of the most computational expensive machine-learning workloads. Several application-specific accelerators are being developed by the **COMP4DRONES** consortium for some of the most compute-intensive and challenging tasks such as simultaneous-localization and mapping (SLAM), neural network accelerators, visual analytics, and other compute intense workload that requires low-latency and real-time execution.

Table 6: Embedded system function, re-configurability, and machine-learning acceleration

Component Name	Component Description
3D SLAM Algorithms to Enable Autonomous Navigation	Optical-based SLAM algorithms require intensive computing power and processing complexity for applying such algorithms in drones. Optical SLAM functionalities are being developed that able to run on real-time GPU embedded (onboard) hardware.
Application-Specific AI/ML Accerator	This component provides a manager that takes as input a set of safety rules and drone's sensors information and performs a check and guarantees of rules to be respected. If safety rules are violated the manager evaluates the action that the drone has to perform in order to go back in a safe situation. The safety rules are defined by a risk assessment.
Embedded AI System	This system is of an embedded AI system with navigation system failures due to signal hijacking or system malfunction detection and reaction. The component is designed as an embedded, AI software module that will rely on commands received from the platform, geographical position derived from geomagnetic D-SLAM, GPS position, and mission polygon to detect possible GPS signal hijacking/spoofing and react accordingly.
Hardware-accelerated Optical flow and SLAM	Although SLAM algorithms are state of the art in mobile robotics, the computing complexity and the consequential power consumption is a barrier for applying such algorithms in drones. A hardware-based (FPGA) optical flow accelerator is being developed which would greatly reduce the complexity related to the frontend algorithms of the SLAM. Further, parts of frontend and backend algorithms will be probed for acceleration in a heterogeneous (FPGA + MPU) system with the long-term goal being the possibility of computing SLAM entirely in hardware.
Reinforcement Learning Stabilization	The objective of this component is to stabilize the UAV under non-nominal conditions complementary to the classical control techniques. It is based on artificial intelligence techniques and more precisely on Deep Reinforcement Learning (DRL). An artificial neural network (NN) is trained using the DRL technique to stabilize a UAV using a dedicated simulator until obtaining satisfactory results.
Embedded AI Obstacle Detection and Avoidance	An embedded algorithm based on Deep-NN that uses raw data from sensors like LIDAR, camera, GPS, and IMU to reach a goal position in an unknown environment. Through sensor fusion, the drone is able to react to environmental changes by performing obstacle detection and avoidance. No map required. NN is trained in tailored synthetic scenarios, then used in real ones.
Clearance Algorithm	The clearance algorithm must ensure the absence of intruders during critical phases: when dropping sensors and during take-off and landing.
Precision Landing	A set of sub-components are used to allow the UAV to land more precisely than with GPS. Additionally, this permits the UAV to land when the communication and GPS are not available.
Enhanced Navigation SW	The navigation SW is in charge of computing the geo-referenced position and attitude. It does that by fusing the information from the inertial sensors (IMU), from several GNSS receivers, and from other low-cost sensors (barometer, temperature). It uses advanced data fusion algorithms, e.g., extended Kalman filtering.
Bio-inspired Localization Algorithms	Grid cells, which were discovered more than a decade ago, have been shown to be a key component of a mechanism that provides updates about location. Nevertheless, there have been limited attempts on utilizing this discovery for application in robotics. These discoveries are explored with the long-term ambition of utilizing them for localization solutions for future drones. This includes starting off with training of Long Short-Term Memory (LSTM) Recurrent Neural Networks (RNNs).
Autopilot Navigation	This software component is in charge of the autopilot navigation capabilities embedded in the drone.
Autonomy, Cooperation, and Awareness	This component provides algorithms to meet functional requirements (autonomous and cooperative actions, and reference generation), operational requirements (management of critical situations with improved situation awareness and with power autonomy awareness), and usability requirements (compensation and rejection of environmental perturbations, measurement uncertainties, and faults).
Cooperative Planner	This software stack provides support for cooperation between drones and rovers on a (global) planning level. By sharing a drone's plans, in a standardized manner, more

	optimal group behaviour can be achieved, through a group planner component. By designing this group planner through multi-agent techniques, this planner can be distributed over multiple robots.
Map Enhancement	There are various sources for the creation of information-rich maps/grids, which can be used for SLAM and navigation. This component explicitly focuses on a generic way of providing query-like access to this information, providing temporal, shared situational awareness. This is achieved by leveraging the current map meta-information (timestamps and frame ids) augmented by additional meta-information per map-layer (e.g., trustworthiness, resolution, capture-time-range, etc.).

4.2.2.3 Interfaces and Control Technologies for Real-Time Mission Control

An equally important aspect addressed in WP4 is the creation of interfaces and control stations for mission real-time visualization, monitoring, flight management, geofencing technologies, and real-time control as shown in Table 7. Particular focus is devoted to the development of flight management systems that can be integrated into the Unmanned Traffic Management (UTM) system.

Table 7: Interfaces and control technologies for real-time mission control

Component Name	Component Description
Ground Control Station (GCS)	The GCS shall allow for the safe management of multiple agents (UAV, UGV, USV, humans, etc.) for large airspaces while complying with the latest regulations and being compatible with U-space services.
Avionics Encoder	Encoder for the reception, treatment, and parametrization of the video received from the 4 HD optics and tracking features.
UTM Ground Service	Ground service provides diverse functionalities both in pre-tactical and tactical phases. Precise ground data model gives the UTM system the ability to more precise calculations both in pre-tactical and in tactical phases of the flight.
UTM Airspace Structure	Airspace structure should be guaranteed. Obstacles and geofences should be properly marked in the UTM HMI for safety and awareness purposes. Flight rules (pre-flight phase) and alerts (flight phase) should be also applied based on this airspace structure.
UTM Flight Plan Management	Flight rules for proper Flight Planning Management should be implemented. Rules to be applied should comply current with regulations. Due to regulations are being continuously upgraded, flight rules should be reviewed regularly.
UTM Trajectory Algorithms	Trajectory algorithms will enable UTM to detect possible conflicts and raise specific alarms when the results of the calculations detect that a specific alert should be triggered.
UTM Flight Plan Authorization	The flight plan authorization process has lots of steps (some internal and some others need to be communicated to the end-users). All actors involved should have the ability to obtain flight plan status.
UTM Telemetry and Tracking	UTM system receives drone telemetry/track from a GCS/flying app. The UTM system may receive tracking information from various sensors and merge (when possible) the track in a unique central track for each flying drone.
Visual analytics	Visual analytics module for mission control and system monitoring. A hardware-in-the-loop simulation environment can be used as a base for such a GUI.
Droneport Traffic Control	Droneport (DP) Traffic control is system for multiple drone coordination during battery management. It monitors and predicts battery state of charge for each unit and manages the requests for Droneport

4.2.2.4 Runtime Security and Safety Systems

Runtime security monitoring and safety systems are a fundamental pillar of WP4. For the evaluation of the various risk scenarios in uncertain environments, run-time safety mechanisms and a framework are proposed. In the proposed framework the runtime functionality of the drone is ensured by monitoring the execution of predefined invariants (see

Table 8). Specific use case scenarios coming from the **COMP4DRONES** stakeholders have been considered, with potential system interventions for improving system safety.

Table 8: Runtime security and safety systems

Component Name	Component Description
Algorithms for Run-time Security Monitoring	A scalable ROS based module for intelligent decision-making component will be developed and tested for resolving critical situations (for example object avoidance). It has the following main specifications: be built in the chain of decision-making units (Autopilot, Human pilot), be scaled to existing drone architecture and configuration settings, take into account view dynamic factors.
Autonomous Decision Making in Critical Situations	The navigation system shall include a runtime manager to detect abnormal robot/drone behaviour (hardware or software failures, environment uncertainty), triggering a different execution mode (e.g., a safe degraded mode), or re-plan the mission altogether.
Path Planning Algorithms	The algorithms use the information coming from the environment (presence of obstacles, number, and position of the plants on which to operate the campaign acquisition or to apply the spryer, etc.) to compute the minimum-length path for the UAV (Unmanned Aerial Vehicle) or UGV (Unmanned Ground Vehicle).
Runtime Safety Monitoring	The proposed component advances the state of the art by providing a runtime monitoring module that deals with uncertain-unsafe situations by providing some kind of envelope of permissible behaviours, without compromising safety. 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.
Simcenter Amesim	Simcenter Amesim is a system simulation software package. It allows simulating physical multi-domain systems, to perform steady-state and transient analysis, and to test systems with MIL/SIL/HIL and Real-Time. It is also an open platform with interfacing capabilities to other software tools. In particular, it can be coupled with other software tools providing environment and sensor modelling capabilities (e.g. Simcenter Prescan) to analyse autonomous flight algorithms.

4.2.3 Component for Enabling Trusted Communication

The trust in the communication system that the project pursues means enlarging the conditions under which communications continue to function, regarding two aspects: the physical availability of the link (reliability), and accidental or malicious attacks to which the system is subjected (security).

In WP5, the first aspect is tackled by the components grouped under the name "Robust Multi-Radio", which address multilink capabilities and which, acting on the communications subsystem, improve the other UAS subsystems that rely on the former: fleet coordination, indoor navigation, payload data retrieval, vehicles monitoring and management. Moreover, a suite of components to solve specific fixed-wing UAV issues is proposed.

The second aspect has two specializations: (1) addressing security risks before an attack occurs, and (2) defending an attack while it is occurring. The former (the means to prevent attacks) is tackled mainly by the "Security Management" components that principally address the cryptography (from the hardware up to the protocols). The latter (the means to detect the signs of an attack and to react by activating the appropriate countermeasures) is tackled mainly by the "Reactive Security" components that apply to data transmission and to Global Navigation Satellite System (GNSS) and that employ algorithmic and Artificial Intelligence (AI) techniques.

4.2.3.1 Robust Multi-Radio Communications Components

The Robust Multi-Radio Communications Components offer the following solutions: integration of commercial off-the-shelf link technology such as IEEE802.11 and LTE; bandwidth aggregation and store-and-forward on the different available links; Low Power Wide area Network (LPWAN); support for a fleet of UAV; communication among Ultra-Wideband (UWB) nodes (fixed on the UAV) that support the positioning for indoor navigation; adaptation of video coding and compression to the link bandwidth; edge gateway to interface the entities participating in the network and cloud gateway for monitoring and management; a suite for fixed-wing UAV. Table 9 reports the summary of the Robust Multi-Radio Communications Components.

Table 9: Summary of multi-radio communications components

Component Name	Component Description
Link Manager and Scheduler	Link manager monitors connection availability and quality of the different base communication channels for drone use.
Link State API	An API to supply communication link state meta-information to path manager.
NEON Drone Communication Router	Collection of functional blocks for bandwidth aggregation and for dynamic reconfiguration of available interfaces (communication links). Traffic is distributed between available interfaces based on a configurable mode (per-flow or per-packet) for efficient and resilient communication.
Robust Communication	Software components for robust communications by means of store-and-forward methods, using mechanisms from Disruption Tolerant Networking (DTN).
Communication for UAV Identification and Monitoring	The component provides an integrated Low Power Wide Area Network (LP-WAN) communication link by which the unmanned vehicles can be identified and monitored.
Safe Fleet Communication	The generic architecture allows a fleet of UAVs (or agents) to collaborate. This architecture requires a synchronized Knowledge Base (KB) that stores the status of the mission and the UAVs. The KB needs to be shared among the agents that is achieved through the communication system. The component for the communication system is improved to use public 4G, or VPN and stream services, or a combination of both.
Enabling an Improved Indoor Positioning	Part of the Indoor Positioning System (IPS) suited for long indoor infrastructures (e.g. a gallery), where fixed Ultra-Wideband (UWB) transceivers (the anchors) are deployed, while a mobile one (the tag) is on the UAV. Measuring the propagation time of the UWB signals that anchors and tag exchange gives anchors-tag distances (ranging) from which (and from data fusion) the UAV position is computed. The component addresses the Medium Access Control (MAC), by reducing the collisions and optimizing the anchors that participate in the ranging (robustness), and the enrichment of the information that anchors and tag exchange and that the higher layers can access.
Adaptive Video Coding and Compression	Methodology to control video compression frame rate and down sampling according to the available bandwidth of the communication channel.
Communication Scheme for Unified System Management	Integration layer and orchestration of the system in which the other components are inserted. It offers different communication protocols, works with user-defined interfaces, provides transparent drones-cloud communications, and offers a unified view and management of the system.
Communications – Radio Links	Reception, management, and forwarding of HD (High Definition) video from UAV payloads to the operator's ground station, of telemetry from the UAV to the GCS (Ground Control Station), and of command and control from the GCS to the UAV.
Communications – Ports	Communications and wiring ports between payload and fuselage, and between fuselage and avionics for configuration, control, and video.
Communications – GCS-Autopilot	Communication between the frontend and the backend of the Ground Control Station (GCS).
Communications – GCS-CMPD	Human-machine interface of the GCS for communication between the GCS and the CMPD (Drone Mission and Data Processing Centre).
Communications – UAV-GCS-CMPD-UTM	End-to-end communication between the UAV and the Unmanned Aerial Vehicles Traffic Management (UTM) platform.

4.2.3.2 Security Management Components

The Security Management Components improve the security of the data transmission. The hardware component is a chip that supports the Transport Layer Security (TLS), hardening the security aspects of pure software-based systems that are prone to hacking and identity spoofing attacks. The software component on one side enhances existing security protocols with novel features (forward secure 0-RTT key exchange integration into TLS 1.3 for low-latency, integration of post-quantum cryptographic primitives for long-term security), and on the other side incorporates novel types of protocols such as the use of anonymous credentials for authentication with strong identity privacy.

Moreover, there are components that extend at system level and have adaptation capabilities to consider IoT applications (fixed sensors and a drone used as gateway), and to identify and correct issues that may suppose a threat to drone-to-drone and drone-to-infrastructure communications. Table 10 reports the summary of the Security Management Components.

Table 10: Summary of security management components

Component Name	Component Description
Hardware Security Component	The component addresses security aspects. It supports an enhanced Transport Layer Security (TLS). For security-reasons, it is a chip (hardware) physically separated from the microcontroller.
Cryptographic Primitives and Protocols	Collection of cryptographic primitives and protocols for drone constrained environments. The protocols provide means to satisfy both low latency requirements and strong security guarantees. Long-term security and resilience to quantum computers are also considered.
Generic Autonomic Management Framework	Component with self-adaptation capabilities for reliable and secure communications. A drone is used as a gateway for collecting sensor data, the system automatically tracks the situations at runtime and adapts the settings of the use case components (e.g., sensor nodes), when the unsatisfactory situations occur (e.g., due to weather conditions or other interfering wireless signals).
Security Management Toolchain	Component to ensure that the drone is free of known vulnerabilities. A node processes the information that the drone periodically sends, showing the inferred conclusions on a visualization interface it implements too.

4.2.3.3 Reactive Security Components

The Reactive Security Components have Introduction Detection System (IDS) capabilities: they detect and mitigate security attacks to data transmissions or to Global Navigation Satellite System. The components provide other functionalities such as encryption and decryption, help in establishing countermeasures, provision of reliable and accurate navigation data through GNSS and other on-board sensors data fusion. Table 11 reports the summary of the Reactive Security Components.

Table 11: Summary of reactive security components

Component Name	Description
Distributed IDS with In-drone Machine Learning–based Probes Detection	Lightweight Intrusion Detection System (IDS) works on network traffic patterns and on carried data plausibility, for drone to drone and drone to ground links. When possible, the IDS extracts information on the attacks for notifying the experts and proposing countermeasures.
GPS Spoofing Detector	The component detects spoofing attacks to the GPS based on SNR and related features. The algorithm employs Machine Learning techniques to parse sentences produced by the receiver in order to extract the relevant features and classify the signal.
Lightweight Cryptography	The component provides software encryption and decryption for resource constrained devices and has Intrusion Detection System (IDS) functionalities based on topology check.
Navigation System with Anti-jamming and Anti-spoofing Features	GLAD+ (GNSS-based Low-cost Attitude and position Determination) provides trustable positioning and attitude. Based on the fusion of multi-antenna/multi-receiver data and of on-board sensors, it can be adapted to the characteristics and to the expected dynamics of an UAV scenario. It provides the autopilot reliable and accurate navigation data and can be exploited for the payload, e.g., for digitization missions. The component includes jamming and spoofing detection, and so enables countermeasures.

5 Guidelines for Drone System Development

After selecting a set of technologies based on the concept of operations of a drone system, the step comes where the system is being developed. Thus, in this section, we present a number of guidelines that support the system development. These guidelines are about the development process, specific drone features, enabling the development of safe drones, technology re-use, mixed-critically aspects, evaluation and performance optimization, and hardware-based security.

5.1 The Development Process

In the following, we give some recommendations about the overall development process which include rapid learning cycles, decision analysis, and development of machine learning-based components.

5.1.1 Rapid Learning Cycles

Rapid Learning Cycles (RLC) is an adaptation of agile development for hardware and physical products, and other parts of the business where decisions are irreversible or expensive to change[10]. However, RLC provide a structured process to support overriding the killing of innovative ideas, by supporting teams to “fail fast to learn fast”[11]. Further, RLC support scheduling, optimizing resource utilization, and bring more decision points to projects in order to assess whether a project has sufficient probability of success. RLC also bring more frequent reflection and quick learning with previous work in mind, allocation of resources in manageable chunks, as well as making obvious mistakes in early phases of development. The main idea is to use the principles of agile to deliver value early, while avoiding the waste of revisited decisions. Idea is to pull “learning” forward in the process and this way to eliminate uncertainty. Another aspect is that irreversible decisions are postponed as long as possible, and then taken at the right time with the best available knowledge.

The RLC process is carried out in a number of steps:

- A concept paper with the aim of capturing and consolidating the voice of the customer;
- The identification of key decisions. Those are decisions that combine a high impact on the business case with the fact that the solution is unknown;
- Clear identification of the knowledge that is missing to decide with confidence;
- Systematic problem solving. The idea is to learn as quickly as possible to identify and remove obstacles, and to push decision to later to preserve flexibility;
- Periodic learning cycle events (meetings where learnings are captured and reviewed/challenged by experts). Typically, teams have two weeks to do as much as they can to close a knowledge gap, then they need to report what they have learned;
- Integration events (alignment meetings to close key decisions);
- Finally, the knowledge is captured in A3 summary reports for each knowledge gap and integration event. The A3’s also covers design and justification. The principle of using A3 reports is to condense what is usually written in large reports into one single sheet of paper.

