



STO TECHNICAL REPORT

TR-IST-149

Capability Concept Demonstrator for Interoperability within Unmanned Ground Systems and C2

(Modèle de démonstration d'un concept de capacité
en vue de l'interopérabilité des systèmes
terrestres inhabités et du C2)

Final report of NATO RTG IST-149.



Published June 2021





STO TECHNICAL REPORT

TR-IST-149

**Capability Concept Demonstrator for
Interoperability within Unmanned
Ground Systems and C2**
(Modèle de démonstration d'un concept de capacité
en vue de l'interopérabilité des systèmes
terrestres inhabités et du C2)

Final report of NATO RTG IST-149.

Authors: F. Schneider, K. Mathiassen, P. Bounker, J. Glówka, A. Tiderko,
M. Baksaas, R. Kozik, J.J. Röning and G. De Cubber.

The NATO Science and Technology Organization

Science & Technology (S&T) in the NATO context is defined as the selective and rigorous generation and application of state-of-the-art, validated knowledge for defence and security purposes. S&T activities embrace scientific research, technology development, transition, application and field-testing, experimentation and a range of related scientific activities that include systems engineering, operational research and analysis, synthesis, integration and validation of knowledge derived through the scientific method.

In NATO, S&T is addressed using different business models, namely a collaborative business model where NATO provides a forum where NATO Nations and partner Nations elect to use their national resources to define, conduct and promote cooperative research and information exchange, and secondly an in-house delivery business model where S&T activities are conducted in a NATO dedicated executive body, having its own personnel, capabilities and infrastructure.

The mission of the NATO Science & Technology Organization (STO) is to help position the Nations' and NATO's S&T investments as a strategic enabler of the knowledge and technology advantage for the defence and security posture of NATO Nations and partner Nations, by conducting and promoting S&T activities that augment and leverage the capabilities and programmes of the Alliance, of the NATO Nations and the partner Nations, in support of NATO's objectives, and contributing to NATO's ability to enable and influence security and defence related capability development and threat mitigation in NATO Nations and partner Nations, in accordance with NATO policies.

The total spectrum of this collaborative effort is addressed by six Technical Panels who manage a wide range of scientific research activities, a Group specialising in modelling and simulation, plus a Committee dedicated to supporting the information management needs of the organization.

- AVT Applied Vehicle Technology Panel
- HFM Human Factors and Medicine Panel
- IST Information Systems Technology Panel
- NMSG NATO Modelling and Simulation Group
- SAS System Analysis and Studies Panel
- SCI Systems Concepts and Integration Panel
- SET Sensors and Electronics Technology Panel

These Panels and Group are the power-house of the collaborative model and are made up of national representatives as well as recognised world-class scientists, engineers and information specialists. In addition to providing critical technical oversight, they also provide a communication link to military users and other NATO bodies.

The scientific and technological work is carried out by Technical Teams, created under one or more of these eight bodies, for specific research activities which have a defined duration. These research activities can take a variety of forms, including Task Groups, Workshops, Symposia, Specialists' Meetings, Lecture Series and Technical Courses.

The content of this publication has been reproduced directly from material supplied by STO or the authors.

Published June 2021

Copyright © STO/NATO 2020
All Rights Reserved

ISBN 978-92-837-2278-2

Single copies of this publication or of a part of it may be made for individual use only by those organisations or individuals in NATO Nations defined by the limitation notice printed on the front cover. The approval of the STO Information Management Systems Branch is required for more than one copy to be made or an extract included in another publication. Requests to do so should be sent to the address on the back cover.

Table of Contents

	Page
List of Figures	vii
List of Tables	ix
IST-149 Membership List	x
 Executive Summary and Synthèse	 ES-1
 Chapter 1 – Contributions of Belgium to NATO RTG IST 149: Interoperability in Heterogeneous Teams	 1-1
1.1 Introduction	1-1
1.2 Approach to Interoperability	1-2
1.2.1 Ontology Definition	1-2
1.2.2 Interoperability Levels	1-3
1.2.3 Adjustable Automation	1-4
1.2.4 Multi-Robot Cooperation	1-6
1.3 Analysis of Existing Standardization Initiatives	1-8
1.4 Interoperability Standard	1-9
1.4.1 Services Sets	1-9
1.5 An Interoperable Layer: The Library	1-12
1.5.1 JAUSRobot	1-13
1.5.2 JAUS C2I	1-13
1.5.2.1 JAUS Fleet Handler	1-14
1.5.2.2 JAUS Robot Handler	1-14
1.5.2.3 Management of the Communications	1-14
1.6 Individual Platforms Adaptations	1-15
1.6.1 ROS to Robot Node	1-15
1.6.2 Other Robot Adapter	1-16
1.7 Field Validations: Examples of Multi-Robot Cooperation	1-18
1.7.1 Cooperative Multi-Stage Aerial Surveillance	1-19
1.7.2 Aerial Scouting for Traversability Analysis	1-19
1.7.3 Victim Search	1-21
1.7.4 Carrier	1-23
1.7.5 Communications Relay	1-24
1.7.5.1 Air, Sea, Ground Cooperation During euRathlon 2015	1-25
1.8 Conclusions and Acknowledgements	1-26
 Chapter 2 – Interoperability Standards of UGVs and Tests of Interoperability of UGVs at Robot Competition M-ELROB	 2-1
<i>Abstract</i>	2-1
2.1 Proposed UGV Standards	2-1

2.2	ROS	2-2
2.3	BML	2-2
2.4	Platform / Payload	2-3
2.5	Test Tracks	2-4
2.6	M-ELROB	2-4

Chapter 3 – UGV Interoperability Profile (IOP) and Tests of OCU Interoperability at Training Ground in Rena, NOR **3-1**

<i>Abstract</i>	3-1
3.1 IOP OCU Standards	3-1
3.2 ROS-JAUS Bridge	3-1
3.3 UGV Handoff Between OCUs	3-1
3.4 CHE-DEU Common Platform	3-2
3.4.1 Interoperability Tests in Rena, Nor	3-2
3.5 Outlook	3-3

Chapter 4 – Polish Conceptual OCU **4-1**

<i>Abstract</i>	4-1
4.1 Scope of Cooperation	4-1
4.2 Polish Conceptual OCU	4-1
4.2.1 Description	4-1
4.2.2 Hardware Setup	4-1
4.2.3 Software Setup	4-2
4.3 Conducted Tests	4-3
4.3.1 Tests with Robots	4-3
4.4 General Findings	4-3
4.5 Video Interoperability	4-6
4.6 Supported IOP Functionalities	4-7
4.6.1 Messages	4-7
4.6.2 Commands	4-7
4.7 Conclusions and Experience Gained	4-7
4.7.1 Technical Findings	4-7
4.7.2 Interoperability Findings	4-8

Chapter 5 – Contribution of the University of Oulu **5-1**

<i>Abstract</i>	5-1
5.1 Description of the System	5-1
5.2 Custom Copter's System Overview	5-3
5.3 "Plugfest" Outcome	5-3
5.4 UAS Control Software Architecture	5-3
5.5 Video Streaming from UAS Over Wi-Fi to GCS	5-5
5.6 Offline UAS Testing with SITL	5-6
5.7 UAS Crash Before the IOP Experiment in Rena Norway	5-7

Chapter 6 – US Overall Support 6-1

<i>Abstract</i>	6-1
6.1 IOP	6-1
6.2 Interoperability Attributes	6-2
6.3 Interoperability Attribute Reference Groups	6-2
6.4 Implementation of Standards	6-3
6.5 Control and Status Messages and JAUS Profiling	6-3
6.6 Custom Services, Messages, and Transports	6-3
6.7 System Context	6-3
6.8 Reference Architecture	6-4
6.9 Operator Control Unit (OCU)	6-5
6.10 Common Communications Link (CCL)	6-5
6.11 Unmanned Ground Vehicle (UGV)	6-6
6.12 External Command and Control (C2)	6-6
6.13 Other Unmanned Vehicles (UxVs)	6-6
6.14 UGV Logical Internal Architecture	6-6
6.15 Overview of IOP Usage Process	6-8
6.16 IOP Usage	6-9
6.17 Usage of Communications IOP	6-9
6.18 Usage of Payloads IOP	6-9
6.19 Usage of Control IOP	6-9
6.20 ROS-M	6-9
6.21 Logistics and SA	6-10
6.22 SAE	6-10
6.23 GVR-BOT	6-11
6.23.1 Legacy PackBot	6-11
6.23.2 VR-BOT	6-11
6.24 Mapping Sensor	6-12
6.25 Controller	6-13

Chapter 7 – Norwegian Defence Research Establishment 7-1

<i>Abstract</i>	7-1
7.1 Trial Organization	7-1
7.1.1 Trial Location – Close Quarters Combat Training Ground	7-2
7.2 IOP Implementation on Robot	7-2
7.2.1 Olav Description	7-2
7.2.2 IOP Implementation on Olav	7-4
7.2.2.1 Using Fraunhofer IOP Bridge	7-5
7.2.2.2 Using Gstreamer to Provide RTSP Stream	7-6
7.3 Experiment Setup	7-8
7.3.1 Network Setup	7-8
7.3.2 Operation Center Setup	7-9
7.4 Experiences from Trials	7-10
7.5 Media Coverage and Dissemination	7-11

Chapter 8 – Fraunhofer Institute for Communication, Information Processing and Ergonomics (FKIE)

8-1

8.1	Interoperability Trial Participation	8-1
8.1.1	Robot System “Telexmax”	8-2
8.1.2	Payload for Robot System “GARM”	8-4
8.1.3	Control Station “Fernmeldekabine 2B”	8-6
8.2	Communication Infrastructure	8-9
8.3	Conclusion	8-11

Chapter 9 – References

9-1

List of Figures

Figure		Page
Figure 1-1	Relevant JAUS Services	1-10
Figure 1-2	Additional Non-Standard JAUS Services	1-11
Figure 1-3	JAUS Topology	1-12
Figure 1-4	Robot Adaptation Strategy	1-12
Figure 1-5	Multiple Aircraft in the Air at the Same Time	1-20
Figure 1-6	Low-Resolution Fixed-Wing Assessment and High-Resolution Multi-Rotor Assessment	1-20
Figure 1-7	Outdoor Rotor-Craft Ensuring Optimal Routing of Ground Team	1-21
Figure 1-8	Automated Victim Detection on a Rubble Field in an Urban Search and Rescue Operation	1-21
Figure 1-9	Victim in the Water Being Localized by Outdoor Multi-Rotor Aircraft	1-22
Figure 1-10	Collaborative Indoor Victim Search	1-22
Figure 1-11	Collaborative Maritime Victim Search and Rescue Operation Involving Aerial and Maritime Platforms	1-23
Figure 1-12	Large Unmanned Ground Vehicle Deploying the Small Unmanned Ground Vehicle on the Top of a Building	1-24
Figure 1-13	Unmanned Surface Vehicle Deploying the Unmanned Rescue Capsule	1-24
Figure 1-14	Rotor-Craft During the euRathlon Challenge	1-25
Figure 1-15	Teodor and Small UGV During the euRathlon Challenge	1-26
Figure 1-16	3D Map of the Crisis Area, Obtained by Combining 3D Maps from the Aerial and Ground Platforms	1-26
Figure 2-1	The BML-GUI Which is Developed by the Fraunhofer Institute FKIE Allows the Commanding of a Multi-Robot System by a Single Operator, and the Display of the Data Collected by the (Sensors of) the Robots	2-3
Figure 2-2	Robot Platform, ‘Garm’ with a Standardized Payload	2-3
Figure 3-1	Robot Platform, ‘Garm’, with a Reconnaissance Payload	3-2
Figure 3-2	Parade of Robotic Platforms Participating at Interoperability Test in Rena	3-3
Figure 4-1	The Polish Conceptual OCU Consists of a Docking Station with Joysticks and Buttons	4-2
Figure 4-2	A Tablet Computer is Inserted into the Docking Station	4-2
Figure 4-3	GARM Robot, Teleoperation	4-3