Key decisions should not be made without the knowledge to make them confidently. The philosophy is that as the relevant knowledge gaps are closed, it becomes possible to narrow key decisions by eliminating weak parts of the solution space. Here, a knowledge gap is the “distance” between the knowledge already in possession of the organization and the knowledge needed to make a decision. Designing the question to cover the unknowns is essential. Examples are:

- What is the missing information regarding the customers interests and related targets?
- How large is the gap between the competitors’ solutions and the new solution?

[10] https://rapidlearningcycles.com/

[11] Katherine Radeka, The Shortest Distance Between You and Your New Product: How Innovators Use Rapid Learning Cycles to Get Their Best Ideas to Market Faster, 2nd Edition, 2017

- What specifications do you know that you cannot meet with your current understanding of the solution and your own company's capabilities?
- What makes the new solution "new" and different? Where are the gaps between the performance you have now, and the performance you need? What about cost, quality, new features and the user experience? What about waste in your customer's value stream?

Key decisions combine a high impact with a high unknown. Key decisions must be taken in order to complete the solution. Examples of questions that lead to a key decision are:

- What decisions must be made for the solution to succeed?
- Will this decision impact customers' interest or performance?
- Will this decision impact multiple systems, system architecture?
- Will this decision eliminate a large part of the solution space?

An example of rapid learning is the test-before-design approach, which supports learning and closing of critical knowledge gaps before detailed design is initiated.

5.1.2 Decision Analysis and Structured Decision Making

Decision analysis is the discipline comprising the philosophy, methodology, and professional practice necessary to address important decisions in a formal manner[12]. Decision analysis includes many procedures, methods, and tools for identifying, clearly representing, and formally assessing important aspects of a decision.

Decision analysis is a method to support evaluation and decision and to categorize various pieces of information and give a common opinion as a team. For larger decisions a form is used, indicating the decision, decision maker and the persons and functions, expertise and role and evaluation of the various options. Various alternative solutions are presented and the different criteria against which all solutions should be assessed are indicated.

A typical formal description of a decision is indicated in Table 12.

Table 12: Decision analysis scope

Name issue	Title
Description issue	The problem to be solved
Context	Circumstances in which the issue occurs
Deadline	The at-the-latest date of the decision
Cause	The reason for action
Consequences/impact	(Un)Wanted result of the issue for organization/project/product
Goals /requirements	The results to be achieved by decision
Boundaries	What is out of scope
Other	For example, assumptions made

For each decision a set of criteria is formalized. This can be done using Table 13.

Table 13: Criteria for each decision

		Description
S	Impact on strategy business/product line	[Description of relevant criteria]
T	Time (deadline/effort)	
R	Risks	
O	Other criteria (e.g., uncertainty)	
€	Finance (profit/loss analysis)	
F	Functionality/requirements to be met	

[12] Wikipedia. https://en.wikipedia.org/wiki/Decision_analysis

When the criteria and the interpretation of the criteria are agreed with the team of stakeholders, alternative solutions are generated. These solutions are then ranked by the team that makes the decisions. Table 14 indicates some ranking methods. Finally, the decision is documented and archived.

Table 14: Decision analysis methods

Decision Analysis and Resolution			
Simple methods		**Multi criteria methods**	
	Decree	Asses relative importance of criteria	
	Expert judgment		Pairwise comparison
	Extrapolations		Multi-role method
	Modelling and simulation		Team consensus
	Pros and Cons analysis	Assess alternatives against criteria	
	Satisficing		Ranking
	Study/survey		Scaling
	SWOT-analysis	Assessment consolidation	
	Testing		Even-swap
	User review		Simulation
	Voting		Operational decision analysis (ODA)

5.1.3 Process for Developing Machine Learning-based Components

Many functionalities of modern drone systems that are traditionally designed and implemented with classical ad-hoc solutions can be improved by adopting Machine Learning (ML) based approaches. Examples include (but are not limited to) the positioning system, the planner, the data acquisition software and processing pipeline. These functionalities are well-suited to be implemented via machine learning approaches due to the fact of being (a) heavily dependent on the availability of data; (b) not having an unambiguous mathematical model that control their behaviour and (c) nevertheless to depends on some hidden pattern in the data. The design of ML functionalities for drone systems are complicated by the presence of severe hardware constraints. These constraints are dictated by the fact that most drone platforms come with limited computational power, memory capabilities and network connections. Moreover, most drone functionalities need to operate in (nearly) real-time (e.g., planning and localization).

Addressing these additional constraints require enriching the design and development pipeline with additional macro-stages related to the evaluation of the HW/accuracy trade-off of the methods and the optimization of the algorithm onto the selected platform. A typical process would look like the following:

1. Feature Engineering (*)
2. Model selection (*)
3. Hyper-parameters optimization
4. Performance estimation
5. System design and implementation (*)
6. Deployment

Where the steps marked with (*) need to take into account the available HW. As for 1) engineering the features based on the HW might include severe restrictions as some features can be expensive to calculate. This is the case of high degree polynomial features or even expensive image patches. Similarly, 2) is limited by the available HW. As a concrete example, one can think of Nearest Neighbor (NN) classifiers. These non-parametric models are known to obtain the best possible prediction in presence of very large datasets. However, the memory occupation of such models is O(n*d) for n examples of d features each, and the prediction time is of the order of O(n*f(d)) if the similarity function takes f(d) time to be computed for each example. These facts together with the needs of very large datasets (i.e., n very large) imply that NN are not well-suited for embedded applications, either due the memory occupation or due to the violation of real-time prediction constraints. For these reasons, step 5) is dedicated to implement trade-offs that allows for constraints satisfaction. For example, if one stick with NN methods, a typical trade-off that can be implemented in 5) is based on the dataset reduction which reduce the memory demands of the method from O(n*d) to O(k*d) with k much smaller than n (e.g., k = log n). Notice that such reduction also reduces the prediction time to O(k*f(d)). The price to be paid is a drop into the accuracy of the classifier.

Step 5) also impacts other important decisions such as the prediction rate, i.e., the frequency at which the method is required to output a prediction. Indeed, each prediction has a cost in terms of energy and due to battery limitations, it is necessary to optimize the number of predictions for a given functionality. The optimization needs to fulfil the autonomy requirements of the system while ensuring the timely classification of the monitored event. As a result, optimizing for the energy requires a trade-off even on the real-time requirements which, as we already mentioned, are also impacted by the algorithm itself. It might even be the case that the resulting set of constraints is unfeasible and HW upgrades either in terms of computations or in terms of battery are required. For this reason, the pipeline is executed in cycles. Finally, step 5) also decides which part of the ML system shall be implemented in HW (if any). One recent trend is that of the FPGA-based design, where the computationally demanding algorithms are directly realized in FPGA. This approach alleviates the burden of the prediction time and the related real-time constraints, but introduced additional constraints on the energy consumption and the mechanical design of the drone.

As a result, the design of ML functionalities for drone systems is a complex and challenging process where the well-known benefits of such an approach need to be fit within a stringent set of HW constraints that, if not taken appropriately into consideration, might undermine the realization of the project.

5.2 Enabling the Development of Safe Drones

Current drone architecture was designed after JARUS CS-LURS document (Version 1.0 from 30-10-2013) - https://www.nlr.nl/downloads/jarus_cs-lurs.pdf. In this document, a single failure on critical components causes safety risk**,** and then a solution for single component failures of safety critical components is needed. This solution targets failures on components such as:

a) Motor
b) Remote control
c) GPS
d) Flight controller
e) Battery/Power supply
f) C3 link system

For failures in points a, b, and c, there are common solutions like standard failsafe operations foreseen in standard flight controllers like DJI A3 and Pixhawk. For failures of points d, e, and f there are special requirements for authorization of drones in populated area. In the following, we describe methods to cope with such types of failures.

5.2.1 Flight Controller Failure

Critical failures like flight controller failure required a solution. This solution can be achieved by implementing a second flight-controller with a redundant power supply that is in parallel use of the main controller. Just in case the output signals coming from the controllers to the motors have a bad signal, the so called “slave controller” takes over control. Bad signals can be caused by:

- Total collapse of power supply to controller
- Loose or broken input cable to controller
- Single component failure of controller

Both controllers give the PWM signals for motors to a redundancy board that has an automated relay switch if any of the PWM output signals is out of reference. In order to that, even when the main controller has a 100% failure of the whole component, the aircraft can be controlled fully by the second controller in use.

As an example, drones like the VER COMMON DJI M600 can be equipped with two A3 flight-controllers separated / connected via a redundancy board. In Austria, there are some redundancy boards on the market. They are all licensed with reference to Austrian utility Model AT 13698. This utility model was filed by Mr. Richard Koch. The structure can look like shown in Figure 11.

5.2.2 Battery Failure

It is required to have a redundant battery (pack)/ power supply. When one battery fails it must be avoided, that the battery damages the second used battery (pack) due to a short circuit or similar!

During flight, when one lithium polymer (Lipo) battery pack gets a short circuit, the battery safety board decouples the Lipo from the board net.

There are several solutions for decoupling circuits and provide LiPo redundancy. As an example, Figure 12 and Figure 13 show two possible setups with Metal Oxide Semiconductor Field Effect Transistor (MOSFET) and one with power diodes technology.

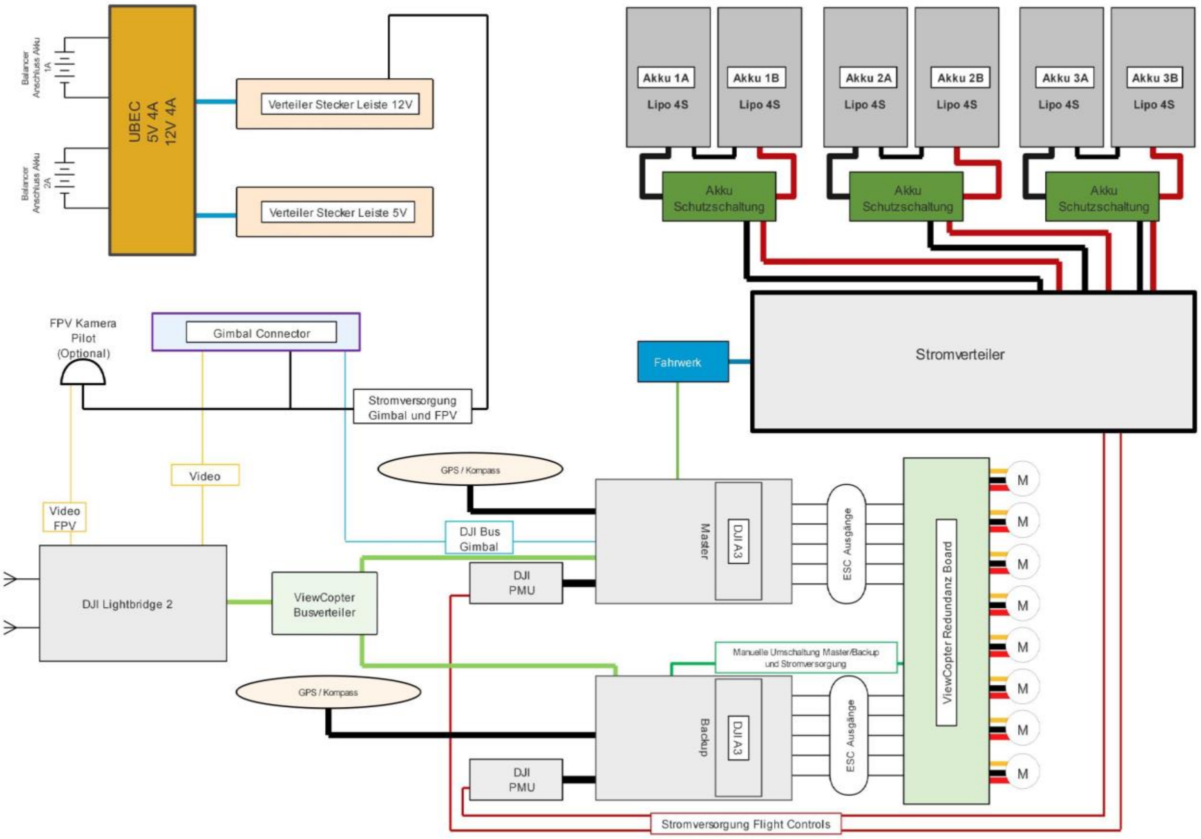


Figure 11: Position of redundancy board within the drone infrastructure

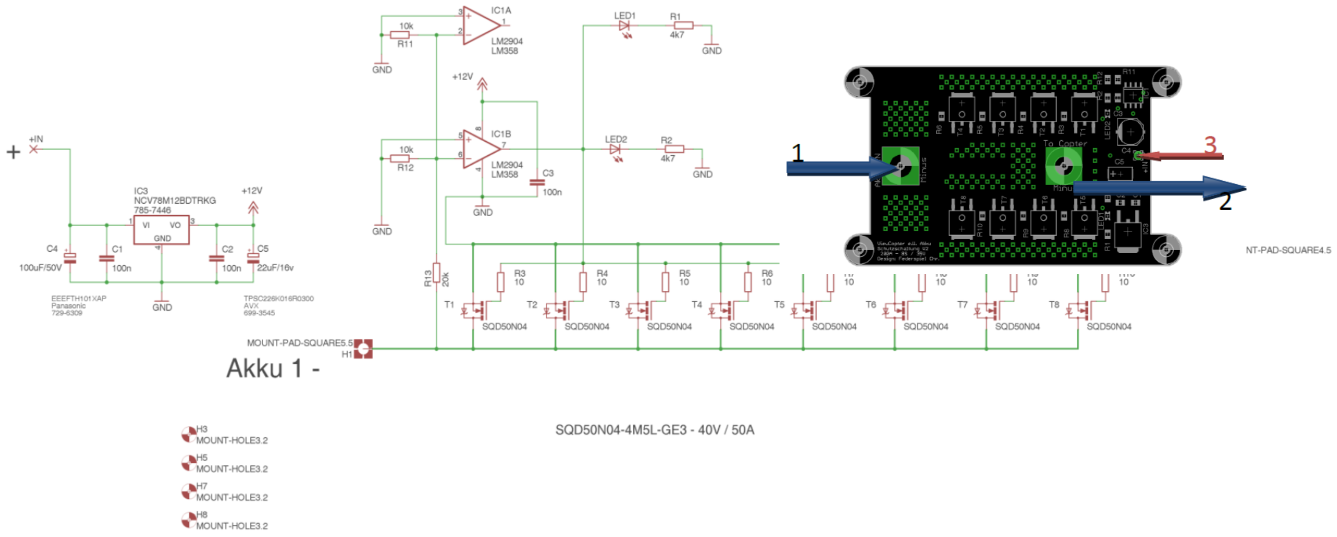


Figure 12: LiPo decoupling - MOSFET

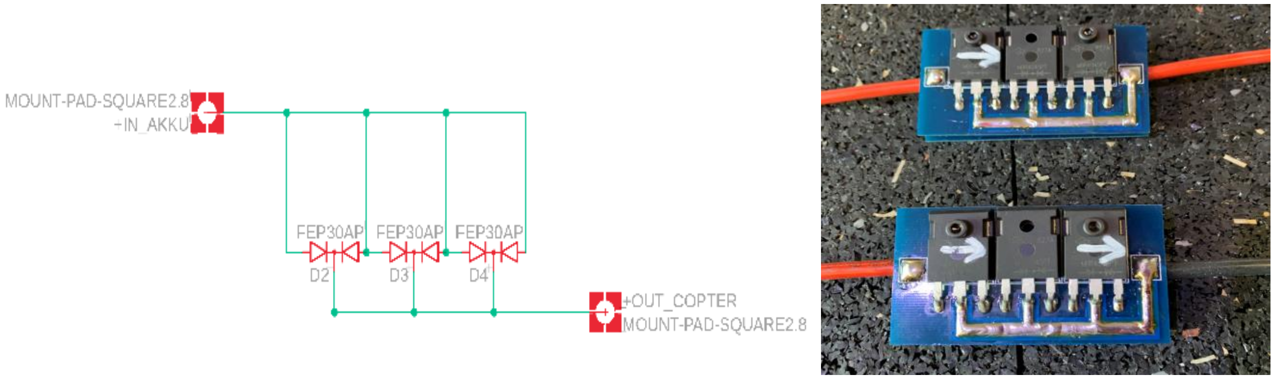


Figure 13: LiPo decoupling – DIODES

5.2.3 Automatic Battery Manipulation/Replacement

Automatic battery replacement system must ensure safe and a reliable service to add a true autonomy to drone operations. The reliability of the system depends on the reliability of each subsystem: battery module mechanical connection, electrical connection of power connector, data connectors, process of battery replacement (automatic manipulation), control system, etc.

The necessary to take in account that the drone batteries are not equipped with over current protection or controllable contactors usually. It means that any connection of battery with wrong polarity, short circuit connection or connection with high transition resistance leads to serious damage of drone components or whole drone including fires of batteries for example.

Battery module mechanical (Figure 14) connection should:

- Eliminate the possibility of reverse polarity placement (module and connector position should not be symmetrical).
- Mechanically connect the module properly or does not allow to connect (to avoid the indefinite connection).
- Indicate the proper connection to the system (end switches, control circuit, etc.).
- The design should take into account wear of mechanical part and wear should not degrade electric connection (pin misalignment, lower connection force, etc.).
- The module should not be sensitive to pollution, dust, etc.

- Live part of power contacts should not be easily accessible or prone to short circuit (not in one surface and as far as possible).

For power connector (Figure 15):

- It is highly recommended to use multi contact connector for each pole to ensure redundancy.
- Spring loaded contacts are recommended (long stoke if possible).
- Dimension of connector should allow normal operation on 70% of contact pins properly connected only (contamination, damage, etc.).
- For system above 25V is recommended to use anti spark system (necessary for systems above 42V). Sparking degrades contact surface quality and increase the translation resistance of connector.
- Elimination of high current during the connection (high capacity of drone, payload, etc.) all pins are not connected at once.
- Power connector should provide information if is connected properly.
- If equipped with data pins, they should be located not in the neighbourhood of power contacts for safety reason.

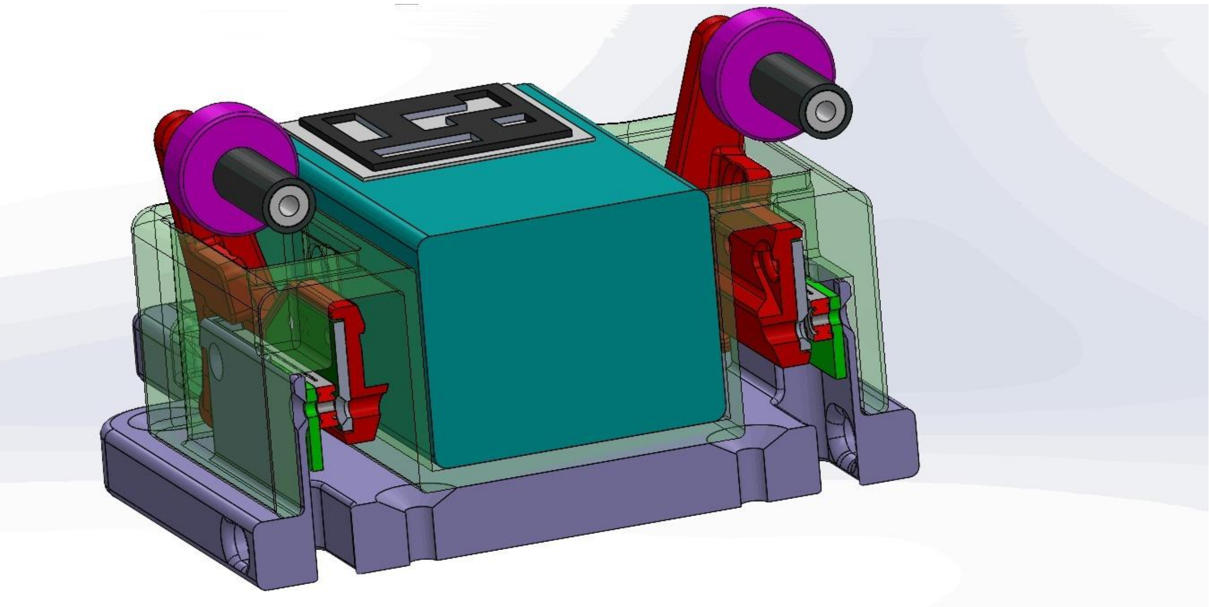


Figure 14: Battery module example (cut view)

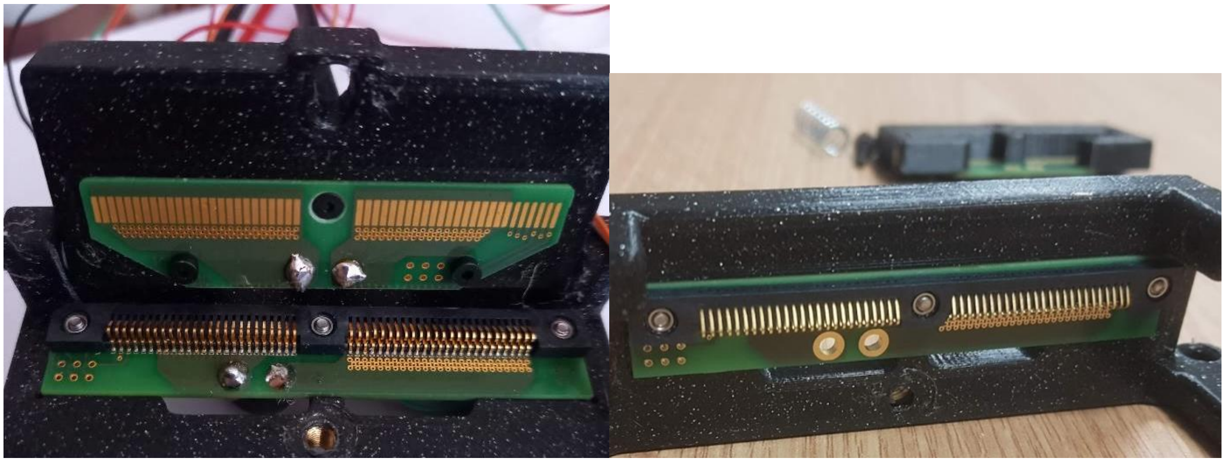


Figure 15: Multipin connector example

Control system should:

- Collect the condition and state of each managed battery.
- Give the information about inserted battery to the drone (real battery capacity, health of the battery).
- Identification, state and health of the battery could be integrated into the battery module (NFC tag in example), or this information could be a part of battery database managed by control system (battery identification is necessary only (QR, ARUCO code, NFC, etc.).
- Release the drone for next operation after all check are completed OK.
- The drone is not power during the battery exchange. If continues power-up is needed, it is necessary to equip the drone with back up battery (for the controller only) or add an auxiliary power source.

5.2.4 C3 Link System Failure

C3 (command, control, and communication) link systems are enablers of end-to-end communication between UAV and control systems, and as such, are one of key systems supporting the UAV operation.

According to step #9 b) of JARUS guidelines on Specific Operations Risk Assessment v2.0, no probable failure of the UAS or any external system supporting the operation shall lead to operation outside of the operational volume. Thus, compliance with this requirement shall be substantiated by a design and installation appraisal and shall minimally include independence, separation and redundancy feature.

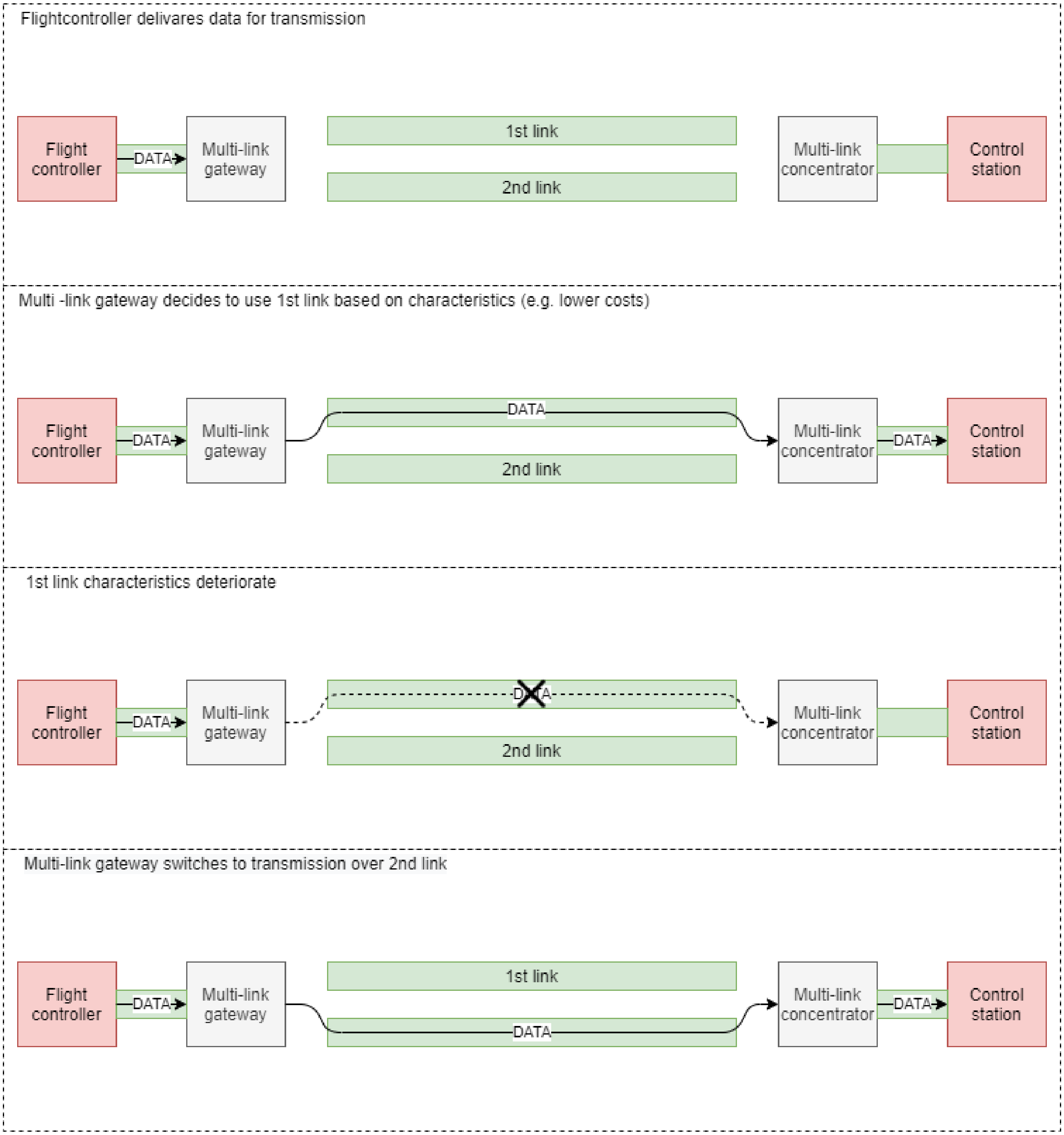


Figure 16: Introducing redundancy to C3 link systems.

When explaining robustness of C3 link system, one usually references to ability of system to deliver despite the changes happening in its environment or even system itself, and to achieve such resilience in scope of link systems, redundancy is introduced. Redundancy (aka multi-link) may be simple by using same link technology (e.g., 2 LTE modules using different operator, or 2 BLE connections to 2 different AP) or may be complex by multiplexing across different technologies (e.g., using in parallel 2 LTE modules with different operator and additionally BLE connection), thereby using each link independently. In described scenarios if characteristics of any of the links deteriorates, the link working in parallel can take over and send/receive data instead as described by Figure 16.

Multi-link approach increases overall integrity and assurance level of communication system, thereby providing better availability. Additionally, as link technologies differ by its characteristics (latency, bandwidth, data rate, costs, etc.), multi-link approach may also provide increases in overall link quality and cost-effectiveness by utilizing advantages across different link technologies.

5.3 Re-using Technologies from Existing Platforms

The **COMP4DRONES** project takes a step-by-step procedure for developing UAS. First the system's concept of operations is identified, based on the user requirements, through which the technologies are selected for usage in the drone platform. This process is supported by tools which adhere to the architecture defined. However, in real-life UAS projects, the process would not be as linear and sequential as this base architecture defines. The main reason is that projects are not done in a vacuum, and no project fully starts from scratch within the project organizations. Companies that start on a drone project will have existing tooling, procedures and most importantly, existing drone platforms available.

It is realistic to assume that the availability of existing drone platforms (within the project's organization) will influence the ConOps process. In the assessment of user requirements, in the discussion on project scope and in showing the users what is possible, these prior available technologies and platforms are important drivers for future projects.

This is an important measurement for the maturity of technology. The level of efficiency and reuse of such technology. In the case of drones, an important example of such reuse lies in the drone platform itself. A drone service provider does not need to obtain, design or implement a new drone for each mission. A drone platform is an investment in a reusable asset, and as such, the availability of that asset should influence the mission design. Existing assets form a baseline and starting point for the scope of mission planning.

In Figure 5, **C4D** general UAS development procedure, this opportunity for reuse drives the development of the demo, feeds the ConOps process and helps selecting and filtering the key technologies available for implementation of the UAS.

5.4 Development of Mixed-critical Drone System

Mixed-criticality refers to systems integrating applications or components with different criticalities in terms of safety, but which still share the same execution platform[13]. Drones and drone-based systems are mixed-criticality systems in nature. This can be observed in the usual fact that they integrate flight control and payload electronics, normally associated to safety and mission critical functionalities. Notice that high integrity systems, like avionics, have traditionally relied on resource separation and redundancy techniques. However, the need for safety and at the same time efficiency and lower costs is a game enabler in the drone field. Thus, there is a push towards more integrated HW/SW platforms, which intensifies the resource sharing, and thus the mixed-criticality aspect.

Criticality refers to the risk associated to a failure or malfunctioning on a given component or application of the system on the safety of persons. A wider definition could lead to consider also damages that could be economical. Several standards exist, associated to different domains like electronics (e.g., IEC 61508)[14] , avionics (DO178B)[15], automotive (ISO2626)[16], etc. While they present differences, they seem to agree on considering at least two aspects when evaluating the risk, i.e., the impact and the probability of occurrence. The impact measures the intensity and scope of the consequence. For instance, when

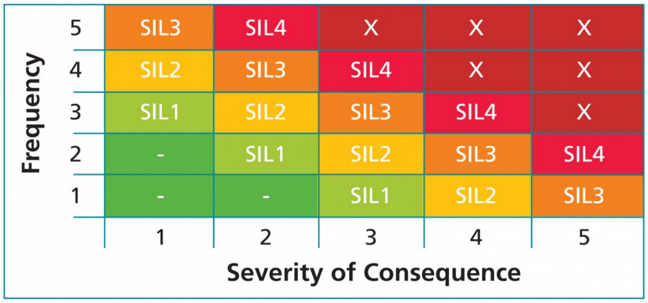
[13] Baruah, SK; Burns, A; Davis, RI. "Response-Time Analysis for Mixed Criticality Systems" (PDF). University of York. Retrieved 19 February 2013.

[14] IEC International Standard. Functional Safety of electrical/electronic/programmable electronic safety-related systems – Part 1: General requirements. Ed. 2.0. 2010, APrl.

[15] https://en.wikipedia.org/wiki/DO-178B

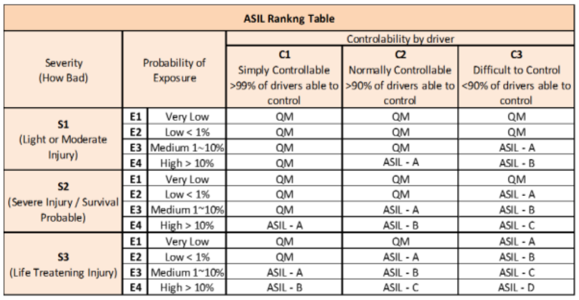
[16] ISO 26262-9:2018 Road vehicles — Functional safety — Part 9: Automotive safety integrity level (ASIL)-oriented and safety-oriented analyses.

evaluating personal impact different types of scales can be posed, e.g., number of persons dead, catastrophic/critical/marginal/negligible. These scales can be combined and extended with impacts on scenarios/objects (e.g., catastrophic/damages/no-damages). The risk of the occurrence is another key factor for criticality which has leads to quantifications (e.g., failure likelihood, mean time to failure).



Frequency					
5	SIL3	SIL4	X	X	X
4	SIL2	SIL3	SIL4	X	X
3	SIL1	SIL2	SIL3	SIL4	X
2	-	SIL1	SIL2	SIL3	SIL4
1	-	-	SIL1	SIL2	SIL3
	1	2	3	4	5
	Severity of Consequence				

(Source : www.crossco.com)



ASIL Rankng Table

Severity (How Bad)	Probability of Exposure		Controlability by driver		
			C1 Simply Controllable >99% of drivers able to control	**C2** Normally Controllable >90% of drivers able to control	**C3** Difficult to Control <90% of drivers able to control
S1 (Light or Moderate Injury)	**E1**	Very Low	QM	QM	QM
	E2	Low < 1%	QM	QM	QM
	E3	Medium 1~10%	QM	QM	ASIL - A
	E4	High > 10%	QM	ASIL - A	ASIL - B
S2 (Severe Injury / Survival Probable)	**E1**	Very Low	QM	QM	QM
	E2	Low < 1%	QM	QM	ASIL - A
	E3	Medium 1~10%	QM	ASIL - A	ASIL - B
	E4	High > 10%	ASIL - A	ASIL - B	ASIL - C
S3 (Life Treatening Injury)	**E1**	Very Low	QM	QM	ASIL - A
	E2	Low < 1%	QM	ASIL - A	ASIL - B
	E3	Medium 1~10%	ASIL - A	ASIL - B	ASIL - C
	E4	High > 10%	ASIL - B	ASIL - C	ASIL - D

(Source : www.quality-one.com)

Figure 17 - Risk Assessment tables in terms of SIL (IEC61508) and ASIL (ISO26262).

Another aspect where it seems there is coincidence is in the need for an objective attribute for criticality, i.e., on a criticality level. On each standard such a criticality level receives a specific name (SIL in IEC 61508, ASIL in ISO 26262, DAL in DO178B), and it is defined upon a map relying on how impact and risk scales have been defined.

The criticality level (criticality in short) utility and importance is not just related to its ability to perform comparisons and taxonomies. It is a key starting information for design and verification methodologies. This way, certification/standards rely on them to state aspects like design and verification procedures (e.g., demand for performing unitary test, code coverage analysis, etc.), constraints on them (e.g., separate design from verification team), or specific requirements (e.g., probability of failure below a threshold).

5.4.1 Mixed-Criticality on analysis

Hard-real time research community has pioneering on the application of criticality levels on hard-real time schedulability analysis. The Baruah et al. work referred at the beginning of section 5.4 illustrates the added value of considering mixed-criticality information, to make possible at the same time certification and real-time behaviour for non-safety critical tasks. In this context, a clear distinction between criticality and priority is done. While priority is a possible dynamic attribute used in the scheduling decision, and thus related to a scheduler, criticality is an attribute associated to tasks, to indicate which of them will be associated to a safety certification process. Those tasks are the ones subject to the analysis of the certification authority with more stringent constraints (higher worst-case-execution times or WCETs). The relevant fact is that the criticality information has a role and impact on schedulability analysis and its results (basically, schedulability assessment and scheduling) to get real-time guarantees both for the whole set of applications and for those subject to certification.

While schedulability analysis is a main design area where mixed-criticality needs to be addressed, it is not the only one. Schedulability analysis is about static analysis, able to consider resource sharing, i.e., tasks executed over a single processor, time interferences and communication at some extent, with an eminently analytical approach which obliges to constraint model complexity and leads complex formulation and meritorious efforts from last research. However, as said, there are other type of design activities and approaches. In contrast, other approaches, like CONTREX project[17], complements activities in the areas of predictable computing platforms and segregation mechanisms with techniques to consider non-functional properties (called extra-functional) properties in their context. A first complementary aspect is that CONTREX put the focus on extra-functional properties, to cover not only

[17] K. Gruttner et al. “CONTREX: Design of embedded mixed-criticality CONTRol systems under consideration of EXtra-functional properties”. Journal of Microprocessors and Microsystems. V5, Pages 39-55. June, 2017.

time requirements, but also requirements on power, energy or temperature. Thus, in the frame of its avionics demonstrator, CONTREX shows how the excessive computational load in the payload functionality can raise the temperature of the SoC area where the general-purpose processor is executing and causing an interference on the predictable processors in SoC devoted to navigation and flight-control. The ability to predict and providing tools to avoid by design, or via monitoring and reacting on these extra-functional interferences, paves the road to enable size, weight, power and cost efficient an implementation of yet save and certifiable drones.