Figure 4-4	Garm Robot, Teleoperation with OCU	4-4
Figure 4-5	GVR-Bot (TARDEC), Teleoperation	4-4
Figure 4-6	Telerob (WTD 41), Teleoperation	4-5
Figure 4-7	Olav (FFI), Teleoperation	4-5
Figure 4-8	Chromium Uses a Combination of Range Requests and an In-Memory Sliding Window to Buffer Media	4-6
Figure 5-1	Custom S500 Quadcopter	5-2
Figure 5-2	Custom Copter System Overview	5-3
Figure 5-3	List of IOP Components and Services Utilized	5-4
Figure 5-4	UAS Control Architecture	5-5
Figure 5-5	ROS Node Graph of the Remote GCS Laptop	5-6
Figure 5-6	Mission Planning / Simulation with SITL	5-6
Figure 5-7	Architecture of SITL	5-7
Figure 5-8	Quadcopter After Crashing	5-7
Figure 5-9	Hexa- and Octocopter	5-8
Figure 6-1	RAS-G IOP Context	6-4
Figure 6-2	RAS-G IOP Reference Architecture	6-5
Figure 6-3	Example UGV Logical Internal Architecture	6-7
Figure 6-4	GVR-BOT	6-12
Figure 6-5	Mapping Sensor Mounted on GVR-BOT	6-12
Figure 6-6	Controller	6-13
Figure 7-1	Trial Location in Norway	7-1
Figure 7-2	Map Showing the Trial Area	7-2
Figure 7-3	The Robotic System Olav	7-3
Figure 7-4	Control Architecture of Olav	7-4
Figure 7-5	Network Setup from Olav's Perspective	7-9
Figure 7-6	The Operation Center	7-9
Figure 7-7	All Experiment Participants and Robots	7-11
Figure 8-1	Robot System Telemex	8-3
Figure 8-2	Autonomiebox for Telemex	8-3
Figure 8-3	Payload for Reconnaissance Scenario	8-4
Figure 8-4	Robot Chassis GARM from RUAG and Payload from FKIE	8-5
Figure 8-5	Part of Control GUI While Control GARM	8-7
Figure 8-6	Visualization of Robot Position in Rviz	8-7
Figure 8-7	Multi-Robot View	8-8
Figure 8-8	Exterior View of FKIE Control Station	8-8
Figure 8-9	Inside View of FKIE Control Station	8-9
Figure 8-10	Network Topology	8-10

List of Tables

Table		Page
Table 1-1	Examples of the Ontology Concepts Organized by Categories	1-3
Table 1-2	Definition of the Levels of Interoperability	1-4
Table 1-3	LoI for Each of the Robots	1-4
Table 1-4	Definition of the Level of Automation	1-5
Table 1-5	Level of Automation for Each of the Robots	1-6
Table 1-6	Goals to Roles and Tasks Decomposition	1-7
Table 1-7	Examples of Robot Roles	1-7
Table 1-8	Interface Gaps	1-10
Table 1-9	Services Provided by Each Vehicle	1-16
Table 1-10	Custom Services Provided by Each Vehicle	1-17
Table 2-1	Levels of Interoperability	2-4
Table 5-1	UAS Specifications	5-1
Table 6-1	Attribute Reference Group Selection Types	6-2
Table 7-1	Detailed Explanation of the Streaming Pipeline in the Perception Module	7-6
Table 7-2	Detailed Explanation of the Streaming Pipeline for the RTSP Server Publishing the Perception Module Video Feed	7-7
Table 7-3	Detailed Explanation of the Streaming Pipeline for the RTSP Server Publishing the RWS Video Feed	7-8
Table 8-1	Participants' Hardware for the Trial	8-1

IST-149 Membership List

CO-CHAIRS

Dr. Kim MATHIASSEN
Norwegian Defence Research Establishment
NORWAY
Email: kim.mathiasen@ffi.no

Dr. Frank SCHNEIDER
Fraunhofer Institute for Communication,
Information Processing and Ergonomics (FKIE)
GERMANY
Email: frank.schneider@fkie.fraunhofer.de

MEMBERS

Mr. Paul BOUNKER
US Army RDECOM TARDEC
UNITED STATES
Email: paul.j.bunker.civ@mail.mil

Dr. Geert DE CUBBER
Royal Military Academy
BELGIUM
Email: Geert.Decubber@mil.be

Mr. Jakub GLÓWKA
Industrial Research Institute for Automation &
Measurements PIAP
POLAND
Email: jglowka@piap.pl

Dr. Thomas NUSSBAUMER
RUAG Schweiz AG
SWITZERLAND
Email: Thomas.Nussbaumer@ruag.com

Prof. Juha Jaakko RÖNING
University of Oulu
FINLAND
Email: juha.roning@oulu.fi

PANEL/GROUP MENTOR

Dr.-Ing. Michael WUNDER
Fraunhofer Institute for Communication, Information Processing and Ergonomics (FKIE)
GERMANY
Email: michael.wunder@fkie.fraunhofer.de

Capability Concept Demonstrator for Interoperability within Unmanned Ground Systems and C2

(STO-TR-IST-149)

Executive Summary

This report describes the research and experiment efforts of the NATO STO RTG IST-149 Capability Concept Demonstrator for Interoperability within Unmanned Ground Systems and C2. Unmanned Ground Vehicles (UGVs) are becoming increasingly relevant in the modern battle-space. These systems can carry massive sensor suites, delivering unprecedented streams of data right from the front line. On the other hand, these systems still require tele-operation in most cases. It is very important to realize that the ISR data will remain largely useless if there is no proper way to exchange the information between the coalition partners and/or pipe it into C2 systems. The main purpose of the group was to find ways to improve this situation, and more specifically to investigate possible standards for controlling UGVs from Operator Control Units (OCUs) and receiving data back, and testing them in a real world scenario.

The efforts of the project have been two-fold. The Belgian contribution is work done in the EU project ICARUS. This project involved a team of assistive unmanned air, ground and sea vehicles for Search and Rescue. The interoperability was verified in several different experiments. The ICARUS consortium consisted of several international partners, where Belgium was the link to this group. The second effort was a combined effort in the group to have an experiment demonstrating interoperability between the UGVs and OCUs available within the group. The group had a final demonstration in Rena, Norway, in 2018.

Both efforts used the Joint Architecture for Unmanned Systems (JAUS) with the Interoperability Profile (IOP) to successfully enable interoperability between the systems. The trials showed that it is possible to extend the systems quite easily and achieve compliance with parts of the standard in a relatively short time. Both Fraunhofer FKIE and TARDEC had developed software to pass information from the IOP domain to and from the Robotic Operating System (ROS). ROS is a widely used software for developing autonomy for UGVs and other types of robots and used by many of the partners in the group. The software provided by Fraunhofer FKIE and TARDEC was vital to the success of the trials.

The report also discusses how the IOP standard can be used to define requirements for a system before procurement. The standard itself defines a set of attributes that can be specified as requirements when procuring a new system, either as mandatory or optional requirements. This makes it easier for procurement offices to define the requirements and for vendors to conform to the requirements, and it also becomes clear which capabilities an OCU needs to have when connecting to a system with regards to controlling it and visualizing data available from the system.

The group's trials in Rena, Norway, in 2018 focused on tele-operations of the UGVs and receiving position and video feedback from the UGVs. As this was a successful trial, the next step would be to test interoperability using higher level control input and feedback, as for instance, sending waypoints to the UGVs and receiving maps of the environment around the system based on the system's perception.

Modèle de démonstration d'un concept de capacité en vue de l'interopérabilité des systèmes terrestres inhabités et du C2 (STO-TR-IST-149)

Synthèse

Le présent rapport décrit les travaux de recherche et d'expérimentation du RTG de la STO de l'OTAN IST-149 intitulé « Modèle de démonstration d'un concept de capacité en vue de l'interopérabilité des systèmes terrestres inhabités et du C2 ». Les véhicules terrestres sans pilote (UGV) prennent de plus en plus d'importance sur le champ de bataille. Ils peuvent transporter des ensembles massifs de capteurs et fournissent des flux de données sans précédent, directement depuis la ligne de front. D'autre part, ces systèmes ont tout de même besoin d'un téléopérateur dans la plupart des cas. Il est très important de comprendre que les données ISR resteront largement inutiles s'il n'existe pas de moyen correct d'échanger les informations entre les partenaires de coalition et/ou de les injecter dans des systèmes de C2. Le principal objet du groupe était de trouver comment améliorer cette situation et, plus précisément, d'étudier des normes possibles pour commander les UGV depuis des unités de télécommande (OCU) et recevoir des données en retour, puis de les tester dans un scénario du monde réel.

Les travaux de ce projet avaient deux volets. La contribution belge est le travail qui a été effectué dans le projet ICARUS de l'UE. Ce projet impliquait une équipe de véhicules aériens, terrestres et maritimes sans pilote assistant les opérations de sauvetage. L'interopérabilité a été vérifiée à l'occasion de plusieurs expériences différentes. Le consortium ICARUS se composait de plusieurs partenaires internationaux, dont la Belgique, qui assurait la liaison entre les membres. Le second volet était une expérience combinée du groupe, visant à démontrer l'interopérabilité entre les UGV et les OCU disponibles au sein du groupe. Le groupe a fait une démonstration finale à Rena, en Norvège, en 2018.

Les deux volets de ce projet ont utilisé l'architecture commune pour les systèmes sans pilote (JAUS) et le profil d'interopérabilité (IOP) afin de permettre l'interopérabilité entre les systèmes. Les essais ont démontré qu'il était possible d'élargir assez facilement les systèmes et de se conformer à certaines parties de la norme dans un délai relativement court. L'institut Fraunhofer FKIE et le TARDEC avaient tous deux mis au point un logiciel pour transférer des informations du domaine IOP de et vers Robot Operating System (ROS). ROS est un logiciel couramment utilisé pour développer l'autonomie des UGV et d'autres types de robots et utilisé par beaucoup de partenaires au sein du groupe. Le logiciel fourni par Fraunhofer FKIE et le TARDEC a été essentiel à la réussite des essais.

Le rapport discute également de l'utilisation de la norme IOP pour définir les besoins d'un système avant l'acquisition. La norme elle-même définit des ensembles d'attributs qui peuvent faire l'objet de spécifications de besoin (obligatoires ou optionnelles) lors de l'achat d'un nouveau système. De la sorte, les bureaux d'acquisition définissent plus facilement les exigences et les fournisseurs répondent plus facilement aux besoins. Les capacités nécessaires à la commande d'une OCU et à la visualisation des données disponibles dans un système lors de la connexion avec celui-ci apparaissent clairement.

Les essais du groupe à Rena, en Norvège, en 2018, se sont concentrés sur les téléopérations des UGV et sur la réception de retours vidéo et d'informations de position en provenance des UGV. Puisque l'essai a été

une réussite, l'étape suivante serait de tester l'interopérabilité avec des données de commande entrantes et sortantes de niveau plus élevé, par exemple, en envoyant des points de cheminement aux UGV et en recevant des cartes de l'environnement autour du système, basées sur la perception du système.



Chapter 1 – CONTRIBUTIONS OF BELGIUM TO NATO RTG IST-149: INTEROPERABILITY IN HETEROGENEOUS TEAMS

Geert De Cubber

Royal Military Academy of Belgium
BELGIUM

1.1 INTRODUCTION

There are nowadays many different types of unmanned systems being used in different domains and, certainly, this number will increase significantly in the upcoming years. In general, large scale systems aiming at solving all problems with one single type of platform have proven to be expensive and not flexible enough. Heterogeneous teams, composed by Unmanned Air, Ground, Surface, and Underwater Systems (UxS), of different types and sizes, offer the possibility to exploit the best features of each kind and combine them to obtain compound capabilities, which have demonstrated to be more cost-efficient and adaptable to new scenarios. Recent research efforts have focused on developing the autonomy of the team by increasing the interactions between these systems, making them aware of each other, executing tasks that require cooperation, and finally implementing flock or swarm coordinated behaviors.

In order to optimize the use of resources, the activities of Belgium inside RTG-149 were co-framed inside the EU-H2020 ICARUS project, which has a similar focus on robot interoperability. The ICARUS project involves a team of assistive unmanned air, ground and sea vehicles for Search and Rescue operations. In order to effectively support the on-site person responsible for the operations, these systems must be able to collaborate as a seamlessly integrated team, coordinated from the Robot Command and Control station (RC2) in the field. It must therefore be stressed that the contributions shown in this chapter are not only a result of the Belgian delegates, but are the result of a multi-national collaboration with Spain (EURECAT, INTEGRASYS), Belgium (RMA, Space Application Services), Portugal (Escola Naval, INESC Porto), Switzerland (ETH Zürich), Germany (University of Kaiserslautern), Poland (Institute of Mathematical Machines) and NATO-CMRE.

A heterogeneous fleet is the one composed by elements of different kinds such as the ICARUS team, including up to ten different vehicles (Long-Endurance Fixed-Wing, Outdoors Multi-rotor, Indoors Multi-rotor, Large UGV, Small UGV, Teodor UGV, U-Ranger USV, ROAZ USV, MAREAS AUV and several Rescue Capsules). Each robot has been developed by a different provider or partner, using their own designs, frameworks and middleware. Thus, a strong effort had to be devoted to their integration as a team and this is the work described in this chapter. Although many standards have been proposed by the community, most of the field robotic systems have their own command and reporting protocols, and consequently require their own ground control stations. This profusion of protocols makes the cooperation between systems difficult. The lack of unified standards poses an unnecessary burden on the operation and maintenance of multi-vehicle systems. The work described in this chapter aims at contributing to the harmonization of the multiple standardization initiatives for the coordination of heterogeneous teams.

The ultimate objective of this RTG is to achieve robot interoperability, which can be understood as the ability of robots to operate in synergy to the execution of assigned missions. Interoperability enables diverse teams to work together, sharing data, intelligence and resources.

1.2 APPROACH TO INTEROPERABILITY

Interoperability is the key that acts as the glue among the different units within the team, enabling efficient multi-robot cooperation. Seamless and non-ambiguous interaction between different robots of any provider and domain demands a common, well-defined interface.

We propose the adaptation of all the vehicles to a single standard external interface as a method to ensure interoperability. Each robot development team is free to use their own tools inside their systems as long as the interaction with the rest of the team follows a set of definitions and rules referred to as the interoperability standard. This follows the Façade pattern [1] very frequently used in Software Engineering. It essentially hides the complexities of the implementation and provides the outer components with a simpler interface. It is typically deployed as software library implementing a wrapper or adapter template. On one side, this library implements the interoperability standard interface and, on the other side, it provides a set of classes and functions (an API) for its integration with the specific middleware or software provided by each platform.

This approach may initially seem to reduce the level of integration among the agents if we compare it against natively sharing an internal protocol in all systems, but it promotes the maximum decoupling between the custom implementations, with its particularities, and the definition of the common interface. In the long term, this has shown to improve the seamless integration of the maximum number of systems and domains at a lower cost. The integration of new platforms into the team have literally been done in a matter of few hours during the project, provided that on-board hardware resources and communications are made available by the robot provider.

Therefore, the ultimate goal of the work on the heterogeneous team is to consolidate a common Command, Control and Payload interface to be agreed and adopted by all robotics platforms and Control Ground Stations (CGS) involved in an operation. This approach provides a common framework for the development of collaborative unmanned assets, minimizing the integration time and costs by avoiding ad hoc implementations.

There are other advantages in using interoperability standards. The use of a widely accepted interface helps to easily integrate new technologies with minor modifications to the existing systems. This facilitates the insertion of new technology for their operational use in the field, as end-users rely on proven technology and the preliminary validation will focus only on de-risking the new developments. Another advantage of the use of standards is that it will facilitate the backwards and forwards compatibility between existing and future vehicles and CGS provided by different providers. This can benefit companies to maximize the revenue from a specific product.

Our strategy in terms of interoperability is to build upon existing body of work in the field, avoiding duplicating and re-inventing proven technology. During the initial steps of the work, the most relevant multi-domain interoperability protocols for unmanned systems were identified and evaluated against the end-user requirements and foreseen scenarios. During this phase, several collaborations with other European [2] and NATO [3] initiatives, together with the organization of workshops involving end-users and stakeholders, were extremely relevant to gather good quality information on the state of the art in the field.

1.2.1 Ontology Definition

One of the challenges in multi-robot multi-domain interface standardization is to be able to embrace all type of systems, independently of their domain, particularities (i.e., size, operational modes, etc.) or constraints (i.e., computational resources, communication bandwidth, etc.). Therefore, in order to methodically evaluate the existing initiatives, an analysis of the robot-specific Interfaces Control (ICD) and Functional Specifications (FSD) was performed to generate what we referred to as the project interoperability needs. Any information that was domain- or platform- specific was removed from the analysis to ensure the level of abstraction required to ensure standardization. Likewise, the needs were further developed through an analysis of other potential vehicles that could be integrated into the system in the future.

This set of needs has been formalized as an ontology. An ontology is “an explicit, formal specification of a shared conceptualization” [4] We use it to describe the set of concepts required to coordinate a multi-robot Search and Rescue operation. This includes concepts, at different levels from robots to systems, capabilities and sensors, and their relationships and assumptions. There have been previous and parallel efforts in this field. Namely, the IEEE Robotics and Automation Society (IEEE-RAS) created a working group named Ontologies for Robotics and Automation that aims at the definition of a core ontology for robotics and automation [5]. The work performed has a strict focus on heterogeneous multi-robot operations in Search and Rescue, and as such, it proposes an application-specific ontology, addressing tasks and platforms involved in Search and Rescue missions.

This analysis resulted in a description of the set of multi-domain concepts and relationships or messages commonly found in unmanned systems. Table 1-1 summarizes the key categories and provides some example of interactions between systems.

Table 1-1: Examples of the Ontology Concepts Organized by Categories.

Category	Description and examples
Transport	Inter-process communication such as send, receive, broadcast, etc.
Commands	Generic accessors such as set, get, etc. for any standard concept
Management	Heartbeat, system status, clock synchronization, alarms, etc.
Telemetry	Pose and velocity reports in appropriate system coordinates, etc.
Telecontrol	Teleoperation, waypoint and mission management, etc.
Perception	Imagery, ranging, audio, etc.
Manipulation	Joint and end-effector control of robotics arms
Mapping	Maps, Digital Elevation Models, Point clouds
S&R Intelligence	Sectors, disaster alerts, humanitarian information

The complete ontology was used in a gap analysis for the evaluation of the existing standards as described in Section 1.3.

1.2.2 Interoperability Levels

A key concept that enables interoperability among the largest number possible of unmanned systems is the Levels of Interoperability (LoI). This concept is introduced by STANAG 4586 [6] and has been adopted and adapted for the purpose of our project. LoI defines the different degrees of compliance with the standard interface. It proposes a mechanism to account for a large variety of approaches and levels at which different systems can be integrated, accounting therefore for more integration strategies and combinations.

STANAG 4586 defines LoI as “the platform, subsystem or sensor ability to be interoperable for basic types of functions related to unmanned systems”. These levels show different degrees of control that a user has over the vehicle, payload or both. However, these definitions have been adapted to our project as follows.

Therefore, the Levels of Interoperability are defined in Table 1-2.

Table 1-2: Definition of the Levels of Interoperability.

Level of Interoperability	
LoI 1	Indirect receipt/transmission of telemetry, control and payload data: the UxV data is received from (or sent to) another source (another CGS, web-server, etc.).
LoI 2	Direct receipt/transmission of UxV telemetry and payload data, but without control authority over it.
LoI 3	Direct control and monitoring over the UxV without launch and recovery. A dedicated control station keeps control for the safety critical operations of the platform (i.e., take-off and landing, deployment and recovery, etc.) and hand it over to the CGS once ready for mission.
LoI 4	Highest level of interoperability. The CGS has full control of the UxV.

The levels of interoperability for each of the system are outlined in Table 1-3.

Table 1-3: LoI for Each of the Robots.

Level of Automation per Robot					
	LoI 1	LoI 2	LoI 3	LoI 4	Notes
Long-Endurance Fixed-Wing			X		Take-off and landing procedures for the UAVs are handled by the proprietary control stations. The system is handed over to the C2I once in air.
Outdoors Multi-rotor			X		
Indoors Multi-rotor			X		
Large UGV				X	
Small UGV				X	
U-Ranger USV	X				U-Ranger is a highly equipped and extremely fast USV. Integration is done through its proprietary CGS for safety purposes.
ROAZ USV			X		ROAZ USV is primarily operated from a proprietary CGS. When a mission starts, control is handed over to the C2I.
Rescue Capsule				X	

1.2.3 Adjustable Automation

One of the most critical end-user requirements is to ensure that a member of the Search and Rescue team in the field always supervises the robot operations to ensure safety and effectiveness. Robotics asset can generally be remotely controlled. Most of these systems also provide on-board autonomy modules that

allows the operator to plan a mission to be autonomously executed by the system. This should presumably help reducing the workload of the operator. However, in a realistic scenario, unexpected events are highly likely to occur and the intervention from the operator, such as manually overriding the mission execution, is often required. This increases the cognitive workload of the operator leading to stress and potential mistakes, which is even more critical in the context of multi-robot operations.