Another complementary aspect was addressed at CONTREX. At the architectural level, monitoring mechanisms at HW and SW levels, to allow the detection of metrics leading to constraints unfulfilled to risks. Moreover, at the design process level, it proposes of a multi-level simulation-based approach, with the consideration of virtual models comprising both software and hardware, at different levels of detail, and capable to estimate the non-functional properties (time, energy, power, temperature metrics) of interest.

Moreover, the consideration of criticalities has been also proposed for other design activities and combining the aforementioned analytic and simulation-based approaches[18,19] where a combination of static analysis (based on constraint programming) and simulation-based analysis (over performance models) is proposed for faster, but criticality aware design exploration at early design stages.

5.4.2 Mixed-Criticality on Modeling

Up to this point, most usage of mixed-criticality information at the design flow has referred to some type of assessment or analysis. A key related aspect is modelling, i.e., how the information is captured and transferred to the design phases, what kind of elements need to be captured and how they can interrelate. The CONTREX UML-MARTE methodology[20, 21, and 22] proposes the ability to capture criticality information as an "abstract" (standard semantics independent attribute), while at the same provide a modelling element to associate criticality to a given standard semantics for a specific model. Moreover, as this methodology is a component-based methodology, supporting different levels (application component, RTOS components, HW components), it also proposes the association of criticality to components. Furthermore, a maybe more disruptive proposal of this methodology is the ability to support the association of criticalities to requirements on extra-functional properties. This is not that strange if it is realized that the criticalities explicitly associated to tasks of schedulability analysis approaches, actually lead to an implicit requirement on a task related extra-functional property, i.e., its response time. In that case, the criticality is actually on the requirement that such a response time does not exceed a given deadline. Starting from this baseline, the methodology proposes a wider, more generic association of criticalities to requirements on extra-functional properties (not only time, but energy, power, and other type of extra-functional requirements). This should enable the consideration that an application (captured as an application component) can have associated more than one requirement (e.g., on response time and on power budget), each with its own criticality. It will allow to consider that a requirement of a given criticality can be transversal, i.e., be associated to several components, or even to the whole system.

[18] Herrera F., Sander I. (2015) Combining Analytical and Simulation-Based Design Space Exploration for Efficient Time-Critical and Mixed-Criticality Systems. In: Louërat MM., Maehne T. (eds) Languages, Design Methods, and Tools for Electronic System Design. Lecture Notes in Electrical Engineering, vol 311. Springer, Cham. https://doi.org/10.1007/978-3-319-06317-1_9.

[19] F. Herrera, I. Sandeer, K. Rosvall, E. Paone. "An efficient joint analytical and simulation-based design space exploration flow for predictable multi-core systems". In RAPIDO '15: Proceedings of the 2015 Workshop on Rapid Simulation and Performance Evaluation: Methods and Tools. January 2015 Article No.: 2 Pages 1–8 https://doi.org/10.1145/2693433.2693435

[20] F. Herrera. "UML/MARTE modelling for mixed-criticality systems". Tutorial "CONTREX: Virtual Integration Testing for Mixed-Criticality Systems under Consideration of Power and Temperature Constraints" in HIPEAC 2016. 2016-01

[21] https://www.teisa.unican.es/gim/pub_files/file_603.pdf

[22] https://www.teisa.unican.es/gim/pub_files/file_585.pdf

5.4.3 Mixed-Criticality on Cyber-physical systems

The aspects are also applicable on cyber-physical systems (CPS) and cyber-physical Systems-of-Systems (CPSoS) [23]. It is the case of models that go beyond the drone electronics boundary when considering one drone (CPS) or a drone swarm (CPSoS), and the modeling of physics, aerodynamics, environment (wind, objects, etc.) via drone simulators (e.g., Paparazzi, Airsim) or robotic frameworks (e.g., ROS and Gazebo). For those, higher-level view models, it is likely the need to consider different criticalities for drones, that can be modelled as a sensor and actuators, or for their subsystems.

5.4.4 Impact on COMP4DRONES

All the mentioned aspects related to the capture and usage of mixed-criticality on the models and specifications of drone systems are relevant both for the **COMP4DRONES** general architecture (developed in D3.2) and for the project methodology (addressed in this report).

With regard to the **COMP4DRONES** methodology, noticing that the building blocks considered in D3.2 regard to system-level (or SW component), a main consideration is on the convenience of enabling the capture, at least, of the criticality level (as a standard independent attribute) and let its association to each basic block.

For the **COMP4DRONES** methodology, addressed in this report, the possibility of considering resource sharing relevant for performance analysis is mandatory. Such resource sharing can happen at several dimensions (computing, storage, and communication) and levels, i.e., several threads can clash within a process, several applications on a partition, several partitions on a processor, etc. For this assessment, is interesting to understand the type of support provided by tooling associated to the methodology (these clarifications are introduced in WP6, and moreover if they support mixed-criticality information at their input).

Finally, it is relevant to push an overall understanding on how **COMP4DRONES** tuple (architecture, methodology, and tools) enable to tackle safe design of drones as mixed-critically systems, such it can push the adaptation/flexibility of current regulations and standards. By the end of the project, it should be possible to assess if that **COMP4DRONES** tuple can contribute in some way to alternative solution to the redundancy requirement imposed by some regulations pointed out in section 5.2. Likely, **COMP4DRONES** will not be coping with all the problematics, but it can help on many of them. For instance, trying to answer questions as if it possible to instantiate two flight controllers (or other critical building blocks of the C4D general architecture) on the same SoC? And if it is possible to guarantee via segregation mechanisms or via architectural, procedure, and tool solutions the functional integrity and the performance, at least up to some bounds (with the same type of quantifications and assessments as for the redundant solution) to convince the regulatory boards on this feasibility? In this way **COMP4DRONES** has the chance to, not only to consider mixed-criticality as it is expected for the existing standards/regulations, but also to impact on their future evolution.

5.5 Evaluation and Performance Optimization

In this section, we describe two main concepts affecting the architecture evaluation and performance optimization which are the concept of functional chain and the software stack.

5.5.1 Software Architecture and the Concept of Functional Chain

A functional chain (see Figure 18) is a set of dependent functions. It can be represented as a chain of functions executed in order, usually starting from sensor(s) acquisition and ending at some behaviour change in the system, including a command sent to actuator(s) in order to change the physical state of the system. Critical functional chains are subject to real-time constraints that apply on end-to-end

[23] F. Mallet, E. Villar, F. Herrera. "MARTE for CPS and CPSoS: Present and Future, Methodology and Tools". Available at https://hal.inria.fr/hal-01671190/document.

delays. End-to-end delays are the main metrics that architecture evaluation and performance optimization rely on.

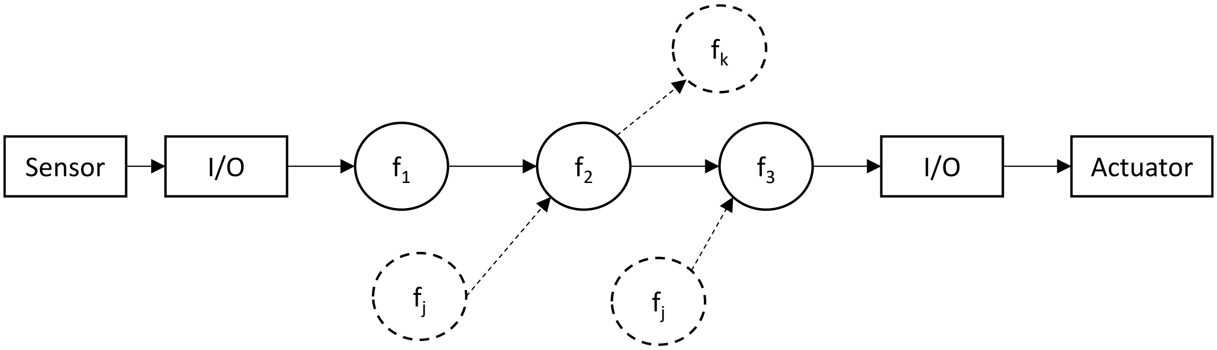


Figure 18: Example of a functional chain, dashed lines represent other functions and dependencies that are part of other functional chains

Threads are the basic execution units of functions. In some cases, interrupt service routines can be seen as very light threads with limitations. Therefore, a function, to be executed, has to be executed by a thread. Except in some very specific cases (e.g., CUDA, OpenCL), threads represent a sequential execution of the functions they contain, and can be executed on one core at a time. Several threads can nevertheless be executed in parallel on different cores, or interlaced on the same core, using time slices controlled by a scheduler. The existence of several threads (except for the specific case of only one thread – the main program – and interrupt service routines) requires the use of an operating system, in charge of arbitrating the use of the core(s) using a scheduler.

Threads can be activated only in the following circumstances:

- One thread maximum per core can be active all the time;
- Clock activation, a thread is activated following an interrupt triggered by the internal clock. Theses threads are most of the time periodically activated. These threads are called time-based.
- Activation by interruption: a lot of Input/Output (I/O) devices can trigger an interrupt on certain events. These interrupts can be used to activate a thread. A thread activated by an interrupt other than the local clock is called event-based.
- Activation by another thread using a synchronization mechanism (private semaphore, mailbox, etc.): a thread waiting on an activation mechanism inherits from the rhythm of the activating thread.
- The internal clock can also serve for specific threads which are activated when an event (often an I/O triggered interrupt) does not occur in an expected time interval. These threads are called watchdog threads.

It is a hard point to reason on end-to-end delays during the design of the system because at this stage a lot of variables are unknown. For example, the worst-case execution time of a function on the target platform and the software stack (e.g., operating system) requires an implementation of this function to have been already made on this platform, and characterization of the worst-case interference of other threads executing, either on the same core, or on a different core that shares contention points with the considered core. Then the worst-case response time of the hosting thread can be computed using scheduling theory. Finally, at the end, when an implementation of the threads on the target platform exists, timing analysis theory can be used to compute the worst-case execution time of the threads.

A good way to proceed during the design phase, which is used in civil avionics, is to work with time budget for the different functions (that can be included in their hosting thread). Later in the validation phase of the software, the fact that threads meet their timing budget has to be verified. If it is not, corrective actions must be taken.

The activation pattern of the thread hosting a function has a major impact on system performance. For example, in Figure 18, if the sensor was to send an event, used by the I/O device to trigger a thread, and the thread was executing in sequence f_1, f_2, f_3, then the worst-case end-to-end delay of the function chain starting at the time an event occurs on the physical quantity measured by the sensor, and the system response on the actuator would simply be the worst-case response time (WCRT) of the thread plus a technological delay implied by sensor, actuator, and I/O devices. If rather than being activated by the input device (event-based), the task was periodic with a period T (time-based), the worst-case end-to-end delay would be T plus the WCRT of the task plus the technological delay. A rather inefficient design on a simple uniprocessor platform regarding the technological delay would be to host each function in a different periodic thread with respective periods T_1, T_2 and T_3, and have them to communicate asynchronously. In this case the worst-case end-to-end delay would be T1+T2+T3+$WCRT_1$+$WCRT_2$+$WCRT_3$+ technological delay.

If the system is distributed on several processors, then of course, transversal time of the networks used to transport the outputs of a function to the next function also has to be added. It can be noted that since taking into account several practical factors (e.g., mutual exclusions) makes exact schedulability analysis computationally intractable, the more complex the system, the more conservative the schedulability analysis method, the more pessimistic the performance analysis becomes.

A general rule of thumb is that when a functional chain crosses an asynchronous communication point (periodic thread doing polling on a sensor, or communicating functions hosted in different threads having their own activation rhythm), the cost on the end-to-end delay of the functional chain is to add the period of the thread reading the output of the previous element in the functional chain.

Therefore, it is generally more efficient to have a threading model as simple as possible, using as few threads as possible for the system.

The UAV designers usually follow this rule of thumb, and most open-source cots autopilots use a central thread, which could be called a cyclic executive, to execute most of the critical core functions of the autopilot.

There are several benefits of having a simple multithreading architecture:

- Reducing end-to-end delays.
- Reducing the overhead due to scheduling (less preemptions, leading to less cache memory tempering by other threads).
- Less memory usage because less data has to be replicated in different threads, moreover, the memory stack to reserve for a thread executing a sequence of two functions is always smaller than the sum of two memory stacks to reserve if the two functions were executed by their own thread.
- Less debugging effort.
- Less problem posed by the fact that events may not be perceived by two functions within the same execution of a functional chain. As an example, if there is a mode change, within the same execution of the functional chain, a function in a thread may have perceived the mode change, while the next function of the functional chain may have started its thread execution earlier, before being preempted, and not have noticed the mode change.
- In the extreme case where there is only one thread and interrupt service routines, it is possible not to use an operating system (OS), reducing the effort of analysing the OS impact.

The benefits of threads in an application are:

- Ability to start threads on events (event-based threads), reducing the delay between the occurrence of an event and the execution of the corresponding function.
- Fault tolerance: if a thread is event-based, and if this thread is part of the execution of a functional chain, then the absence of the event required to execute this thread could jeopardize the execution of the functional chain. In this case, using threads, such as watchdog threads, can be

an alternative that can be substituted to the event-based thread in the case where the event does not occur.

- Allow preemptions: a long function, if included in a cyclic executive, will delay the execution of the next loop of the executive. Including it in a concurrent thread will allow this function to be preempted by other tasks (or concurrently executed if several cores are available).
- Take profit of several cores at the same time on a multicore platform.

5.5.2 Software Stack and Hardware

For the sake of self-validation up to certification by a third party, it is necessary to compute the Worst-Case Execution Time (WCET), to be able to compute the Worst-Case Response Time (WCRT) of threads and thus the end-to-end delays. It imposes some requirements on the platform and the operating system. An important concept introduced in ISO 26262 is the notion of freedom from interference. A function of interest, taken in isolation, can be considered free from interference if the interferences created by the execution of other functions on the function of interest can be bounded. This central concept in software safety can be derived all through the hardware to software stack.

A function executed, either in a thread (that can be the main program in a mono-threaded application) or an interrupt service routine, is sharing several resources with the other functions executing on the same platform. Any shared resource can be seen as a contention point, with an arbitration policy:

- The processor is arbitrated by a scheduler, usually implemented in the OS. Scheduling theory allows to characterize the maximum interference of the other functions and/or threads and/or processes.
- In an OS, memory is shared among threads of the same process, therefore any access by another thread (by mischief or inadvertently) to the memory assigned to a thread can affect its behaviour. This means that if two threads share the same process, the criticality of the process is inherited from the highest criticality of the hosted functions, and therefore that every function hosted by this process shall inherit the criticality level of the highest criticality of its hosted functions.
- Cache memory: the cache memory has an important impact on the execution time of a function, especially on recent processors. Preempting a function may replace some of its lines of cache, requiring it to reload them when resuming its execution, and increasing its WCET. The Cache Related Preemption Delay scheduling and timing analysis model takes this into account.
- Local processor optimizations (pre-fetch, pipelines, etc.) tend to address average performances, which is detrimental to worst-case performances, while not being, or being poorly, documented.

The operating system can have a large impact on the actual duration of the execution of a function. General purpose operating systems tend to favour flexibility and average case performance, which is often detrimental to worst-case performance.

- Virtual memory: the virtual memory uses a swap file on a local drive to store memory pages when more memory is assigned than the available physical memory. Every real-time process should disable this functionality.
- General purpose OSes (GPOS) may have a long (or even unbounded) kernel latency. Under stress, the kernel may lock the processor during a long time, incompatible with any real-time constrained application. Real-time operating systems (RTOS) shall therefore be preferred, and when still resorting to a GPOS, real-time patch should be applied to reduce the impact of kernel latency.
- Some OS may automatically use energy saving functionalities (such as Dynamic Voltage Frequency Scaling) that can impact the WCET by slowing down a core, such functions shall be disabled or carefully controlled.
- Some operating systems offer the notion of process, which are a way to partition the memories of different processes. This can be helpful to improve the freedom from interference between the

data and instructions of two processes. Nevertheless, the timing interference of the processes sharing the same platform shall be carefully considered.

- Some operating systems offer the notion of partitions (e.g. ARINC 653), which ensure memory isolation as well as static processor sharing. This offers the safest way to isolate processes and to guaranty their freedom from interference.

Moreover, in multicore architectures:

- The front side bus is shared between the cores, as well as the access to memory banks. Except for some open architectures (like RISC V), the arbitration policy is not or only partially documented.
- Memory access requests are usually not meant to reduce worst-case response time of the memory but rather maximize the average throughput of the memory.
- In general, guaranteeing freedom from interference on a multicore platform is a tedious process, and the Cast-32A standard was a first attempt to identify ways to deal with it.
- Threads migrations between cores can generate high overhead (stress of the front side bus, the memory, and cache synchronization) and therefore shall be disabled or carefully controlled and accounted for.

5.6 Hardware-based Security for Drones

Similar to the emerging area of IoT, future drone systems also can be seen as distributed IoT devices. Particularly since drones are intended to move and reside in the environment (in comparison to industrial IoT that moves freely outside protected buildings) the wireless communication and integrity of the data must be secured similarly to other comparable wireless IoT applications. Therefore, this section provides a guideline for drone system integrators (derived from IoT-Security guidelines) about how their future drone systems could benefit from the integration of a hardware-based security chip into future drones. Various security attacks of recent years have indicated that the methodology of including hardware-based security modules into connected embedded systems (such as drones or general IoT devices) and the potential advantages are not yet widely known or used.

5.6.1 Risks of IoT and Connected Infrastructure

The risks of IoT mirror those of any networked computer system. However, because the IoT will impact many different sectors and have a role in controlling physical infrastructure and services, these risks are amplified. A successful attack on an IoT device or system can have significant impact on users, device manufacturers and service providers by affecting the physical as well as the cyber world. It may expose confidential information such as private user data as well as know-how, intellectual property and process intelligence. In addition, it can lead to interruption of operations, compromise of business continuity and even danger to a company’s brand image, success and very existence.