Adjustable Automation (AA) is the ability of a robot to behave autonomously and dynamically change its level of independence, intelligence and controllability to adapt to different tasks and scenarios [7]. AA present advantages when dealing with communication delays, human workload and safety [8]. Having systems that can dynamically reduce or increase the level of automation running on board provides a more flexible and reliable system.

AA is achieved by supporting multiple levels of automation in the robots, e.g., fully autonomous, guided by the operator, and fully controlled by the operator. The C2I also supports adjustable automation by automatically changing its display and control functions based on the relevance of the information, the current situation the robots encounter, and user preferences.

The level of automation of a robot is related to the degree of intervention of the human operator and other robots in the decision process. However, the fact that a robot is autonomous does not imply that it has to make all its decisions by itself. Different levels of automation and classifications have been described in the literature [9]. Specifically, Lacroix *et al.* [10] define five levels of automation according to the robot responsibilities towards a fleet of robots (tasks allocation, mission coordination, etc.), which is mostly relevant for tightly coupled coordination. The levels of automation are understood in terms of tasks execution and are reduced to essentially three modes, as shown in Table 1-4.

Table 1-4: Definition of the Level of Automation.

Level of Automation	
Level 1	Teleoperation. No automation on-board the robot. The robot is directly controlled by the operator.
Level 2	Semi-autonomous. Execution capabilities. The robot is able to manage partially ordered sequences of elementary tasks, and to return Execution status of the tasks. An operator is supervising the mission from the RC2.
Level 3	Fully autonomous. Deliberative capabilities. Complex tasks request are managed (tasks planning and scheduling).

A platform can seamlessly carry out a given task at different automation levels, depending on the robot operator choice, the mission plan priorities, workload and constraints of the mission and platform. As mentioned before, the concept of adjustable autonomy implies the ability to adapt and dynamically change between these levels of autonomy depending on situational changes. Some examples of adjustable autonomy are:

- A UAV may provide fully autonomous navigation in nominal conditions, but may fall back to semi-autonomous navigation in the presence of victims detected on the sensor stream.
- The RC2 operator may have initially designed the mission to manually operate the outdoor multi-rotor to inspect a building, but operation enters a highly complex area and he/she decides to enable semi-autonomous mode to ensure all corners are correctly surveyed.

Most platforms provide all three operation modes. However, there are specific constraints in some platforms due to their size or domain. Namely, the large UGV is usually remotely operated or waypoint guided. Such a

large system should not be tasked with predefined missions. On the other hand, the U-Ranger is such a fast maritime system that the operator should rely on the on-board autonomy, which is equipped with collision avoidance functionality. Obstacles at sea are difficult to see by the operator and therefore, this system is better commanded at full automation. Table 1-5 illustrates the automation levels available in each of the platforms.

Table 1-5: Level of Automation for Each of the Robots.

Level of Automation per Robot								
	Long- Endurance Fixed-Wing	Outdoors Multi- Rotor	Indoors Multi- Rotor	Large UGV	Small UGV	U-Ranger USV	ROAZ USV	Rescue Capsule
Level 1								
Level 2								
Level 3								

1.2.4 Multi-Robot Cooperation

An effective heterogeneous team management requires the capability to do reasoning about the mission goals in order to provide a task-to-robot decomposition. This tasks allocation must take into account the current capabilities and constraints of each asset in the team. Different strategies for the cooperation are feasible and, therefore, different requirements may be placed on the interface in order to implement these strategies. A heterogeneous team usually contains a set of vehicles with diverse capabilities that can therefore play different roles in the mission. Concepts such as roles, responsibilities, modes of operations and tasks may be part of the standard interface that supports the fleet interoperability.

The platforms involved have been carefully selected to help each other. In other words, they play complementary roles. Several platforms grouped together form a team. Each vehicle has been designed to provide a set of specific functionalities but they can address more complex missions by supporting each other.

Different strategies for the coordination are feasible. A strong end-users need is that any planning decision must be authorized by the on-site operations coordinator. Therefore, according to a traditional classification of multi-robot systems based on the coordination strategy [11], We follow a supervised, weakly-coordinated, centralized approach where the cooperation and interaction between robots is negotiated during mission planning. The planning, coordination, and therefore the ultimate responsibility falls on the Team Operator and occurs at the C2I. Therefore, this coordination approach relaxes to a certain extent the need to have multi-robot related concepts in the interoperable interface. The C2I encapsulates this functionality and can interact with each asset individually. However, a standard for coordinated multi-robot operations remains extremely relevant and was taken into account in the analysis described in the next section.

Some of the key concepts to be unambiguously defined as the basis for efficient mission planning are goal, role and task. A mission goal refers to the overall objective that the fleet must accomplish, for instance, the assessment of a disaster area. The mission planner is responsible for coordinating the fleet and allocating specific roles to each robot. A role defines the robot's behavior and its interactions with other members of the fleet or with humans. A task is the basic unit describing the actions requested from a robot. Typically, the role defines which tasks a robot should and should not execute. A robot is defined by the type of robot and its capabilities. For instance, the long-endurance is a fixed-wing aerial platform with surveillance, mapping, victim detection and communications relay capabilities. These characteristics define the set of tasks that it is able to perform, and therefore the roles it can take. A mission plan is therefore built upon the concepts of roles, tasks, and responsibilities.

Planning flow mirrors the concept of operation of international Search and Rescue teams. Table 1-6 illustrates how goals to roles and task decomposition occurs on the field.

Table 1-6: Goals to Roles and Tasks Decomposition.

System	Responsibilities
C2I	Sectorization + Mission Goals definition SAR Team Allocation to sectors Robot(s) Allocation to Teams Teams Monitoring and Control
RC2	Operations scheduling Roles Allocation Robot Task Planning Robots Monitoring and Control
Robot	Task Plan Execution Progress and Status Report

One of the core services required from all the platforms is the dynamic discovery of features. It allows robots to advertise their capabilities over the network, enabling dynamic planning and supervision from the C2I based on the current state of the team. A robot may take different roles during a mission depending on the responsibilities that the C2I allocates to it. The allocation of mission goals to predefined roles, the decomposition of these roles into tasks, and the configuration of these tasks for a specific robot model are the responsibility of the mission planner. Some predefined profiles are available to facilitate this task. Whereas roles influence the robots behavior, tasks influence the actions that robots perform. They are defined as a set of actions. Each task could be decomposed into subtasks. This subdivision could continue iteratively until a primitive task is reached. Table 1-7 shows some examples of the roles defined in the concept of operations.

Table 1-7: Examples of Robot Roles.

Roles	Description	Modes
Scout	Provides a quick assessment of an unexplored area or route.	Overview of an entire disaster zone (fixed-wing UAS). Traversability / best route exploration (UAS).
Surveyor	Scan in detail an area or building to support a thorough assessment and inspection (structural integrity, victims, hazards, etc.).	2D/3D geo-referenced map of the entire disaster zone as basis for sectorization (fixed-wing UAS). High-resolution 2D/3D geo-referenced map of a sector (fixed-wing UAS at higher altitude, rotor-craft at lower altitude) or a structure (rotor-craft). Building indoor inspections (small rotor-craft and small ground vehicle).
Observer	Steady target observation and assessment, including victims and structures.	Steady hover over a target (rotor-craft), including harsh weather conditions. Victim medical assessment outdoors (rotor-craft and USV) and indoors (small rotor-craft and ground vehicles).

Roles	Description	Modes
Searcher	Victims search.	Outdoors human detection on IR (UAS and USV), indoors (small rotor-craft and UGV).
Rescuer	Support to victim rescue.	Helps victim to escape from hazard areas (large ground vehicle) or support human rescuers.
Deliverer	Safety kit delivery. Robot delivery.	Delivery of a survival kit to a victim, aerial (rotor-craft) or terrestrial (UGV).
Cruiser	Travel to a destination.	All platforms when transiting to a new location where another role is enabled. The larger platforms may also act as a carrier of tools, debris and smaller robotics assets.

1.3 ANALYSIS OF EXISTING STANDARDIZATION INITIATIVES

ISO defines a standard as a set of “requirements, specifications, guidelines or characteristics that can be used consistently to ensure that materials, products, processes and services are fit for their purpose” [12]. In the context of interoperability, a standard shall unambiguously define data types, message and rules to implement the protocol. The analysis of the existing standardization initiatives shows that there exist several predominant initiatives for interoperability of unmanned systems [13]. However, harmonization among them is not yet a fact.

In the context of this analysis, we divided the different initiatives in two different groups:

- Fully operational standards; and
- Partially operational resources.

The first group focuses on systems interoperability, providing a common communication framework between different agents. They provide all the basic functionality required for a multiplatform system. The second group includes initiatives that are, either very popular on specific fields, or are designed specifically for some particular tasks or domain. Most of them show some relevant contributions but they do not provide interoperability for all the possible types of platforms, systems and range of application.

The lack of a single standard of reference for interoperability of unmanned systems makes any choice difficult since it will have an impact one way or another on legacy platforms. However, some alternatives may fit better for a given set of requirements. Harmonizing the existing standards, by combining them into one, or by proposing a brand new standard, would obviously solve most of the problems, but it would have serious implications both in industry and other programs that have adopted them as their standard [13]. This is clearly beyond the possibilities of the RTG in itself.

Among the studies, two candidates stood out from the rest, STANAG 4586 [14] and others related, and the Joint Architecture for Unmanned Systems – JAUS [15]. They are both stable, widely used and complete. STANAG pays a strong attention to the Intelligence, Surveillance and Reconnaissance (ISR) data while JAUS is instead more devoted to command and control interfaces of the platforms, robot navigation and perception.

In this context, both were created to address specific requirements in different domains. STANAG related standards are predominantly military and, even though they have been promoted for civil applications, their requirements are heavily demanding in terms of compliance. STANAG 4586 is mostly focused on UAVs, even though some other types of unmanned systems have been developed to meet this standard. It is perhaps very relevant for the interoperability of military assets across the different NATO members, but it is hard to be adopted by civil or research platforms without a strong investment. For instance, certifying a small

multi-rotor UAV for the STANAG 7085 Interoperable Data Links for Imaging Systems is costly and probably a barrier for small platforms providers. Furthermore, the geographical constraints (NATO only), the focus on the bigger systems and the absence of open available implementation make this option less convenient. Likewise, JAUS was originally designed for UGVs. It is fair to say that JAUS has done great efforts to extend the coverage to any type of platform, and it currently considers any unmanned system as a generic asset in order to become truly multi-domain. Its roots are also military, but it was soon transferred to the Society of Automotive Engineers (SAE International) where it is currently hosted.

According to our analysis, JAUS is fairly aligned with the needs of small unmanned platforms in terms of the interoperability described in Section 2. Also, JAUS has been successfully demonstrated in recent years for collaborative UAV-USV cooperative missions [15]. A quite direct traceability between needs and the JAUS service sets is easily derived. It is already compatible with popular transport protocols (TCP, UDP, serial) independently of the communication link beneath it, which makes it more flexible. And it is already multi-environment (air, ground and maritime). There exist both commercial and open-source implementations. Unfortunately, there is a fee to access the JAUS documentation which may prevent some providers from using it. Nevertheless, the cost is deemed reasonable.

There are many other initiatives with strong support in different communities. According to the principles and needs for standardization defined above, these are considered software frameworks and middleware rather than full standards. For instance, the Robotic Operating Systems (ROS) is used nowadays in many multi-robot systems. However, open-source initiatives are open and flexible by definition which may not provide the expected reference specification for future developments. These initiatives definitely add a lot of value to the development of small unmanned systems, but they do not formally satisfy the interoperability requirements like the standards mentioned previously. They should remain at the platform level and the platforms should comply with an external interoperability standard. It is the scope of the interoperability work to harmonize this heterogeneity into a single standardized protocol.

1.4 INTEROPERABILITY STANDARD

The standard interface for interoperability of heterogeneous fleets is based on the Joint Architecture for Unmanned Systems (JAUS) [16]. JAUS is a Service-Oriented Architecture (SOA) that specifies a list of services commonly found in robotics. The interface describes the subset of standard messages that will be used in the scenario and specifies all the details required to comply with the interface.

1.4.1 Services Sets

The interoperability interface is a Service-Oriented Architecture (SOA). The most common services for unmanned systems interoperability are already defined in JAUS as a set of advanced standards. They are grouped as “Service Sets”. The following ones will be used:

- Core Service Set (SAE AS5710 [17]): essential services such as transport, events, discovery, etc.
- Mobility Service Set (SAE AS6009 [18]): mobile platforms services.
- Environment Sensing Service Set (SAE AS6060 [19]): platform-independent sensor capabilities.
- Manipulator Service Set (SAE AS6057 [20]): platform-independent capabilities common across all serial manipulator types.

The concepts defined in the data model can be matched against specific services in this architecture. Figure 1-1 shows the specific services used in a real operation.

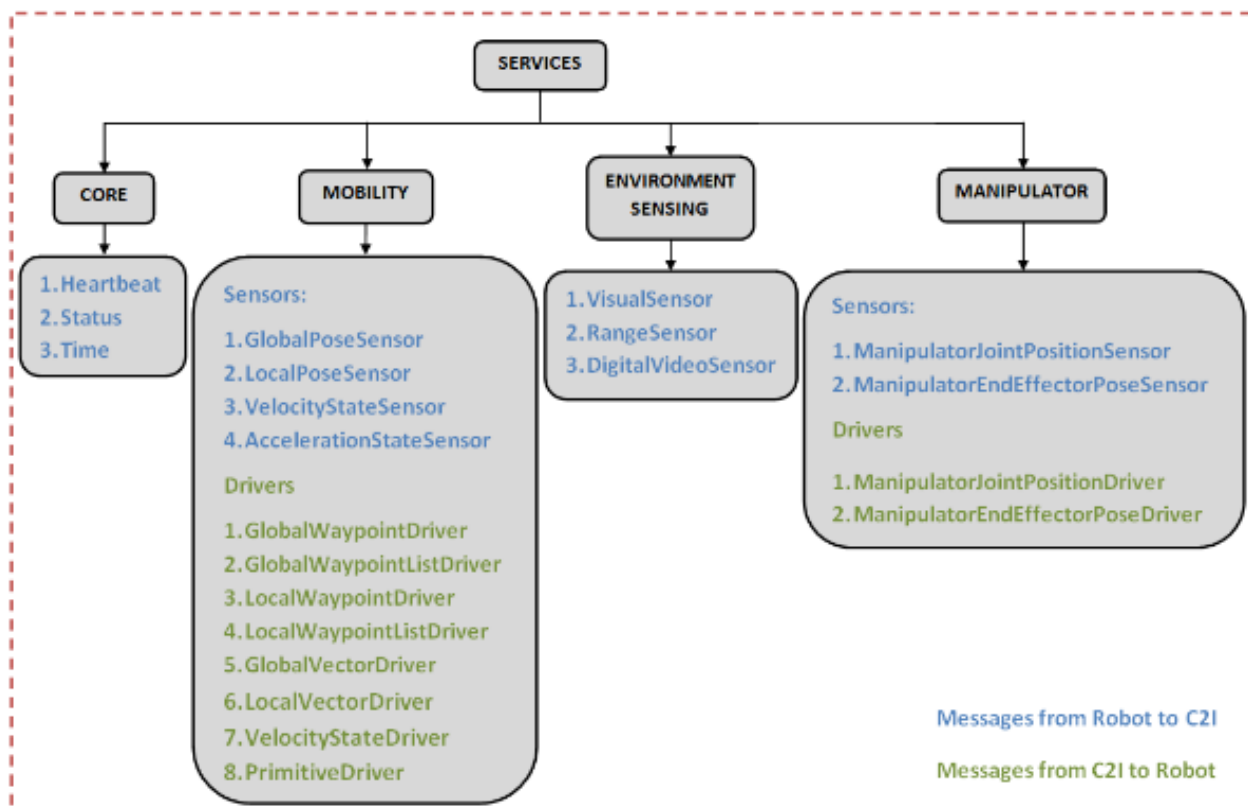


Figure 1-1: Relevant JAUS Services.

However, as we progressed with all the integrations in the project, we discovered that some of the functionalities provided by some of the platforms were not supported by these standard services. We refer to this as the gap analysis. Table 1-8 shows some of these gaps.

Table 1-8: Interface Gaps.

Gaps Analysis				
	Outdoor Quadrotors	Indoor Quadrotors	Ground Robots	Large Sea Vehicles
Survival kit deployment				
Rescue Capsule deployment				
Platform-specific components enable/disable				
Platform extended status				
Manipulator tool selection				
Voice transmission				

Given this, a new set of non-standard services has been defined to fill the gaps of the standard. This new non-standard service set is shown in the following Figure 1-2.

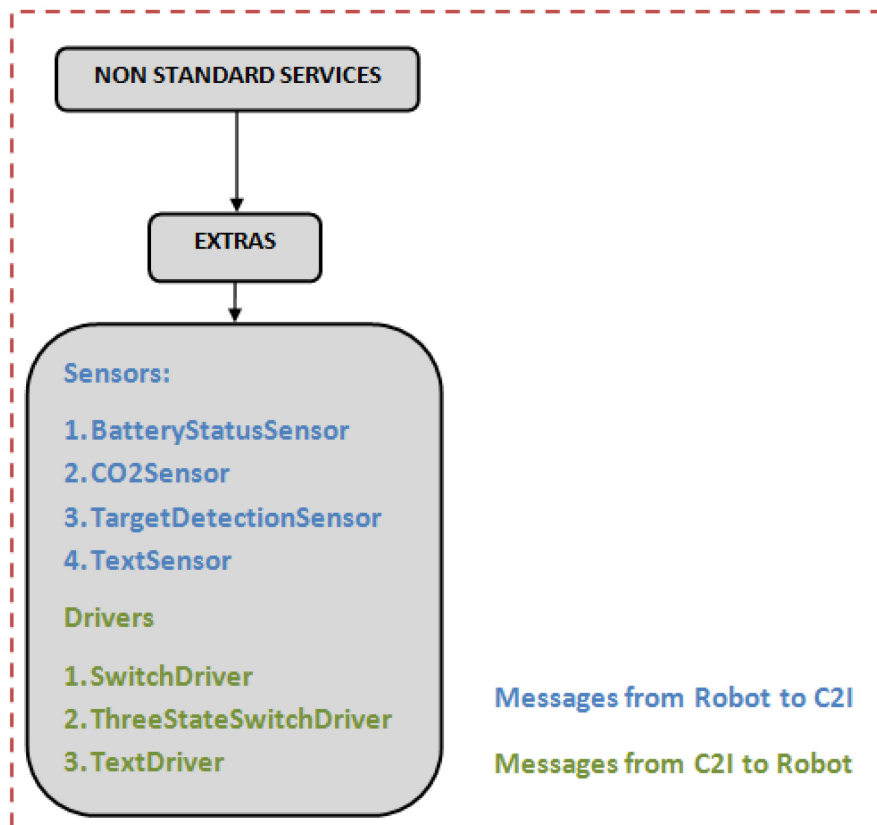


Figure 1-2: Additional Non-Standard JAUS Services.

For each service, a strictly defined message-passing interface (vocabulary) and protocol (rules) for data exchange are available. There are generally three types of messages: Query, Report and Command. Furthermore, the transport service (from the Core Services set) acts as an interface to the transport layer. Therefore, the interface is, in principle, independent from the physical transport layer. However, the current implementation is only available for the UDP protocol.

JAUS also defines a hierarchical and flexible topology built up of subsystems, nodes and components. For the implementation of JAUS, the following assumptions have been made:

- A team is considered a System;
- Each platform is defined as a Subsystem with a single Node. Therefore, all components within the same platform will share the subsystem and node identifiers; and
- As described later, a Node may contain several Components. But a Component will implement only one Service, plus the Core Services which are always present. This restriction allows the C2I to dynamically discover each of the Services available on each robot.

Therefore, a system is depicted in Figure 1-3.

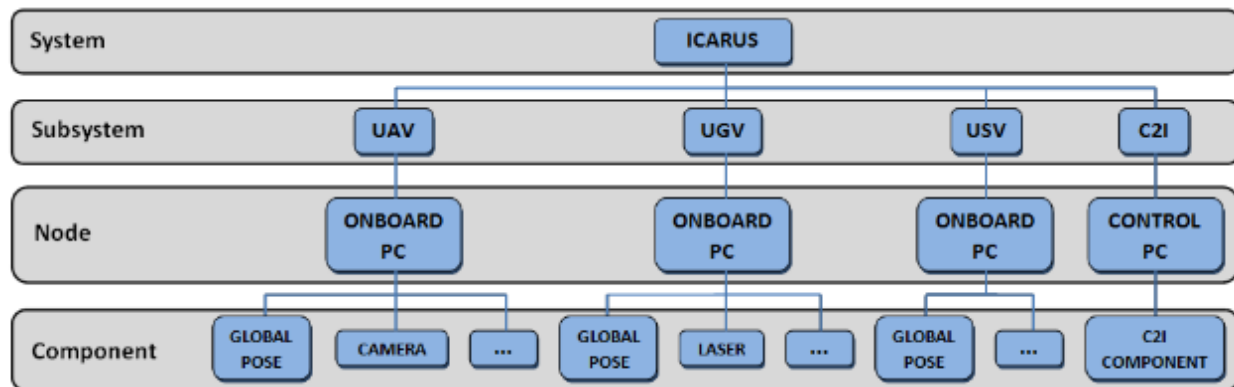


Figure 1-3: JAUS Topology.