For policy makers, the principal concerns related to IoT risk mitigation are the protection of public safety and privacy. It is critical that networked systems controlling industrial and public infrastructure (this also includes future drone applications) are protected from both accidental and malicious attacks. Personal information about individuals that are monitored by IoT devices while going about their daily lives or using such devices to monitor their own property also must be protected both from accidental exposure or deliberate theft with intent to misuse.

Therefore, current research on future autonomous vehicles should not repeat mistakes from the early IoT area, such as smart homes. For example, the rush to the IoT for home monitoring and security appears to have outpaced principles of design for security. A vulnerability study conducted by security researchers in the summer of 2015[24] found serious security flaws in every one of nine internet-connected baby monitors it tested. The researchers noted that every camera had a backdoor that would allow

[24] Article: "Watch out, new parents – internet connected baby monitors are easy to hack", URL: https://splinternews.com/watch-out-new-parents-internet-connected-baby-monitors-1793850489

intruder access. Additional security flaws included the use of default passwords, easily accessed internet portals and lack of encryption. Hackers have created web sites featuring thousands of discovered insecure webcams for curious peepers. Consequently, drone system designs including security in a foresighted manner should be an important learning from various early rushed IoT developments which suffered from various attacks.

5.6.2 Methodologies for Drone Security driven from IoT Security Concepts and Best Practices

Security for the IoT revolves around three main concepts: confidentiality, identity, and integrity. These concepts can be expressed as questions:

- Is the transfer and storage of sensitive data protected?
- Are the components of the IoT system (device, server, etc.) what they claim to be or are they digitally disguised?
- Have the components been compromised or infected?

These typical threats – which typically purely software-based systems have been suffering in the early IoT area – are illustrated in Figure 19.

Therefore, the integration of a hardware-based security module (sometimes denoted as HSM, alternatively also denoted as Secure Element, SE) into the drone system design is proposed, acting as “Root of Trust”. A Root of Trust is the best way that these above-mentioned questions can be positively answered. The Root of Trust is a security chip hardened against attacks and integrated into the IoT device, network, or server. In general, depending on the intended application and used HSM chip variant, it can provide different levels of protection that fulfil some or even all of the roles for hardware security illustrated in the grey boxes in Figure 20. One specific feature-subset of the potential use cases have been used basis for the development of the novel concept “Security-enhanced TLS handshake supported by HSM”. In Figure 20 this the used feature-subset is highlighted green.

However, any drone system integrators should be aware that at least a few other use cases of hardware-security for drones would make sense, such as using a HSM for securing software/firmware updates (using protected signature checks), crucial OS-boot process protection or protection of special critical stored data.

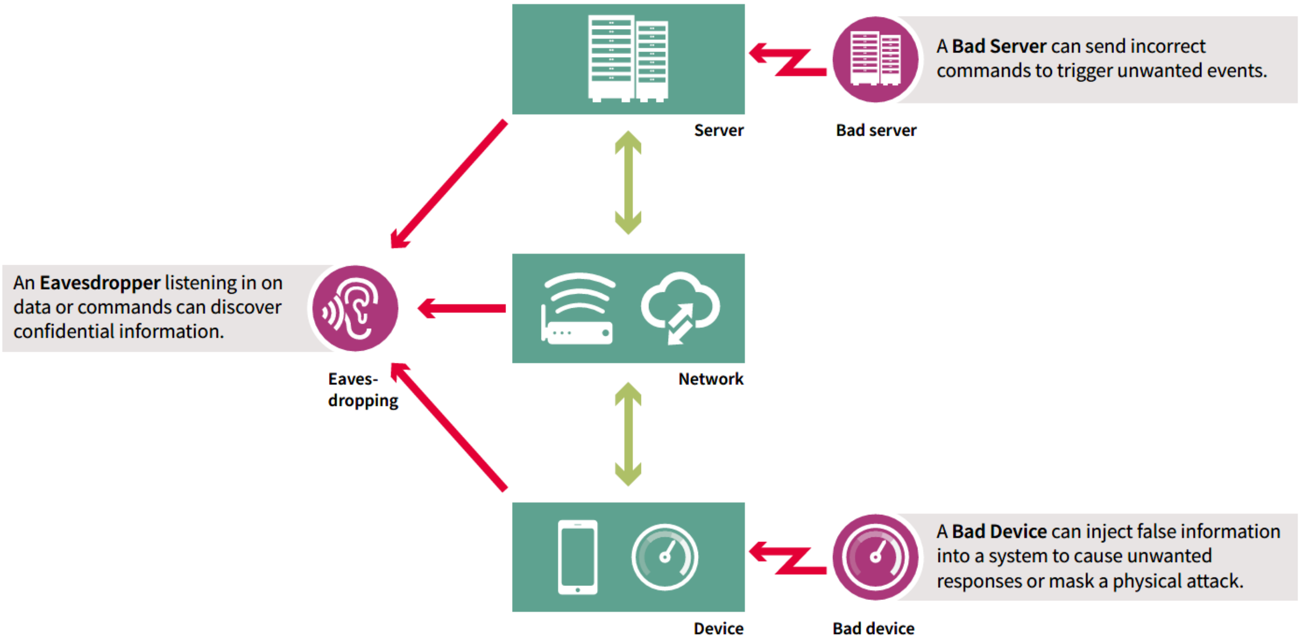

A Bad Server can send incorrect commands to trigger unwanted events.
Server
Bad server
An Eavesdropper listening in on data or commands can discover confidential information.
Eaves-dropping
Network
A Bad Device can inject false information into a system to cause unwanted responses or mask a physical attack.
Device
Bad device

Figure 19: Typical security threats for distributed connected systems (IoT or drones)

5.6.3 Guidelines based on Best Practices using a HSM to Protect Embedded Devices

The lowest level of risk in an IoT system may be a non-programmable end node that simply relays sensor data to some type of gateway or local server which verifies the source and includes the input in its operating data. Even at this level, a low-cost authentication chip with a single pre-programmed identity provides a way to confirm identity throughout the device lifecycle. This also helps to prevent the proliferation of cloned devices at the edge of the network. If there is a requirement that the transmitted data be encrypted or that the device be resold or reconfigured, additional protected storage of keys and certificates should be considered.

The data and commands that flow between devices and servers should be encrypted sufficiently to resist attempts at eavesdropping and false command injection. This requires cryptographic computation capability at both ends, which can be scaled to suit the level of risk.

Even at the lowest level of functionality, hardware-based security uses cryptographic mechanisms to protect secret data. The cryptographic algorithm can be implemented running on a general purpose MCU, but it is advisable for the devices themselves to have at least basic tamper-resistant capability and cryptographic functionality. Such protections are already widely implemented in chips such as those used in credit cards. These chips protect themselves and can even automatically erase their memory if tampering is detected.

Similar to IoT devices, also drone security benefits from a holistic approach that provides for security throughout the lifecycle of every device used in the system. In systems that use large numbers of low-cost devices, secure hardware supply chains support shipping chips directly from the chip manufacturer to the point of assembly. With a preprogrammed identity, the chips then can register themselves “over the air” when turned on. It is easier to defend against intrusion and subversion if each device is fitted with a security key at a central point of control.

All these techniques (tamper-resistant circuits, authentication, and encryption) have been used previously in other systems but are not yet routinely considered for drones. We believe the benefits of hardware-based security (including better performance, improved security (including tamper resistance), and security partitioning (protection against bugs in operating system and application code)) make a strong case for using this technology for improving the security of future drones. This particularly applies for long-time autonomous operated drone use cases, when potential security attacks might keep being undetected.

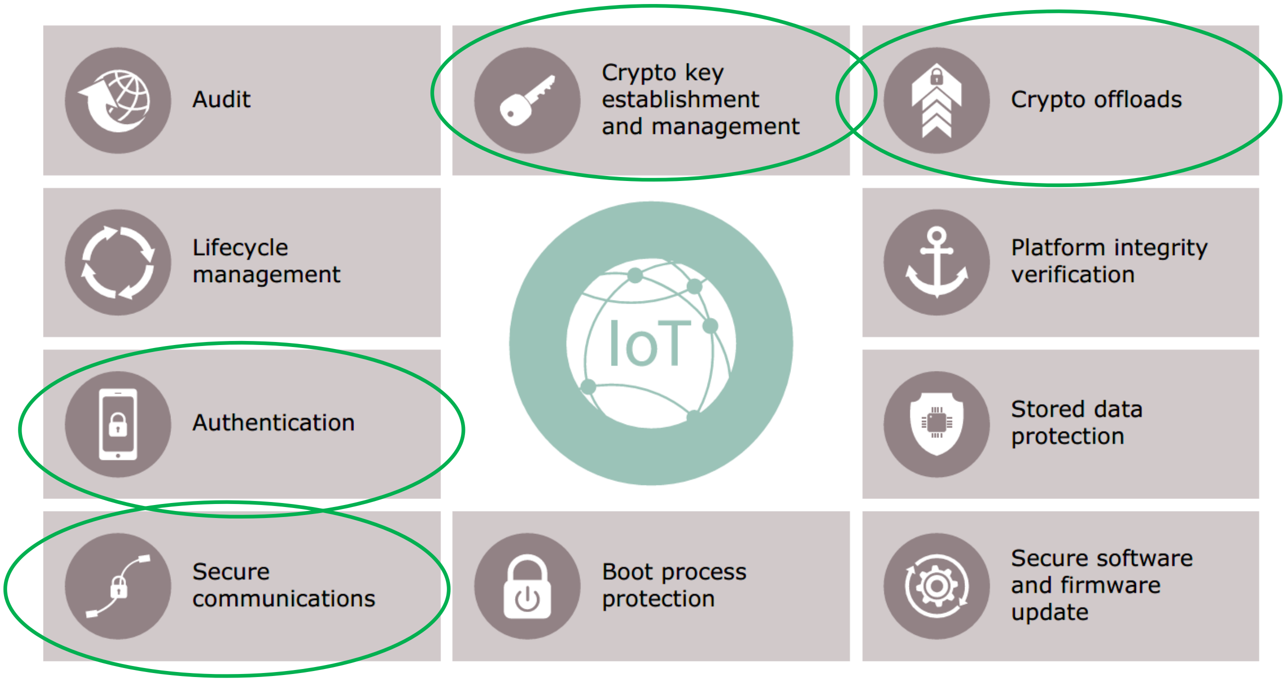


Figure 20: Various use cases for chip-based hardware security

5.7 Development of Specific Drone Features

The drone system includes different features. To ease the development of these features, in this section, we give guidelines to follow in developing specific features such as: communication infrastructure, video analysis, object detection, and mission validation.

5.7.1 Development of a Communication Infrastructure

Drone system refers both to the drone itself as a system, and to a system in which the drone operates. In this section we refer to the second definition of drone system. In particular, we refer to a system that includes different drones in terms of UAV and UGV, but also in terms of different kind of UAVs or UGVs, that can work in a more complex environment where sensors on the field can still play a role. In this case, we can consider the drone (both UAV and UGV) as a part of a cyber-physical system, playing both the role of sensor and actuator.

The traditional utilization of a drone considers gathering information with the drone and analysing the data off-line, after the landing. In a context such as the one of cyber-physical systems such a limited utilization can be consider inefficient, since it prevents the possibility of prompt actions and reaction, and it does not exploit the potentiality of such complex systems.

Thus, drones are required to incorporate more capabilities in terms of storage, processing and energy efficiency but also to be able to cooperate with other components distributed in the systems, that might be different drones in terms of operation (aerial or terrestrial) or in terms of architecture (different hardware).

In such a scenario, a modular and flexible communication infrastructure is necessary to guarantee the interaction among the different components, as well as a unified visualization of data arriving from the different sources, a dashboard highly customizable for non-drone expert. This system has to be able to see the drones as sensors and to treat them as plug and play components. This requires it to:

a) Provide bidirectional communication;
b) Adapt to the communication specifications given by the drone;
c) Communicate with any additional processing module embedded on the drone;

d) Support communication with any additional sensor in the environment;
e) Handle data transfer from the different sources;
f) Visualize in a unified way the data from the different sources;
g) Manage the different components from a unique user-controller panel;
h) Support processing in the communication nodes.

These requirements can be fulfilled with a communication infrastructure that acts as integration layer among different components in the system and as an orchestration actor. Its implementation requires distributed blocks that operate both at the edge and on the cloud:

1. One or more blocks at the edge, able to communicate with different components that use different communication protocols and a user-custom interface to offer flexibility and adaptation to the heterogeneity of the system (linked to requirements a, b, c, d, e);
2. One user-friendly and customizable interface, connected to the cloud, to monitor and manage the whole system from wherever the operator is (linked to requirements a, e, f, g);
3. Models and modules for computation at the edge, to distribute computation in the system and in the nodes of the communication infrastructure (linked to requirements h).

5.7.2 Development of Video Content Analysis

A drone can be considered as a mobile monitoring tool, able to acquire and process different types of information, according to the sensors installed on it. Therefore, it can be integrated within a traditional video surveillance system, composed by fixed cameras installed on the ground.

Among drones' sensors, cameras play a crucial role enabling multispectral images and video flows acquisition (e.g., visible thermal, infrared etc.) according to the types of cameras used. Moreover, thanks to Video Content Analysis (VCA), that is a technology for image and video processing able to extract relevant information, a drone moves from a traditional set of sensors into a smart monitoring tool. It becomes able to automatically detect events of interest and/or relevant targets focusing on people, vehicles, plants, animals, static objects, etc.

There are different approaches for video analysis. Traditional video analysis algorithms are based on background estimation. They separate static background and moving targets, the foreground, by comparing the video flow, frame-by-frame. The result of this pre-processing is a set of metadata describing the characteristics of detected targets (position, sizes, direction, speed, etc.). Notification messages and/or alarms are generated when such characteristics meet predefined rules that define the events of interest to be detected.

Recently, innovative VCA systems are based on artificial intelligence approaches and in particular on deep learning (e.g., Convolutional Neural Network (CNN)). Such algorithms should be trained, to learn how to perform the detection. This phase is based on a training set that consists of a large number of annotated images given to the Neural Network (NN) as input. To provide a concrete example, a neural network will be used to detect grapes in the smart agriculture use case. The training set is a collection of thousand images of grapes. The NN going through this training set to "learn" how to recognize a grape. Once the algorithm (i.e., the neural network) has been trained properly, it can be deployed on the HW that will be in charge of VCA execution. In particular, in this considered scenario, there will be some possible alternatives: video content analysis can be deployed and executed on the drone itself or on ground station or in a dedicated hardware connected to the ground station, like for example a Network Video Recorder - NVR (see the figure below).

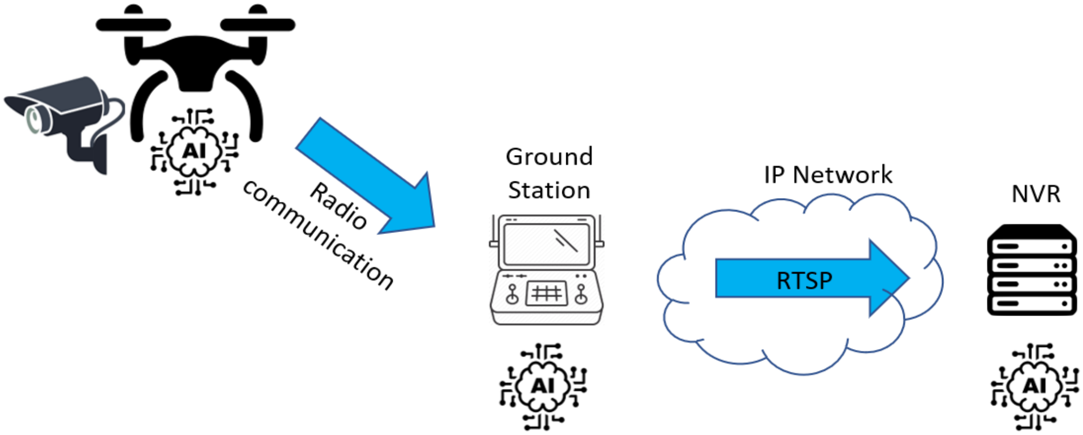


Figure 21: VCA systems are based on artificial intelligence approaches

Each solution has pros and cons. VCA algorithms, particularly those based on AI, are usually resource-intensive, and this can be a limitation for an onboard execution. An optimal compromise between algorithm complexity and detection performance is needed to adopt this configuration. If it is possible, a drone can acquire video in HD, process it locally and stream in real-time only the extracted metadata. Moreover, advanced mechanism can be implemented: in normal conditions the drone transmits low quality video streams; when a relevant event is detected, the drone automatically switches the video transmission to high quality also sending an alert message to the ground station. In this way, a human operator can monitor the situation checking the HD video after receiving an alarm message when a relevant event has been detected.

Processing on ground station or on a dedicated NVR, requires the drone to stream continuously an HD video flow, with a very limited delay if a real-time detection is needed. Obviously, there is no resource limitation constraints in this case, as the HW can be properly sized according to the specific requirements. This configuration cannot be used when the output of the VCA should guide decisions and actuations autonomously taken by the drone.

It is worth noticing that the discussed VCA approaches belong to the payload segment and can be used for business cases and applications. In case of safety related functions, the use of AI must be reconsidered. In particular, certification of machine learning models is one of the main goals of AI in the near future. The eXplainable AI, an extremely relevant hot topic nowadays, may drive this certification process.

5.7.3 Development of Object Detection Component

Object detection is a computer vision technique that works to identify and locate objects within an image or video. Specifically, object detection draws bounding boxes around these detected objects, which allow us to locate where said objects are in (or how they move through) a given scene.

Besides conventional post-data processing system, innovative ways are used in extracting information. Machine learning and deep learning are an arising approach in dealing with large amount of data gained from drones[25]. For infrastructure planning and design, typical data acquired through drones are images. 3D geometrical models can be generated from these images through manually method or semi-automated algorithms. For construction monitoring, either real time videos or 3D models are needed. As for infrastructure inspection, machine learning and deep learning algorithms are employed.