1.5 AN INTEROPERABLE LAYER: THE LIBRARY

All this functionality is provided to the robotics partners as a software library referred to as the interoperability layer. This module acts as a bridge between their internal and external development frameworks. This interoperability layer is also responsible for the integration of the communication network and the Command and Control Station on each individual platform.

A set of C++ classes have been designed to integrate the vehicles into the network. The JAUS-specific functionality has been encapsulated within them. To comply with the interface, a system may directly integrate this library (native integration). However, most robotics systems nowadays are based on either proprietary or open-source middleware (such as ROS). To accommodate these systems into a compliant network, an alternative is to implement an adapter to the robot-specific middleware (translator). The diagram in Figure 1-4 illustrates both cases, native integration (Robot C) and through an adapter (Robot A using ROS, and Robot B using MOOS).

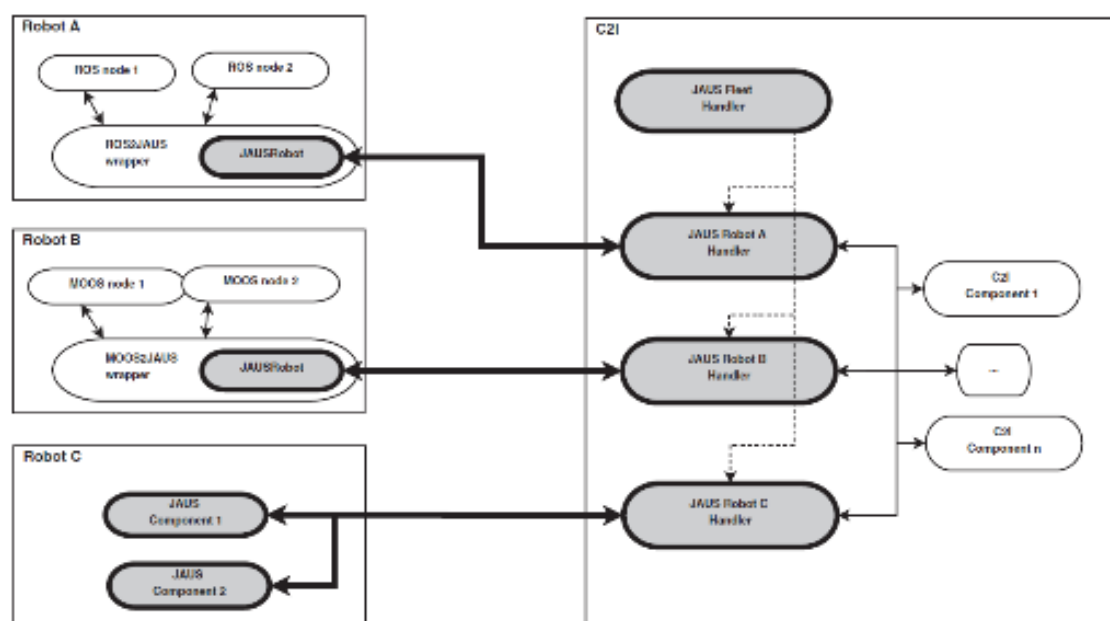


Figure 1-4: Robot Adaptation Strategy.

The following sections describe the software classes encapsulated within the library and depicted in the previous diagram.

1.5.1 JAUSRobot

JAUSRobot encapsulates all the functionality required on-board the vehicle. It represents a subsystem in the JAUS topology (see Figure 1-3). In the JAUS interface, a subsystem will contain only one node. This approach will allow us to provide a component name to each service. Therefore, all components within the same robot share the subsystem and node identifiers.

There are two types of services available on a JAUS robot in addition to the core services: Sensors and Drivers.

Sensors provide access to information generated on the robot (i.e., Global Pose, Image, etc.). There are essentially two types of C++ functions required to integrate this functionality:

- 1) Add Sensor: a JAUS component is added to robot subsystem. For example, if the instance of JAUSRobot is myRobot the following lines are adding a GlobalPoseSensor service:

```
JAUSRobot myRobot;  
myRobot.AddGlobalPoseSensor("OEMStar_GPS", AT_1_HZ).
```

- 2) Set data: updates the data associated to a service. For the example above, the following lines will update the current GlobalPose of myRobot:

```
JAUS::GlobalPose globalPose;  
myRobot.globalPoseSensor->SetGlobalPose(newGlobalPose).
```

Drivers, on the other hand, provide access to actuation capabilities provided by each robot (i.e., go to waypoint). There is an equivalent function required to integrate this functionality:

Add Driver: a JAUS component is added to the robot subsystem. For example, if the instance of JAUSRobot is myRobot the following line is adding a AddGlobalWaypointDriver service to receive waypoints request in global coordinate frame:

```
JAUSRobot myRobot;  
myRobot.AddGlobalWaypointDriver("global_waypoints", authority_code).
```

The authority code parameter of these services is used for pre-emption and needs to be set lower to the one of the clients accessing the driver. Otherwise, commands from the client are ignored.

Therefore, JAUSRobot creates a JAUS component for every new Sensor and Driver. This allows the JAUSFleetHandler class to discover and manage each of them independently.

The JAUS interface is based on call backs for message reception. One more function will register a local call back in order to receive any message coming from the JAUS network:

```
void localProcessMessage(const JAUS::Message* message){};  
myRobot.RegisterJAUSMessageCallback(localProcessMessage).
```

1.5.2 JAUS C2I

On the C2I side, two classes have been designed.

1.5.2.1 JAUS Fleet Handler

JAUSFleetHandler encapsulates all the functionality related to the fleet management. It includes the functionality to discover subsystems and services on the JAUS network, retrieve their names and their current status. For example, if the instance of JAUSFleetHandler is myFleet, the following line allows discovering all subsystems on the JAUS network and retrieving their services names:

```
myFleet.DiscoverFleet().
```

On the other hand, the following line allows also checking for system updates:

```
myFleet.RefreshFleet().
```

In terms of JAUS, it represents a basic JAUS component implementing the Discovery service to retrieve the subsystems available in the network and their services.

1.5.2.2 JAUS Robot Handler

JAUSRobotHandler is responsible for managing a single robot. After the discovery process, an instance of this class must be created and configured. This class will interface directly to the JAUS Robot.

In terms of JAUS, it represents a basic JAUS component. For each sensor service available on the real robot, it creates an event of type Every change. This is the JAUS mechanism to configure the sensor service on-board the robot to send data for every new Set. Periodic events will also be available in the future.

On the other hand, for each driver service available on the real robot, it configures the access control service needed to send commands.

1.5.2.3 Management of the Communications

We tackle interoperability at different levels. This chapter focuses on the interoperability layer. This is a Software Defined Protocol (SDP) that can run over any compatible communication layer underneath. However, we also address the interoperability at the communications level.

A tight cooperation between the interoperability and communications layer allows for a smart management of the network. The communications are a cognitive self-organizing multi-node network. This layer exposes an interface to configure the required data flows, its priorities and other details. Given the current set and number of robots, sensors and the priorities for the mission assigned, the communications layer can assure a Quality of Service for each data stream.

This is traditionally preconfigured manually for each mission. A tight integration between the interoperability and communications layers has enabled the dynamic self-configuration of the team. A team management node runs within the C2I and exploits the discovery mechanism to retrieve all robots and their capabilities. This information is transferred to the communications layer that organized the data flows based on a priori defined priorities. For instance, telemetry and telecontrol streams are given the highest priority since they are safety critical. For each robot, a camera is given the medium priority and all other sensors are lower priority. This a priori priority allocation depends on the number of robots and their characteristics.

These configuration capabilities are also exposed to the C2I and therefore to the operator. The coordinator can at any time change the priority levels, enable new sensors, disable sensors that are not required, etc. The user is also informed with the current status of the network and he is therefore able to change the configuration if the network is overloaded.

Eight messages have been defined to exchange all the information needed between both interfaces (COMMS and JAUS):

- Register Message.
- Activation of a stream.
- Deactivation of a stream.
- Deactivation warning.
- Network status notification.
- Robot's register notification. This is a notification sent from the COMMS interface to the JAUS interface. It is used to ask for the robot's register message presented before. It is needed when, for instance, the COMMS interface starts running later than the JAUS interface.
- Robot's unregister notification (COMMS to JAUS). It is sent when the COMMS interface loses a Robot.
- Robot's unregister notification (JAUS to COMMS). It is sent when the JAUS interface loses a Robot.

1.6 INDIVIDUAL PLATFORMS ADAPTATIONS

All platforms have been adapted to the interface. This automatically ensures the compatibility with the C2I, enabling the multi-robot coordination and combination of data as later described in this report.

1.6.1 ROS to Robot Node

All aerial platforms within the project share a similar approach when it comes to software and hardware design. The hardware setup comprises a low-level board, responsible for the flight control of the vehicle (i.e., autopilot) and, optionally, a high-level board (i.e., on-board pc) responsible for the autonomous navigation and payload data. These autopilots communicate with the vehicle-specific ground station through the MAVLink protocol [21]. The on-board PCs, instead, run the Robot Operating System (ROS). A template has been developed to integrate ROS-based platforms. Therefore, all of them have been adapted using this template.

This template however should be configured to the specific characteristics of each platform, particularly in terms of sensors equipment. The proposed strategy is to implement a ROS-based wrapper to subscribe to ROS topics and interface the protocol. This node is intended to run on-board the robot and provides a wrapper of the interface. The node is implemented within the robot.cpp file.

All the required services described in Figure 1-1 are available for ROS-based systems through this template launch file. These services can be easily enabled by configuring a template ROS launch file. The following XML excerpt shows an example for the specific case of a Global Pose service:

```
<?xml version="1.0"?>
<launch>
  <node pkg="ros2jaus_node" type="robot" name="robot" output="screen">
    <!-- ROBOT CONFIG -->
    <param name="subsystemName" type="string" value="ctae_robot"/>
    <param name="subsystemID" type="int" value="99"/>
    <param name="nodeID" type="int" value="1"/>
    <!-- GLOBAL POSE -->
    <param name="globalPoseEnable" type="bool" value="true"/>
  </node>
</launch>
```

```
<param name="globalPoseSensorName" type="string" value="global_pose"/>
<param name="globalPoseUpdateRate" type="double" value="25"/>
<param name="globalPoseTopicName" value="/EURECAT_robot/global_pose"/>
</node>
</launch>
```

Therefore, an ROS-based robot just subscribes to a set of topics, with a predefined message type. An analysis of the existing ROS messages was performed to select the most appropriate interface definition. When an existing ROS message was deemed both valid and correct, this message was used. However, there were certain cases where the definition of the messages in ROS was either not existing, ambiguous or not valid. In these cases, a new message type was defined under the package `icarus_msgs`. The topic names and update rates can be configured from the ROS launch file:

- `/global_pose` (`icarus_msgs/GlobalPoseWithCovarianceStamped`).
- `/local_pose` (`geometry_msgs/PoseWithCovarianceStamped`).
- `/velocity_state` (`geometry_msgs/TwistWithCovarianceStamped`).