Focusing on object recognition and detection in aerial images captured by drones, a major challenge with the integration of artificial intelligence and machine learning with autonomous drones' operations is

[25] Shakhatreh, H., Sawalmeh, A., Al-Fuqaha, A., Dou, Z., Almaita, E., Khalil, I., Othman, N. S., Khreishah, A., and Guizani, M., "Unmanned aerial vehicles: A survey on civil applications and key research challenges." arXiv preprint arXiv:1805.00881.

that these tasks are not executable in real-time or near-real-time due to the complexities of these tasks and their computational costs. The most accurate modern neural networks do not operate in real time and require large number of GPUs for training with a large mini-batch-size. The proposed solution, and already implemented in **COMP4DRONES** project, is the implementation of a deep learning-based software which uses a convolutional neural network algorithm to track, detect, and classify objects from raw data. In the last few years, deep convolutional neural networks have shown to be a reliable approach for image object detection and classification due to their relatively high accuracy and speed[26]. Furthermore, a CNN algorithm enables UAVs to convert object information from immediate environment into abstract information that can be interpreted by machines without human interference. The main advantage of CNN algorithms is that they can detect and classify objects while being computationally less expensive and superior in performance when compared with other machine-learning methods.

Following this approach, for the object detection, the computer vision components need to be ingested with images captured by the drone. The images will be mainly capture either by a RGB or by a LiDAR camera and once the flight has been finished, images are sent to the ground station to be processed.

Once the images are ingested by the component, and thanks to the CNN algorithms that has been previously trained for the detection of objects in a particular domain, a relation of the detected object, the status and/or the position can be obtained. Talking about application domains, object detection can be applied to the following:

- Crowd counting
- Self-driving cars
- Video surveillance
- Face detection
- Anomaly detection

Of course, this is not an exhaustive list, but it includes some of the primary ways in which object detection can be applied.

5.7.4 Drone Missions Validation

Validation (and testing) is a major step towards the deployment of technological solutions in the real world. The design, development and implementation of cyber-physical systems is error prone and affected by many factors ranging from those more related to the management aspects (composition and number of the involved teams/people, their base knowledge and attitude, the used tools, etc.), to those more related with the technical aspects (inherent complexity, size, performances, etc.). Thus, standardized workflows (V-cycle) are basically mandatory and the validation is one of their fundamental steps. Usually, validating a design or an implementation consists in running the developed system in a well-known environment such that its behaviour can be checked under controlled conditions. In drone applications, the validation of missions can ease their development process. Indeed, it involves running such missions in simulated environments and into simulated physical models such that any failure that would otherwise happen in the real world is not dramatically impacting on the development costs and on the people safety. Further, all the effort that would be required in order to deploy the algorithms into the real drones is absent.

One way to achieve the validation of drone missions is to describe them with the help of Signal Temporal Logic (STL) formulas which both highlights any inconsistency or logical contradiction and provides the first step for its casting into a robust optimization problem. This allows to get trajectories complying with the mission requirements and specifications, and that can be tested by means of any known simulator (e.g., Gazebo), thus providing useful feedback to the designer.

[26] Sherrah, J. Fully Convolutional Networks for Dense Semantic Labelling of High-Resolution Aerial Imagery. Available online: https://arxiv.org/pdf/1606.02585.pdf (accessed on 8 June 2017).

6 Drone System Development Methodology

During the 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 RobMoSys[27]. 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 RobMoSys, system composition requires a structure. This structure has requirements originating from three perspectives (see Figure 22). First, composability is the ability of building blocks to be combined and recombined into different compositions. Second, 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.

In **COMP4DRONES**, we also follow such compositional approach. Thus, in this section, we will describe the **COMP4DRONES** methodology by adopting/enhancing the methodology that has been proposed in the RobMoSys project. We start by describing the composability concept followed by the composition structure (i.e., the reference architecture). Then, we describe the composition workflow, while the tools supporting the methodology are described in the Section 0.

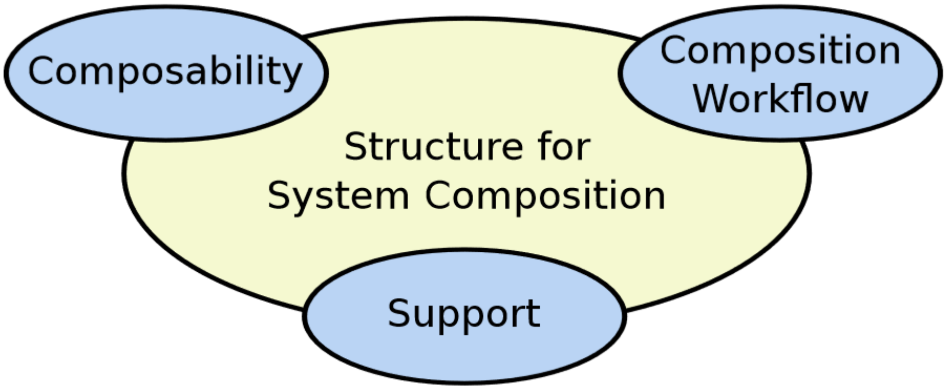


Figure 22: A structure for system composition has requirements originating from composability, composition workflow, and support via tooling[28]

6.1 Composability Concepts

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 requirements[29].

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 syntax

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

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

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

and semantic plus application and technical level perspectives to enable composability for system composition.

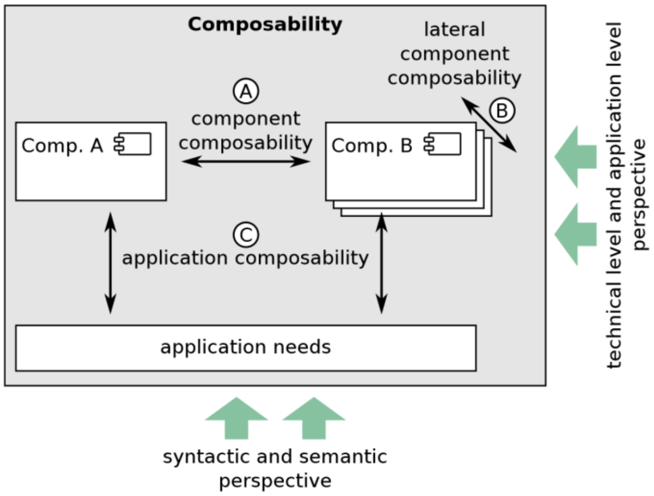

Composability
lateral component composability
Ⓐ
component composability
Ⓑ
Comp. A
Comp. B
Ⓒ
application composability
application needs
technical level and application level perspective
syntactic and semantic perspective

Figure 23: Occurrence of composability[30]

6.2 Structure for System Composition: The Reference Architecture

The reference architecture proposed by **COMP4DRONES** project is presented in Figure 24 (more details can be found in the deliverable D3.2). In Figure 24, the different blocks of UAS and the interactions between them are presented. The architecture is divided into four main clusters: flight navigation, flight control, flight management, and mission management. First, the flight navigation includes the drone perception to gather information needed to navigate the drone from one location to another while avoiding obstacles and preserving the geo-fence using the flight guidance. Second, the flight control executes the guidance commands to fly the drone from one place to another through drone actuation. It also executes commands coming from the pilot directly. Third, the flight management contains functions for planning the flight trajectory and managing the UAV payload, data, and health. Fourth, the mission management have the mission planning and supervision functionality that are managed by the mission manager. Finally, there are external services that provide information for mission and flight planners to plan a valid mission/trajectory.

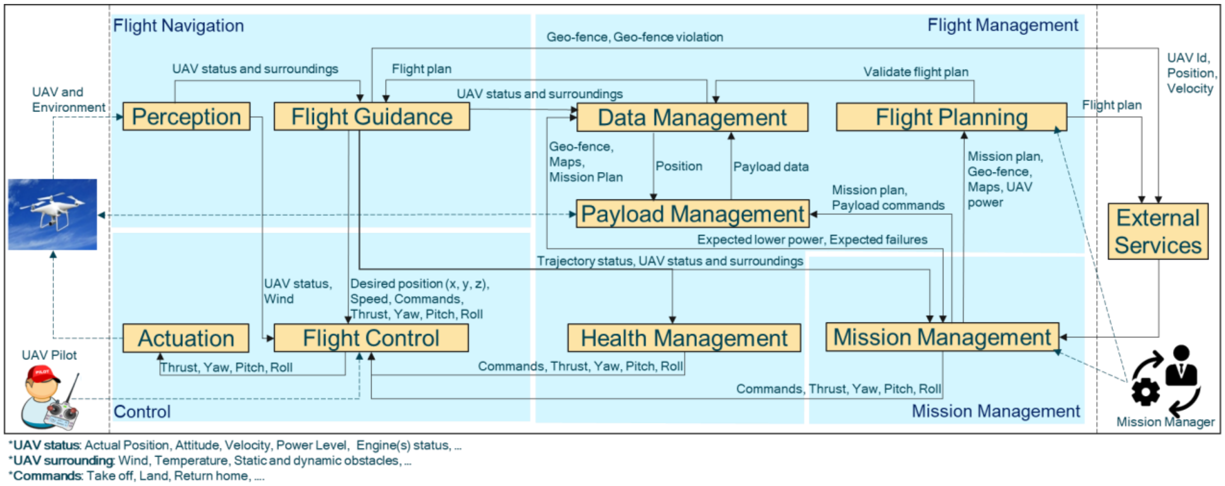

Flight Navigation
Flight Management
Geo-fence, Geo-fence violation
UAV status and surroundings
Flight plan
UAV status and surroundings
Validate flight plan
UAV Id, Position, Velocity
UAV and Environment
Perception
Flight Guidance
Data Management
Flight Planning
Flight plan
Geo-fence, Maps, Mission Plan
Position
Payload data
Mission plan, Geo-fence, Maps, UAV power
Mission plan, Payload commands
Payload Management
External Services
Expected lower power, Expected failures
Trajectory status, UAV status and surroundings
UAV status, Wind
Desired position (x, y, z), Speed, Commands, Thrust, Yaw, Pitch, Roll
Actuation
Flight Control
Health Management
Mission Management
UAV Pilot
Thrust, Yaw, Pitch, Roll
Commands, Thrust, Yaw, Pitch, Roll
Commands, Thrust, Yaw, Pitch, Roll
Control
Mission Management
Mission Manager
*UAV status: Actual Position, Attitude, Velocity, Power Level, Engine(s) status, ...
*UAV surrounding: Wind, Temperature, Static and dynamic obstacles, ...
*Commands: Take off, Land, Return home, ...

Figure 24: Overall drone system architecture – flat view

[30] https://mediatum.ub.tum.de/doc/1399658/document.pdf

6.3 Composition Workflow: Re-use based Agile Development

The composition workflow is the activity of putting together the building blocks. The workflow defines the steps and the order to bring together all participants. It addresses their individual needs for system composition (see Figure 25). 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.

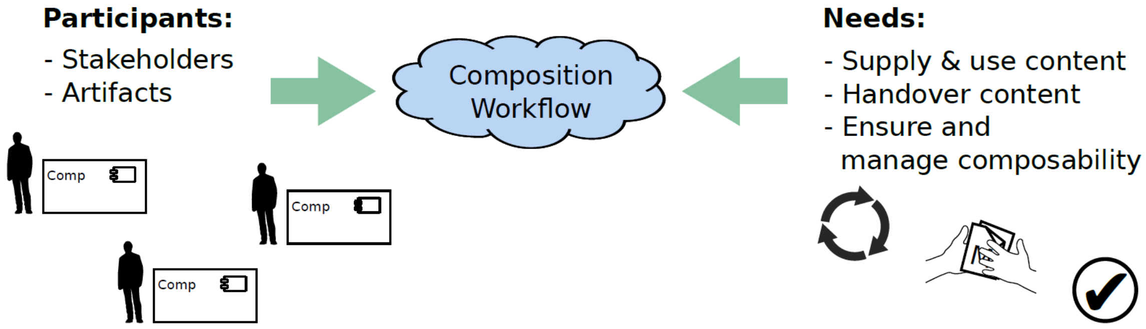


Figure 25: The composition workflow[28]

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 (e.g., Scrum and Unified Process), or other methodologies (e.g., Software Product Line (SPL)) can be applied within these steps to develop building blocks.

Finding a workflow that defines and uses a structure requires to understand its stakeholders. The two main stakeholders in the ecosystem (see Figure 26) 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 structure (i.e., the reference architecture). They can work within clear boundaries and their building blocks can connect through the defined interfaces.

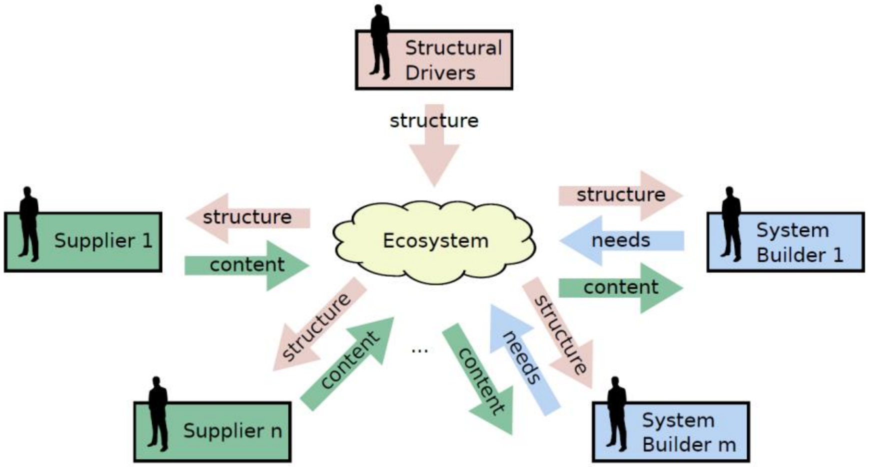
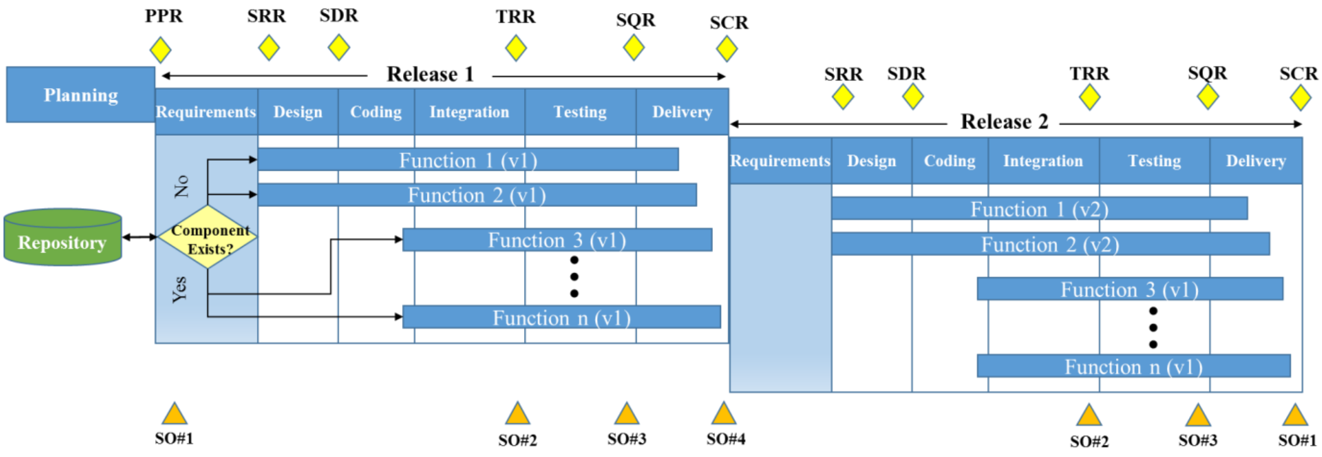


Figure 26: Stakeholders collaborating and interacting in an ecosystem[28]

In the **COMP4DRONES** project, a reuse-based agile development process is going to be followed as shown in Figure 27. 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 (technology selection process is described in Section 6.3.1). 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 27). 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 certificate[31] (that contains all information on the dependability attached to the reusable components).



PPR: Project Planning Review
SCR: Software Coding Review
SDR: Software Delivery Review
SRR: Software Requirements Review
SIR: Software Integration Review
SOI#: Stages of Involvements
SDR: Software Design Review
STR: Software Testing Review

Figure 27: Reuse-based agile development process

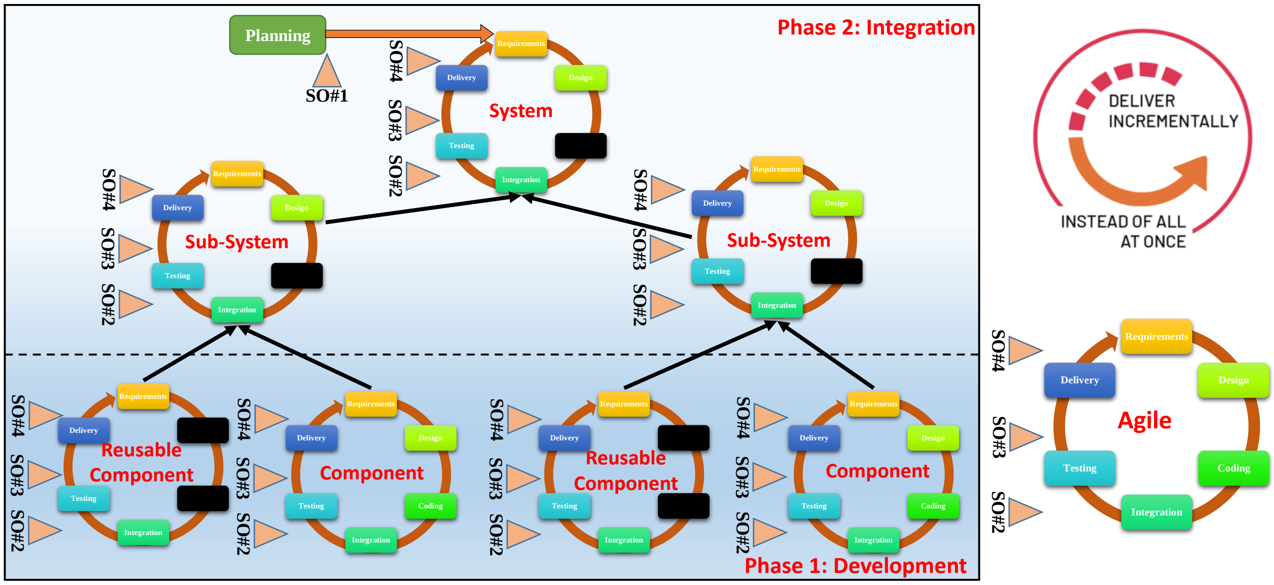


Figure 28: Reuse-based agile development process workflow

[31] 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.