And publishes:

- `/local_waypoint` (`icarus_msgs/LocalWaypointStamped`).
- `/global_waypoint` (`icarus_msgs/GlobalWaypointStamped`).

1.6.2 Other Robot Adapter

The control systems of the ground platforms are implemented using FINROC, a middleware developed by the Robotics Research Lab at the University of Kaiserslautern. The rescue capsule runs OceanSys and exposes a data repository over which any software in the same network can subscribed and receive custom messages. ROAZ implements a similar system, but it is also ROS capable. The U-Ranger on-board autonomy is developed in MOOS and implements a behavioral control strategy.

A template example was provided to each of the partners, together with the use case for ROS and some documentation. Each partner was able to quickly adapt their existing frameworks to the standard interoperable interface and become complying with all team technology, such as the communication infrastructure and the C2I. Table 1-9 and Table 1-10 provide a summary of the services provided by each system.

Table 1-9: Services Provided by Each Vehicle.

Services per Robot								
Service	AROT	ASOLAR	U-Ranger	ROAZII	UCAP	firefly	SUGV	LUGV
Global Pose Sensor	GlobalPose (20 Hz)	GlobalPose (20 Hz)	GlobalPose (20 Hz)	GlobalPose (20 Hz)	GlobalPose (20 Hz)	GlobalPose (20 Hz)	GlobalPose (20 Hz)	GlobalPose (20 Hz)
Velocity State Sensor		Velocity State (20 Hz)						
First Camera	Left Camera (20 Hz)	Visual Camera (20 Hz)		Visual Camera (20 Hz)	Visual Camera (20 Hz)	Left Camera (20 Hz)	Front Camera (20 Hz)	Front Camera (20 Hz)

Services per Robot								
Service	AROT	ASOLAR	U-Ranger	ROAZII	UCAP	firefly	SUGV	LUGV
Second Camera	Right Camera (20 Hz)	Thermal Camera (20 Hz)		Thermal Camera (20 Hz)		Right Camera (20 Hz)	Manipul. Camera (20 Hz)	Manipul. Camera (20 Hz)
Third Camera	Thermal Camera (20 Hz)					Thermal Camera (20 Hz)	Rear Camera (20 Hz)	Gripper Camera (20 Hz)
Fourth Camera	Visual Camera (20 Hz)							
Range Sensor (Point Clouds)			Laser	Radar		PointCloud	PointCloud	PointCLoud
Global Waypoint List Driver	Global Waypoints	Global Waypoints	Global Waypoints	Global Waypoints	Global Waypoints		Global Waypoints	Global Waypoints
Primitive Driver	Cmd Velocity		Cmd Velocity		Cmd Velocity	Cmd Velocity	Cmd Velocity	Cmd Velocity
Manipulator End Effector Pose Sensor							EndEffector Pose	EndEffector Pose
Manipulator Joint Sensor							Joint Position	Joint Position
Manipulator End Effector Driver							EndEffector PoseControl	EndEffector PoseControl
Manipulator Joint Position Driver							Joint Position Control	Joint Position Control

Table 1-10: Custom Services Provided by Each Vehicle.

Custom Services per Robot								
Service	AROT	ASOLAR	U-Ranger	ROAZII	UCAP	firefly	SUGV	LUGV
First Switch Driver	Delivery Kit				Deploy UCAP		Lights	Lights
Second Switch Driver					InflateRaft		Manipulator	Manipulator
Third Switch Driver							Reset	Reset

Custom Services per Robot								
Service	AROT	ASOLAR	U-Ranger	ROAZII	UCAP	firefly	SUGV	LUGV
Fourth Switch Driver							Audio	Engine
Fifth Switch Driver							Speech	ToolLock
First 3-State Switch Driver							Gripper Control	Gripper Control
Second 3-State Switch Driver								Tool Selector
Text Sensor							Text	
Text Driver							TextCmd	
CO2 Sensor							CO2 Sensor	
Robot Extended Status	Battery Status	Battery Status			Battery Status		Battery Status	Battery Status
Video Stream			Video Stream					
Target Detection	Victims	Victims		Victims			Victims	Victims

1.7 FIELD VALIDATIONS: EXAMPLES OF MULTI-ROBOT COOPERATION

The approach to interoperability was initially verified with a set of in-lab integrations and simulated tests. Once the robotics platforms were finalized and ready for field tests, a series of field operations involving different combinations of pairs of air, ground and sea vehicles were organized during the integration trials carried out between July and September, 2014. The purpose was twofold:

- 1) Verification of the completeness, correctness and feasibility of the interoperability interface; and
- 2) Experimentation on the possibilities on multi-robot cooperative Search and Rescue missions.

The results from one of these trials showed that the work on interoperability enabled large scale cooperative mapping with multiple aerial and ground robots in Urban Search and Rescue [22].

The final field validation was carried out together with the final integration and demonstration exercise of the ICARUS project. Three full-team validations were performed during the final project demonstrations:

- The maritime trials and demonstration in Alfeite, Lisbon (Portugal) in July 2015;
- The land trials and demonstration in Marche-en-Famenne (Belgium) in August 2015; and
- The participation in the euRathlon competition in September 2015 where the project received the Best Multi-Robot Coordination Award by the IEEE Robotics and Automation Society (RAS).

At this stage, all platforms had been integrated into the ICARUS system. After start-up, the current capabilities of the team could be dynamically discovered and the C2I automatically configures itself, allowing a Team Operator to plan a mission, assigning roles and tasks to each system. During the mission, all the information flows and the current network status is displayed. The operator can follow the progress of the mission, enable, disable or change the update rate of each of the services. The operator can, at any time, request new missions, take manual control of the platforms that provide this service, resume previous missions, etc. All this functionality was exercised and demonstrated during the validations.

Together, these large scale operational exercises completed the validation of the interoperability standard interface.

Some examples of the multi-robot collaboration experimented are described in the following subsections to illustrate the possibilities of multi-robot cooperation provided by an interoperable team.

1.7.1 Cooperative Multi-Stage Aerial Surveillance

In this multi-robot collaboration concept, the overall mission imposed by the human commanding officer is to provide a general assessment of a predefined sector. This is a typical scenario to be performed when relief agencies arrive on a crisis site. Assets to be deployed for this task are the fixed-wing UAV and the outdoor multi-rotor, both with clearly distinctive roles:

- The fixed-wing aircraft acts as a surveyor system which covers the entire area quickly, flying at an altitude of around 100 m. Note that the altitude limitation is deliberative, as many countries impose a maximum flight altitude of 400 feet (133 m) for unmanned systems and it was our specific target to test the operational capabilities within realistic legislative bounds. Flying at this altitude, the aircraft quickly provides a general low-resolution assessment of the sector, such that areas of interest can be selected.
- The multi-rotor aircraft also acts as a surveyor system, but as it has a lower flight altitude of typically 40m, it covers only a much smaller area. It is therefore used to provide a high-resolution assessment of an area of interest, as identified by the fixed-wing assessment.
- The multi-rotor aircraft also acts as observer system to provide high-resolution multi-view observation of points of interest (victims, buildings, etc.).

Operating multiple unmanned aerial systems in the same airspace is not easy from a safety perspective. In this case, vertical separation of the airspace was used for segregating the operations with the fixed-wing and multi-rotor aircraft. The operators were also in constant contact with one another to synchronize the landing operations.

Figure 1-5 shows the different aerial systems in the air at the same time, whereas Figure 1-6 shows the outcomes: the lower-resolution assessment by the fixed-wing aircraft and the high-resolution assessment by the multi-rotor aircraft. As all data is geo-referenced, this information can be perfectly superimposed on one another.

1.7.2 Aerial Scouting for Traversability Analysis

A common problem for relief teams is that the route they have to take due to their designated destination cannot be reached using the “normal” routes, as roads are blocked by debris or by floods. The mission resulting from this use case is to use a multi-robot team to ensure the traversability of a route and provide an early identification of threats. The assets deployed for this mission are the fixed-wing and outdoor multi-rotor aircraft and the relief team on the ground, including unmanned ground vehicles. The aircraft scan the area to detect blocked and cleared routes to the destination point

and send updated navigation information to the ground team, such that the ground team can travel to the destination as quickly as possible. Figure 1-7 shows the multi-rotor aircraft flying ahead of the ground team, searching for obstacles on the way to the destination.



Figure 1-5: Multiple Aircraft in the Air at the Same Time.

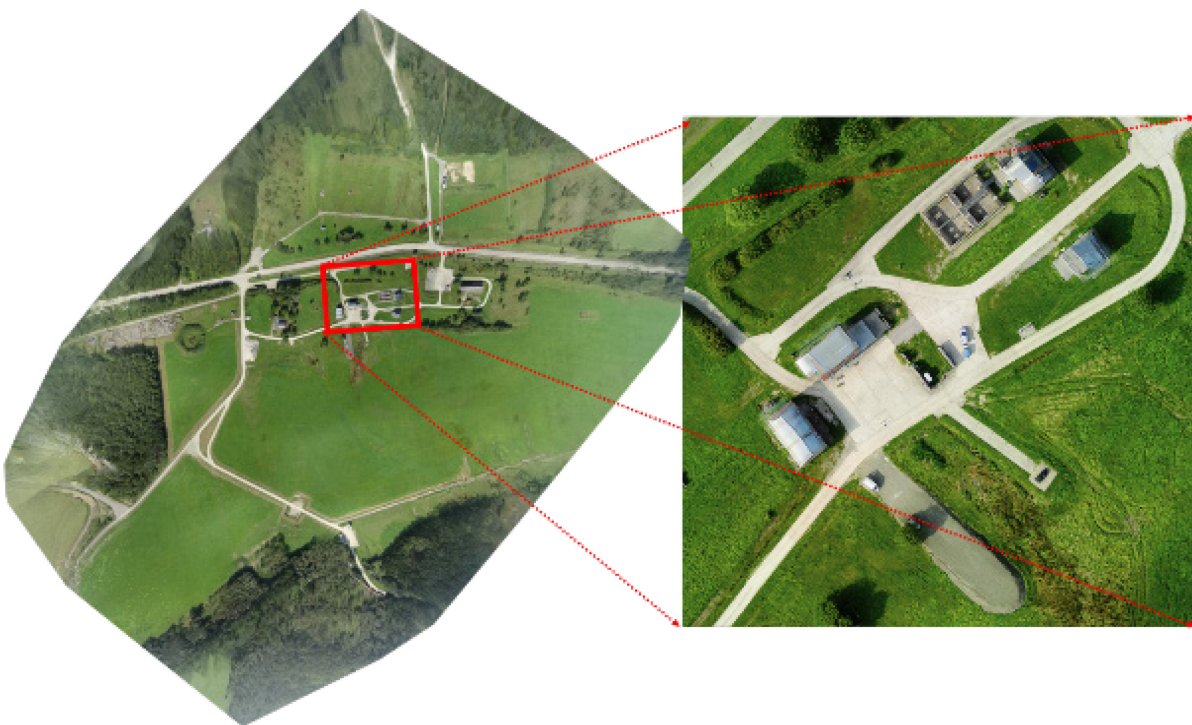


Figure 1-6: Low-Resolution Fixed-Wing Assessment and High-Resolution Multi-Rotor Assessment.