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 28. 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 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.

6.3.1 Technologies Selection Process

Figure 29 shows the four main steps shall be carried out sequentially in order to identify the appropriate technologies based on the ConOps derived from the system requirements:

1. Key technologies COTS analysis
2. Technology evaluation criteria definition
3. Technology evaluation
4. Technology selection

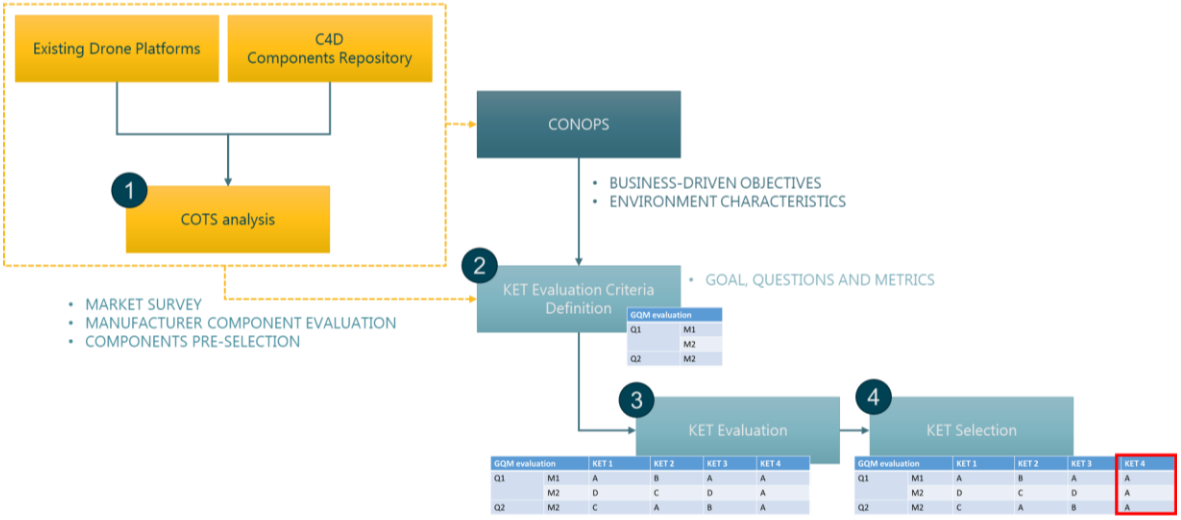
Step 1: Key Technologies Cots Analysis

The Commercial Off-The-Shelf analysis aims to screen for existing technologies that could fulfil the needs expressed in the ConOps. It shall be conducted following the three axes below:

C4D KET Repository (consultation of drone platforms components developed in the **COMP4DRONES** project). **C4D** partners have produced a matrix listing all the key enabling technologies (KET) developed through the use cases. Most of the technologies developed may be relevant to any future system development for drone applications and multiple use cases. Therefore, the review of the **C4D** KET repository shall be the first step in the COTS analysis.

Market Survey: A market survey shall be conducted in order to identify all commercially available components/technologies that may potentially be integrated to the system to be developed. All relevant components or KET shall be listed in the **C4D** Market Survey Template form (Table 15), together with their manufacturers and/or owners.

Component Pre-selection (i.e., components' comparison and pre-selection). Depending on the component/technology availability, price, interoperability or any other typical market survey criteria, a relevance index shall be indicated in front of each item in the **C4D** Market Survey Template form. The comparison of the listed KET relevance index will guide the system-of-interest (SOI) development team through the first step of the component/technology pre-selection process.


Existing Drone Platforms
C4D Components Repository
1
COTS analysis
CONOPS
• BUSINESS-DRIVEN OBJECTIVES
• ENVIRONMENT CHARACTERISTICS
2
KET Evaluation Criteria Definition
• GOAL, QUESTIONS AND METRICS
• MARKET SURVEY
• MANUFACTURER COMPONENT EVALUATION
• COMPONENTS PRE-SELECTION
3
KET Evaluation
4
KET Selection

GQM evaluation	
Q1	M1
	M2
Q2	M2

GQM evaluation		KET 1	KET 2	KET 3	KET 4
Q1	M1	A	B	A	A
	M2	D	C	D	A
Q2	M2	C	A	B	A

GQM evaluation		KET 1	KET 2	KET 3	KET 4
Q1	M1	A	B	A	A
	M2	D	C	D	A
Q2	M2	C	A	B	A

Figure 29: Identifying technologies based on ConOps

Step 2: Technology Evaluation Criteria Definition

This part focuses on the definition (with the Goal, Questions and Metrics approach) of KET evaluation criteria that are consistent with the SOI expressed needs and environment defined through the ConOps.

Goals Questions and Metrics. Definition of criteria based on business-driven goals and environment characteristics provided by the ConOps and the previous COTS analysis. The Goal, Question and Metrics approach allows the definition of KPIs directly related to the SOI objectives. Indeed, the review of the goals stated in the SOI ConOps and COTS analysis allows the development team to define a set of questions to be answered in order to evaluate whether or not the system meets its objectives.

Metrics will then be specifically defined to provide quantified answers to this set of questions, which are the KET evaluation criteria.

Weighing KET Evaluation Criteria. Each of the KPIs specified in the previous step shall be titled and weighted according to their importance.

Technology Evaluation Form. Creation of a formatted matrix displaying the pre-selected components/technologies, the evaluation criteria and their weight. The **C4D** Technology Evaluation Form (Table 16) is a valuable decision-making tool toward the selection of a KET.

Table 15: C4D market survey template

KET	Manufacturer/Distributor	Product/Component	Application	Relevance Index (A/B/C/D)	Comments	Component pre-selection (Y/N)
KET 1	Name 1 / Name 2	Product name 1	UC X	A		Y
KET 2	Name 3 / Name 4	Product name 2	UC X	B		N

Table 16: C4D technology evaluation form

	Questions	Metrics	KET 1	KET 2	KET 3	KET 4
Communication Between Operators	What type of communication is used between operators?	KPI: Data volume transfer ability KPI: Data type transfer ability KPI: Data transfer medium availability	20%	25%	43%	43%
	How many users communicate simultaneously?	KPI: Number of users per communication channel	10%	10%	46%	23%
	Is an exterior service provider involved?	KPI 3:	50%	53%	23%	66%
	Is the channel secure and safe to use?	Frequency bandwidth				
Goal 2	Question 1	KPI 4	95%	99%	95%	92%
	Question 2	KPI 3	32%	23%	96%	64%
	Question 3	KPI 5	25%	16%	87%	21%

Step 3: Technology Evaluation

Once the technology evaluation criteria are well defined for the demonstrator/use case/prototype, actual data shall be collected for each pre-selected technology. A weighted score may then be associated to each criterion and displayed in the **C4D** Technology Evaluation Form by the following:

- Data Collection. Business and operational performance data shall be collected for each of the pre-selected KET.
- Data Analysis and Results Generation. The collected data shall be analysed and run against the KET evaluation criteria. Results shall be added to the C4D Technology Evaluation Form.
- Technology Evaluation Form. Completion of the C4D Technology Evaluation Form in order to display the weighted score in front of each criterion.

Step 4: Technology Selection

A comparative study of the technologies listed in the compiled **C4D** Technology Evaluation Form may finally be conducted. A total score for each KET shall be calculated and a final decision regarding the most appropriate technologies to be selected in order to fulfil the need may be taken.

6.3.2 System Design

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 phase[32]. 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: a) a modeling language, b) a modelling methodology, and c) a modelling framework that implements the modeling languages, preferably customized to support the development methodology.

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, tool supported and disseminated modeling language[33]. 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, behaviour allocations, and constraints to support various engineering analyses. SysML can be itself further specialized using profiles to cope with a specific methodology depending on the domain or application.

6.3.3 Implementation and Technologies Integration

A reuse-based implementation of a drone system is performed through integrating existing technologies. However, sometime not all required components exist and then they need to be developed from scratch. Thus, there are two scenarios in the implementation phase: component development, and system integration (see Figure 30).

First, the component development is the part of the overall development process where the in-house operational software that is needed by the system is created. The components specify and implement the required variability to fulfil expected system requirements. The components are typically large and resemble object-orientated frameworks more than the traditional classes in object-oriented systems. The resultant components can either be a part of the core assets of the repository, or they can be developed for mission specific reasons. Figure 30 shows an overview of the activities and artifacts leading up to component design and implementation.

Second, software system integration is the practice of combining software components and subsystems into an integrated whole. There are two major models for software system integration, the waterfall model and the incremental model. In the waterfall model, integration is a discrete step towards the end of the development cycle. In the incremental model, on the other hand, system integration is a continuous ongoing activity, where components and subsystems are integrated as they are developed and form subsequent versions of the system as a whole. An incremental system integration model is usually preferred compared to a waterfall model since it decreases the risk of experiencing complex integration problems at the end of the developing cycle.

[32] Incose, Incose Systems Engineering Handbook V4, John Wiley and Sons, 2015.

[33] 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.

The effort needed for system integration vary over a spectrum. On one end of the spectrum, systems need a considerable integration effort such as coding component wrappers or actually developing new components to fulfil product requirements. At the other end of the spectrum, system can be built almost automatically by providing specific parameters to a construction tool and launching it. However, most system integration processes occupy the middle of this spectrum[34].

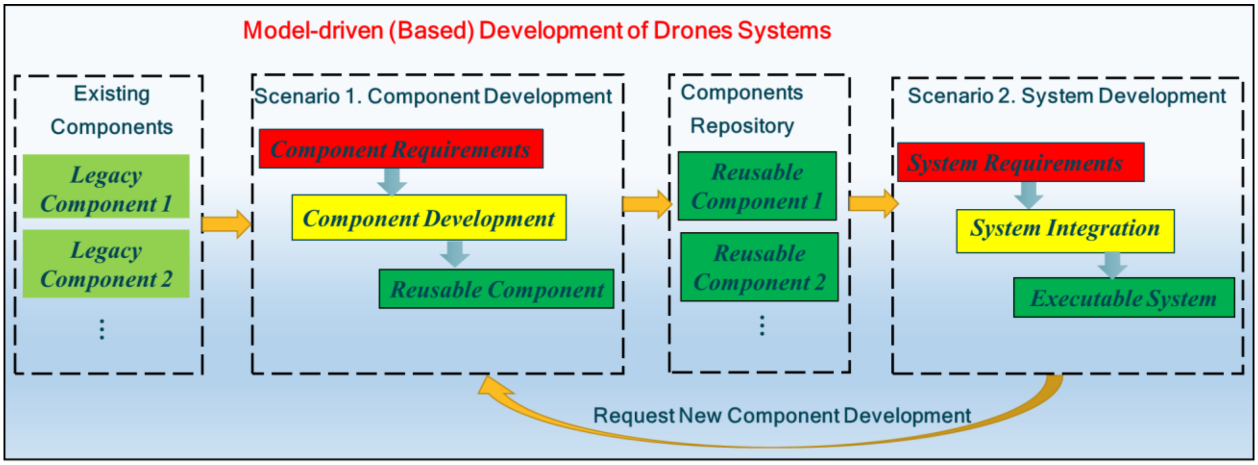


Figure 30: Component development and system integration scenarios

6.3.4 System Verification and Qualification

Two major activities shall be conducted in order to validate, integrate, verify, and qualify any drone system: the definition of an Integration Verification Validation Qualification (IVVQ) Plan, and the production of an IVVQ Summary. The content expected for both of these activities is defined hereafter.

6.3.4.1 Expected content of the IVVQ Plan

The IVVQ Plan document presents an overview of the system-of-interest (SOI) and indicates the objectives in terms of validation, integration, verification and qualification. It describes the general IVVQ process and a detailed view of the activities to be performed all along the SOI development cycle. The IVVQ Plan shall contain the following sections:

Objectives. The objectives section of the drone system IVVQ Plan shall remind the reader of the purpose of the document, introduce the global IVVQ strategy, methodology and processes, and shall state the objectives of the validation, integration and verification processes.

Overview of the drone system. The overview of the drone system section shall present a description of the drone system, sub-systems or components the IVVQ plan applies to. For the understanding of the reader, only the operational and functional elements of the system of interest shall be described in this section.

IVVQ applicability. The IVVQ applicability section of the drone system shall clearly identify the drone system, sub-systems or components the IVVQ plan applies to.

Organization and responsibilities. The organization and responsibilities section shall present the different actors involved in the IVVQ activities and detail their roles and responsibilities.

Global IVVQ process description. The global IVVQ process description section shall identify the objectives and provide a description of the key activities related to each of the integration, validation and verification phases of the IVVQ Plan. The description of the IVVQ activities shall contain their name and description, the list of associated action(s) to be completed, the procedure(s) to follow, the people/group of people/organization responsible for the completion of the action(s), the configuration item(s) they

[34] https://resources.sei.cmu.edu/asset_files/WhitePaper/2012_019_001_495381.pdf

apply to, the inputs needed as well as the expected outputs after the action(s) have been completed. While the methods and tools supporting the realization of the IVVQ activities are mentioned in this section, their complete description shall be presented in the methods and tools section of the IVVQ Plan. Key activities are user requirements validation, system requirements validation, etc.

Methods and tools. The global IVVQ process description section shall present a concise view of the methods and tools used in the frame of the IVVQ plan. A title, a description and a link to the IVVQ activity for which each method or tool is used shall be documented in this section.

Deliverables. The deliverables section shall list and present the templates of deliverables to be provided following the completion of the IVVQ plan activities.

Milestones. The milestone section shall describe the IVVQ activities to be completed and the associated deliverables to be provided for each major system development milestones.

Qualification. The qualification section shall introduce the qualifications currencies to be precise by the system itself, along with its operators (remote pilots and ground crew members) and users (customers, clients and participants) in accordance with the local EU regulations.

Other. Any other elements necessary to the understanding of the IVVQ plan shall be added in different sections: glossary of terms, definitions, applicable and referenced documents, appendix, etc.

6.3.4.2 Expected content of the IVVQ Summary

The IVVQ summary presents the activities performed in the scope of the IVVQ plan, displays the obtained results, states any relevant issues and draws conclusions on the maturity of the SOI.

A provisional version of the document shall be delivered for each of the major system development milestones and a final version shall delivered at the end of a development cycle. The IVVQ summary shall contain the following sections:

- Overview of the IVVQ process applied to the drone system
- Presentation of the IVVQ activities performed:
  - Objectives
  - Description of the activities
  - Applicable configuration items
  - Actors involved in the activities
  - Means
  - Procedures
  - Results
  - Evaluation of the results
- Presentation of the issues and deviations
  - Problem reports management
  - Open problem reports and categorization
  - Deviations to the IVVQ plan
- Operational limitations
- Currencies (remote pilot and ground crew)
- Conclusions

7 Supporting Tools

Methodology 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 31. 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. In the following, we describe the **C4D** tools.

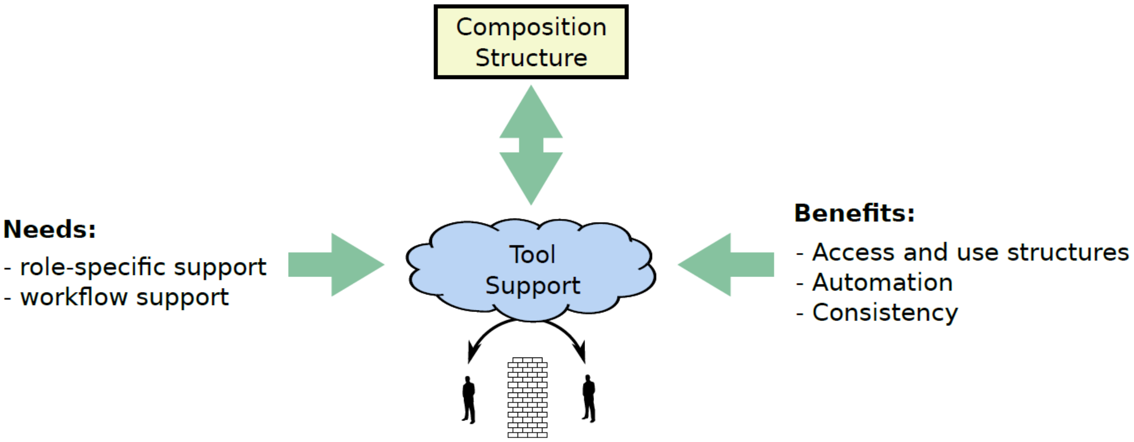


Figure 31: Adequate support via tooling for participants is critical towards system composition[28]

7.1 C4D Tools Workflow

One of the specific objectives of the project is to minimize the design and verification effort for complex drone applications. To achieve this goal, an engineering framework, the **C4D** Design and Verification Framework is being developed in WP6. In this section, the first version of the framework is provided. The framework contains the tools to be used by the use cases in order to facilitate their development. Each tool improves a specific aspect of the design process for drone-based services, facilitating concrete design and verification steps. The **C4D** Design and Verification Framework covers all the steps in the V-Cycle for mechatronic systems as shown in Figure 32.

Of course, the set of tools to be developed in **C4D** do not have the ambition to be self-sufficient. The **C4D** Design Framework will integrate commercial and open-source, third-party tools in order to complete a concrete design flow in a company. Most tools are general-purpose and can be applied to different steps in different projects. In some cases, the tools address a specific design problem and cannot be used out of it. Regarding its focus, some tools have a holistic point of view, enabling the analysis and design of the whole service. Other tools focus on specific aspects or sub-systems. Each tool operates using its own internal representation of the system. In some cases, they use certain standards such as UML, SystemC, Signal Temporal Logic, or Matlab.

Implementing a complex service based on drones requires a complex design process where modeling becomes a fundamental design task. Based on models, the system engineer can compose the system architecture so that design tasks such as integration of new functions, analysis of the system as a whole and of its components, multi-level system simulation, optimization, validation, verification and finally, implementation and deployment can be carried out with less effort, more quality, in a shorter time. The conceptual diagram in Figure 33 shows the tools under development in WP6 and their interrelation.

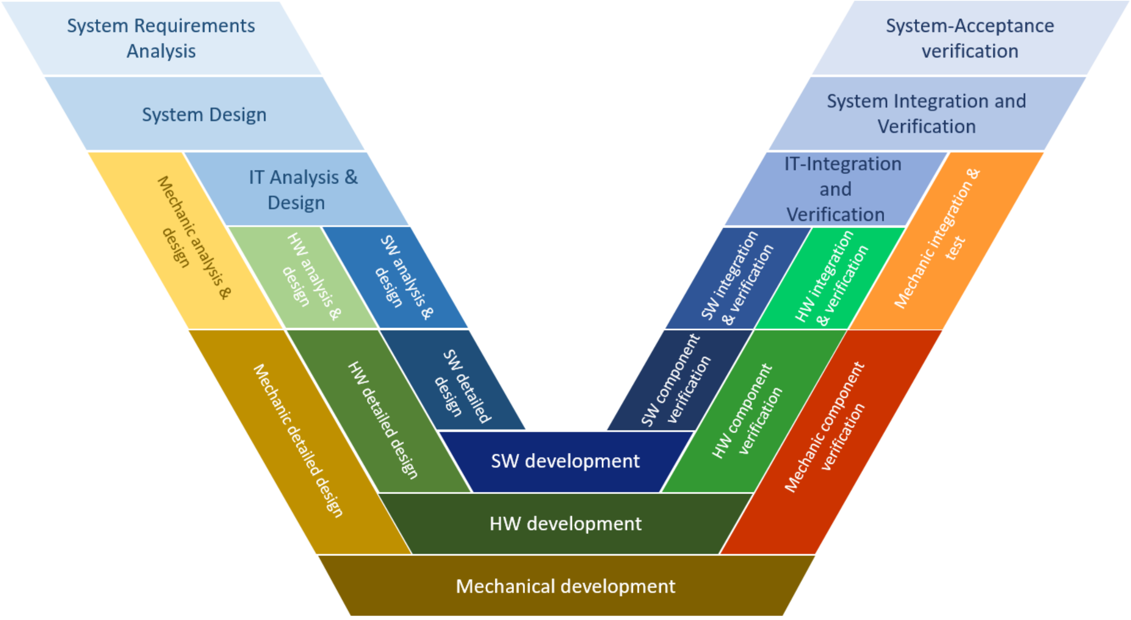

System Requirements Analysis
System Design
IT Analysis & Design
Mechanic analysis & design
HW analysis & design
SW analysis & design
Mechanic detailed design
HW detailed design
SW detailed design
SW development
HW development
Mechanical development
System-Acceptance verification
System Integration and Verification
IT-Integration and Verification
SW integration & verification
HW integration & verification
Mechanic integration & test
SW component verification
HW component verification
Mechanic component verification

Figure 32: V-cycle for mechatronic systems

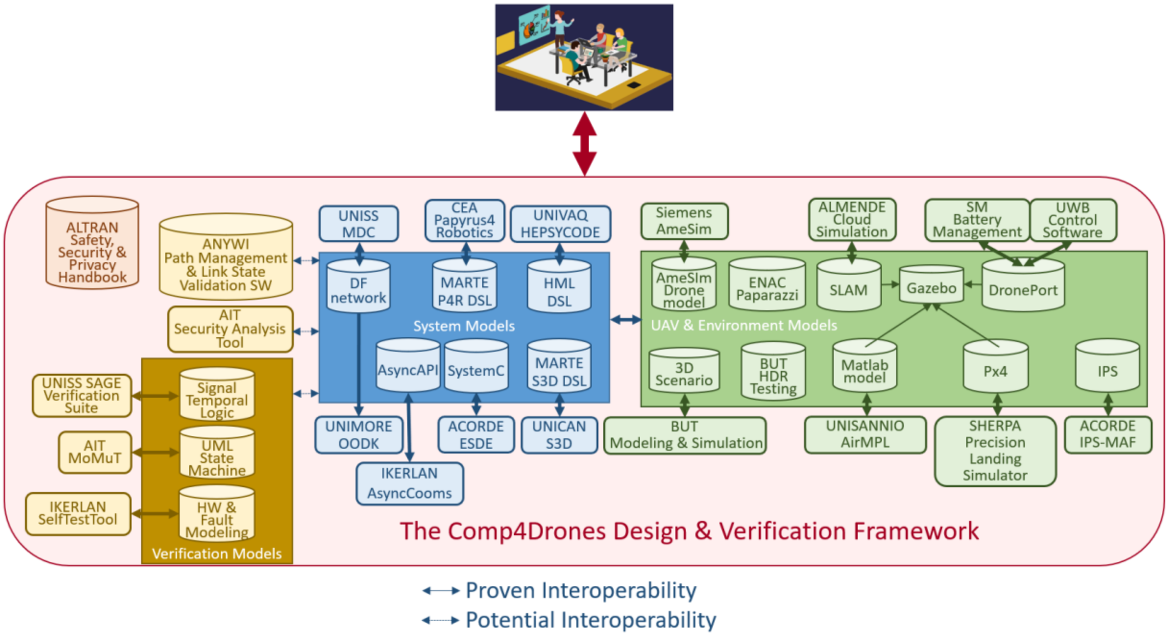


Figure 33: Conceptual diagram of the C4D design and verification framework

In the project, three kinds of models have been identified. The first is system models capturing system under design, even, the whole service for which the system operates. The modeling languages used are UML, UML/MARTE, SystemC and SDL DSL. The second is tools making use of models for drones and the environment they operate in. Most of these tools require third-party tools like Matlab or drone simulators such as Gazebo, Papparazzi or Px4. Some of the tools are focused on a specific problem like a precision landing simulator, a drone-port, or an indoor positioning system. The third group is composed of those tools addressing system verification and validation. Modeling languages used are UML, Signal Temporal Logic, and a proprietary HW and fault model.

Each tool in the framework facilitates different design and verification steps in the design process of drone-based services. Most tools are general-purpose and can be applied to different steps in different projects. In some cases, the tools address a specific design problem and cannot be used out of it.

7.2 Tools Supporting the C4D methodology

The main contribution of tools supporting the **C4D** methodology can be divided into three categories: tools aimed at improving system modeling and code generation, tools whose goal is to support verification and validation processes, and tools for systems analysis and optimization. The contribution of these groups of tools to the **C4D** methodology is described in the next sections (more details are in deliverables D6.1/2).

7.2.1 Drone System Modelling and Code Generation

The use of system modeling languages and tools for designing CPSs has experienced significant growth in recent years. This trend has led to several studies using model-based design for CPSs, especially in **C4D** scenarios for UAVs (see Table 17). For example, S3D provides a model-based approach and tool for designing drones that are explicitly efficient to perform a variety of tasks (e.g., autonomous control, surveillance, navigation, image processing, etc.).

HEPSYCODE, on the other hand, works at electronic system-level and considers only the behavioural model view of the system, while it has been extended for UAV/UAS systems. Other works on drone system design start from an abstract representation of the system, which is progressively refined in subsequent steps until final code generation. This is the case of eSW Design Environment (ESDE), which supports the design and validation of drone systems. It produces reusable software code blocks that efficiently target safe implementations on the navigation platform. By "safe" implementation it means implementations that preserve the executing semantics and for which some level of performance guarantees can also be provided.

In addition, the continuous demand for high performance drone systems has led system designers to use heterogeneous components, often based on FPGA technologies. Due to their complexity, the design methodology employed plays a crucial role in determining the product quality of such heterogeneous HW/SW multiprocessor architectures. However, the selection of a suitable implementation is problematic due to the large number of heterogeneous HW /SW components on the market. In this respect, HEPSYCODE allows to consider the impact that mapping to the HW platform would have on the system behaviour without having to develop a corresponding TLM structural model. This is achieved through an approach inspired by native simulation, but combined with offline timing estimates at the instruction level to avoid the need for binary code analysis.

Another tool is MCD, which derives the coarse-grained reconfigurable co-processing units used in defining and integrating application-specific HW PEs into an overlay compute cluster. A set of tools to facilitate application development for the FPGA-based heterogeneous acceleration platforms is also provided. Furthermore, functionality for drone system modeling is provided by tools supporting mission design and optimization.

Table 17: Summary of tools for drone system modelling and code generation

Tool Name	Tool Description	Contribution to Methodology
Single-Source System Modeling & Design Framework (S3D)	Model-driven tool allows to capture all the relevant information about the system in order to support the different design steps. It starts from initial functional system architecture until the SW stack to be compiled to each computing resource in the decided HW implementation.	S3D has been extended in order to support modeling of drone-based services. It provides the system engineer with a modeling and design framework for services making use of robots in general and drones in particular.

eSW Design Environment (ESDE)	Electronic system-level (high-level) design and validation tool for algorithms to be executed on a microcontroller-based platform.	ESDE supports faster and more efficient design of the electronic components of the drone system, and specifically of the navigation software. The target is to produce reusable software blocks, captured on a standard language (SystemC), which enable high-level functional and basic time analysis. It is capable to integrate everyday more building blocks of the **C4D** general architecture.
HEPSYCODE	Prototypal toolchain for improving the design time of embedded systems. It is based on a system-level methodology for HW /SW co-design of heterogeneous parallel dedicated systems.	HEPSYCODE offers a design space exploration for mixed-criticality systems, while the tool will provide fine-grained result analysis on partitions, time slots and scheduling plans, connectivity issues, and mixed-criticality interconnections on multicore UAVs.
Papyrus for Robotics (P4R)	P4R is an open-source model-based engineering platform that features a set of domains specific modeling languages and tools for robotic applications design.	P4R enables the drone system modelling and code generation.
MultiDataflow Composer (MDC)	MDC has been developed to create coarse-grained reconfigurable hardware specifications. It has been then extended with additional features such as the automatic co-processor generation, compatible with the XILINX design environment.	MCD is adopted in **C4D** to derive the coarse-grained reconfigurable co-processing units to be used when defining and integrating application specific HW PEs into the overlay compute clusters.
FPGA-based heterogeneous acceleration platforms	High-level programming model and runtime system for onboard programmable and reconfigurable compute platform design.	The tool enables interaction with the onboard programmable and reconfigurable compute platform design methodology to accelerate the highly computationally intensive tasks of **C4D** drone reference architecture.

7.2.2 Drone Systems Validation and Verification

The verification and validation of Guidance, Navigation, and Control (GNC) algorithms is fundamental for the development of autonomous drones. Modeling and simulation tools are used to help reduce verification and validation costs by testing these algorithms on virtual test benches before deploying them on physical prototypes (see Table 18). This is the case for the open-source project Paparazzi UAV that address both ground and air software for different drones' platforms. Simcenter Amesim contributes to this effort providing drones high fidelity models simulating the drones flight/ground dynamics and its systems' performance. SimCloud, which leverages web-technology, cloud hosting options for UAS developers to test missions, control systems, and even hardware components, in a virtual environment.

Path planning, a subpart of GNC, is critical in the development of drones and it aims at using algorithms to determine optimal trajectories to guide a drone on its mission. In case high costs and/or strict safety requirements are at stake, the path should be validated against the missions' requirements. This is the purpose of AirMPL. DronePort and Battery Management, aim at minimizing verification effort of path planning approaches for drone fleets for battery recharging or replacement operations. Additional verification tasks supported by tools concern communication security (MoMuT), and specification consistency checking, automatic test pattern generation (Sage suite).

Table 18: Summary of tools for systems validation and verification

Tool Name	Tool Description	Contribution to Methodology
Paparazzi UAV	Open-source project for drones that address both ground and air software, for fixed wing, rotorcrafts, hybrid, and rovers.	Multiple options are available for each part of the autopilot stack by implementing multiple state of the art control algorithms. A flexible design allows to change any part of it. It is then suitable as test platform for a great variety of development and testing. Developments are devoted to make the system more reliable.
Simcenter Amesim	A tool dedicated to modelling and simulation of dynamic and multi-physics systems.	It allows the creation of plant models that can be used as a virtual test bench to support continuous development and verification of GNC algorithms.
SimCloud	Cloud-based platform for full system hardware-in-the-loop simulations.	UAS developers and mission planners can test missions, control systems, and even hardware components, in a virtual environment.
Sherpa Engineering	Drone simulator based on an existing open-source platform.	Support testing and verification of algorithms in simulation, before testing them on the real drone. This will help reducing the verification and testing cost. Developments aim at implementing a precision landing maneuver.
AirMPL	Tool for the testing and validation of drone's missions for precision farming applications.	It enables the validation to ensure that the missions' functional requirements are satisfied. This is achieved by means of formal approaches that can describe the mission in an unambiguous way.
DronePort	Set of tools to improve design of drone components and accessories in environment of CAD and CAE.	The DronePort design tool is focused to design more platform independent design of the DronePort system and battery modules to enable flexible usage of DronePort ground station and battery manipulator. The ongoing development is focused to maximize the flexibility of the DronePort system design and to minimize the integration effort by user.
Battery Management System	Drone battery management simulator is a simulation tool based on Gazebo environment simulator interfaced via ROS. This simulator implements Droneport to guide drone for landing, change/charge its battery and return it back to action.	The simulation of DronePort device is being developed primarily to minimize verification effort by using general DronePort model inside Gazebo simulator. The ongoing development aims at improving of Droneport model within the meaning of its fidelity and suitability for mission fulfilling drones
MoMuT	MoMuT is a model-based testing tool addressing functional and non-functional test-case generation from behaviour models.	MoMuT can be used to test application specific communication. It is applied in the project to a prototypic implementation of a cryptographic key exchange with a small footprint (e.g., low power-consumption). It provides a way to ensure that communication is implemented correctly.
Sage suite	The suite is composed of several tools devoted to different (formal) verification tasks like specification consistency checking, automatic test pattern generation, and neural network verification.	In the **C4D** design framework, the SAGE suite is adopted for model verification. Developments currently address scalability issues.

7.2.3 Drone System Analysis and Optimization

In this section, we discuss how tools dealing with drones' systems analysis and optimization support the **C4D** methodology (see Table 19). Starting from the early stages of a drone systems' design cycle, analysis and optimization are performed at high level to effectively explore the design space and provide relevant information to support design decision. To help drone manufacturers iterate effectively during the initial design phase, a multi-level simulation framework for drone-based services is developed which called SoSim. As the design becomes more refined and the drone analysis and optimization activities involves physical systems and their performance, high fidelity models are needed to enable trade-offs between different drones' architectures based on performance metrics. On top of that, high fidelity models are used as a virtual test bench to support the continuous development and verification of GNC algorithms. For these purposes, software tools dedicated to modelling and simulation of dynamic and multi-physics systems as Siemens' Simcenter Amesim are utilized.

Through the SimCloud platform, simulation tooling for hardware-in-the-loop testing of drone subsystems and components is provided. This allows faster development iterations, and the application of agile methodologies on drone hardware development, especially towards the development of firmware and embedded software.

Drones' reliability depends on the ability to design secure systems. Modelling of drones' systems or components architectures can help identify potential security gaps and provide means to resolve them, as proposed by Security Analysis Tool. For what concerns autonomous drones that operates in closed environments, their reliability depends on the capacity to determine their location without relying of GPS. To develop effective and reliable indoor positioning systems, modelling and simulation tools such as IPS-MAF enables to assess their performance.

Table 19: Summary of tools for analysis and optimization

Tool Name	Tool Description	Contribution to Methodology
SoSim	Multi-level simulation framework for drone-based services able to simulate the system at a very high and pure functional level at the earliest stages of the design cycle.	It enables system simulation at different abstraction levels.
Simcenter Amesim	Software tool dedicated to modelling and simulation of dynamic and multi-physics systems.	It enables trade-offs between different drones' architectures from a performance standpoint. Ongoing developments aim at improving the level of fidelity (propellers, aerodynamics, etc.)
Security Analysis Tool	Threat modeling tool for continuous model-based engineering.	Identification though modeling of potential security gaps which can be resolved by fine tuning the security properties.
IPS-MAF	Indoor Positioning System Model and Analysis Framework	It enables the modelling of an IPS and the high-level analysis of main aspects impacting on the performance of the positioning solution. Developments are devoted to the early assessment of positioning algorithms on anchor and tag firmware and of cost-performance optimal deployments
SimCloud	Cloud-based platform for full system hardware-in-the-loop simulations	It allows quick development iteration, shorter test cycles, and the application of agile methodologies on drone hardware development, especially towards firmware and embedded software.
HEPSIM2	HEPSYCODE SystemC Timing Simulator for HW/SW Co-Design of Heterogeneous Multi-Processor Embedded Systems	It enables the modeling of complex testbenches by considering independent stimulus generators to represent the environment and provides the ability to model complex UAV scenarios. HEPSIM2 Hierarchical Scheduler has been enhanced for multicore systems to provide better usability and scalability.

8 Conclusion

In this deliverable, first, we have described the generic procedure to develop a drone system. This procedure starts with the specification of the system's concept of operations which is then used for identifying a number of technologies/components. These technologies with a set of guidelines are then used for developing the drone system.

Second, we have presented the different drone categories (i.e., open, specific, and certified), the existing regulation requirements that affect the drone system development, and the specific operational risk assessment methodology (SORA). Third, key enabling technologies for drones are described. These technologies are categorized in four groups: drone capabilities for supporting U-space services, system functions, payload technologies, and tools that support the system development. We also present the technologies that are being developed in the project such as generic components to support the reference architecture, components to enable safe and autonomous flight, and technologies that enable the trusted communication.

Fourth, to ease the development of drone systems, a number of guidelines/recommendations are provided. The guidelines are for the development process in general, enabling the development of safe drone by taking into account different failures, re-use of existing platform technologies, considering the mixed-critically aspect during the system development, architecture evaluation and performance optimization, hardware-based security, and development of specific system features.

Finally, the system engineering approach for drone system development is introduced. This approach is composition-based approach, where a composition structure is the main driver of the development (i.e., the reference architecture proposed in the project). The different phases of the engineering approach include concept of operations specification, selection of the reusable technologies, system architecture design, system implementation and technologies integration, and the system's validation and verification. The tools that support the different phases of engineering approach are also presented. The tools targets system modelling and code generation, system validation and verification, and system analysis and optimization.

In the next deliverable (D2.4), a final version of the project framework and methodology will be provided. A special attention will be given to the safety, security, redundancy, and mixed-criticality aspects of the drone systems.