Figure 1-7: Outdoor Rotor-Craft Ensuring Optimal Routing of Ground Team.

1.7.3 Victim Search

Victim search is a primary mission in any rescue operation. Following the typical mission profile, a search area is defined and a multi-robot collaborative search is ordered by the human commanding officer in this predefined area. Multiple collaboration modalities are possible, depending on the Search and Rescue context:

- In outdoor Search and Rescue scenarios, the fixed-wing and outdoor multi-rotor aircraft are deployed. The fixed-wing aircraft can quickly provide a scan of a large area, either clearing the area or indicating preliminary detections, which then need to be confirmed by the outdoor multi-rotor aircraft. The latter then confirms the location and status of the detected victims and can count the number of victims. This operation can take place in an urban search and rescue context, where e.g., victims are to be sought in rubble fields after an earthquake (as shown on Figure 1-8), or in a maritime Search and Rescue context, where victims have to be found in the water (as shown on Figure 1-9).

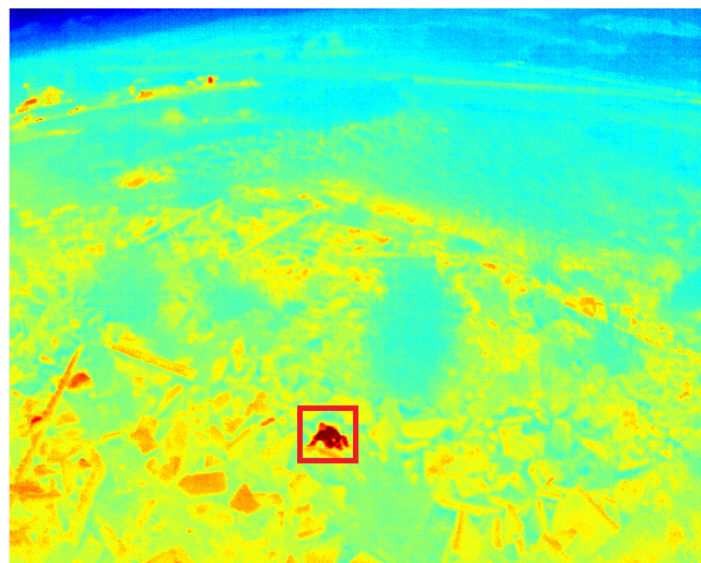


Figure 1-8: Automated Victim Detection on a Rubble Field in an Urban Search and Rescue Operation.



Figure 1-9: Victim in the Water Being Localized by Outdoor Multi-Rotor Aircraft.

- In indoor urban search and rescue scenarios, the indoor multi-rotor aircraft and the small unmanned ground vehicle are deployed to collaborative search for surviving victims inside semi-demolished buildings, as shown on Figure 1-10.



Figure 1-10: Collaborative Indoor Victim Search.

- In maritime Search and Rescue scenarios, the survival times in the water are often very short. Therefore, in an attempt to limit the time-delay between the search phase and the rescue phase in the relief operation, the fixed-wing and outdoor multi-rotor aircraft are deployed, together with the unmanned surface vehicles and the unmanned capsules. This enables the unmanned aircraft to immediately steer the maritime vehicles towards the victims detected and localized by the aircraft. The unmanned surface vehicles also have sensors (infrared cameras) enabling victim detection on board, but as these are relatively small platforms, their field of view in rough sea conditions with high waves is limited. Collaborative victim search between aerial and marine platforms is therefore not impossible, but the greatest benefit of a mutual deployment lies in the combination of the search and the rescue aspect, as illustrated by Figure 1-11, which shows a victim in the water being tracked and localized by the outdoor multi-rotor aircraft, thereby guiding the unmanned surface vehicle to a position in the neighbourhood of the victim, such that an unmanned capsule can be deployed which can inflate a life raft close to the victim in order to save the person.



Figure 1-11: Collaborative Maritime Victim Search and Rescue Operation Involving Aerial and Maritime Platforms.

1.7.4 Carrier

During any Search and Rescue operation, many assets need to be deployed as quickly as possible. This mission profile shows how collaborative robotic agents can help by acting as carrier platforms for small assets and equipment and also for other unmanned systems. As an example, the large unmanned ground vehicle acts as a carrier platform for the small unmanned ground vehicle and both the ROAZ II and the U-RANGER acts as a carrier for the rescue capsules. They not only enable the cargo to be transported to the destination, without any extra burden to the human relief workers, but also act as deployment systems for the smaller unmanned systems.

As an example, Figure 1-12 shows how the large unmanned ground vehicle deploys the small unmanned ground vehicle on the top of a building, whereas Figure 1-13 shows how the rescue capsules are deployed from an unmanned surface vehicle.



Figure 1-12: Large Unmanned Ground Vehicle Deploying the Small Unmanned Ground Vehicle on the Top of a Building.



Figure 1-13: Unmanned Surface Vehicle Deploying the Unmanned Rescue Capsule.

1.7.5 Communications Relay

In the event of a large crisis, previously existing communication infrastructure is often broken or at least severely damaged. However, communication is crucial for having coordinated response operations. Collaborative unmanned systems can act as communication relay tools to extend the communication range over large distances. Of course, the assets which are most useful for this are the aerial tools, as they can

provide line-of-sight communication relay over large distances. In the ICARUS project, an ad hoc link-hopping network was developed, which allows to extend any communication link while the aerial platforms are in the air. This allows the fixed-wing aircraft and the outdoor multi-rotor aircraft to act as communication relays for the ground and marine rescue teams.

1.7.5.1 Air, Sea, Ground Cooperation During euRathlon 2015

Only seldom, rescue operations have to be performed which span three domains (air, ground, marine). However, the Tohoku earthquake and subsequent Fukushima disaster showed that response protocols were ill prepared to approach such multi-domain crises. Therefore, the euRathlon challenge focused specifically on this problematic. In brief, the mission imposed by the euRathlon challenge consists of detection, after a Fukushima-like incident, missing workers under water, outside on the ground and inside in a semi-demolished reactor building. Full information of the concept of operation is available online [23]. These Search and Rescue operations require the simultaneous and coordinated deployment of unmanned aerial, ground and underwater vehicles, gathering environmental data and performing real-time identification of critical hazards in a nuclear accident.

The team deployed for this purpose five robotic systems:

- 1) The outdoor multi-rotor was first deployed to search for the best route for the ground robots to reach the open entrance to the building. Then, it mapped the area in the RGB, gray and thermal spectrum. Finally, it performed real-time detection and localization of missing workers, leaks, etc. (Figure 1-14).
- 2) The Teodor UGV was used as carrier platform for the small UGV and for outdoor 3D mapping (see Figure 1-15 and Figure 1-16).
- 3) The small UGV was used for indoor detection and localization of missing workers, leaks, etc.
- 4) The ROAZ II vehicle was used as a carrier platform for the MARES unmanned underwater vehicle.
- 5) The MARES unmanned underwater vehicle was used for underwater detection and localization of missing workers, leaks, etc.

Even though behaviors specific to euRathlon, such as opening valves, etc., were not originally considered in the concept of operations, they were easily integrated as a proof of the flexibility of the followed approach towards interoperability. Thanks to the different levels of interoperability and automation, the specialized operator could take over at this point and tele-operate the system to finish the mission.



Figure 1-14: Rotor-Craft During the euRathlon Challenge.



Figure 1-15: Teodor and Small UGV During the euRathlon Challenge.



Figure 1-16: 3D Map of the Crisis Area, Obtained by Combining 3D Maps from the Aerial and Ground Platforms.

1.8 CONCLUSIONS AND ACKNOWLEDGEMENTS

The goal of the work performed by the Belgian actors in NATO RTG-149 was to provide a capability demonstrator to validate the compatibility chain from the acquisition of ISR data on unmanned vehicles (be it unmanned aerial, ground or maritime vehicles) up to the end user terminal of a C2 system. Due to synergy-effects with on-going EU-funded research, we chose a Search and Rescue application as primary application domain.

As described in this chapter, a strong effort was devoted to appraising the existing body of work in standardization of multi-robot systems. Given the particular requirements of this RTG, emphasis was placed on

initiatives considering multiple domains (air, ground and sea). Likewise, standards and methods applicable to smaller and lightweight platforms were prioritized. There have been several initiatives addressing both issues. However, harmonization of them is not yet a fact. There is still the need for a single multi-domain standard for interoperability, easily adaptable to both large and small systems. The contribution of the Belgian contribution focused on the selection of the most appropriate existing initiative (JAUS), the evaluation of its application to multi-robot Search and Rescue missions, the elaboration of recommendations for improvements, the adaptation of a set of Search and Rescue robots and the capability demonstration of the interoperable and heterogeneous team in three large scale demonstrations, exploring multi-robot cooperation and real-time centralized supervision and planning of an heterogeneous team.

The research leading to these results has also received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 285417.



Chapter 2 – INTEROPERABILITY STANDARDS OF UGVs AND TESTS OF INTEROPERABILITY OF UGVs AT ROBOT COMPETITION M-ELROB

A. Tiderko and Dr F.E. Schneider
Fraunhofer Institute for Communication
GERMANY

ABSTRACT

The main objective of this project is to validate interoperability standards of unmanned ground systems (UGVs) – or, if applicable, Unmanned Systems (UMS) in general – as proposed by NATO RTGs. The specified STANAGs and other standards are being implemented in real systems. The prototypical mission goal is the acquisition of e.g., ISR data, and the storage and publication of this data such that it is available to coalition members.

Different unmanned systems are used as experimental platforms to establish and verify these in-teroperability standards and also, they are gradually introduced into the robot competition M-ELROB.

2.1 PROPOSED UGV STANDARDS

STANAG 4586 (NATO Standardization Agreement 4586) is a NATO Standard Interface of the Unmanned Control System (UCS) Unmanned Aerial Vehicle (UAV) interoperability [1]. It includes data link, command and control, and human/computer interfaces. The current revision is STANAG 4586 Edition No 3 with mission phase enhancements, an updated list of vehicle identifiers, etc.

STANAG 4586, formally ratified by NATO in 2002, defines five levels of UAV interoperability:

- Level 1 – Indirect receipt/transmission of UAV-related payload data;
- Level 2 – Direct receipt of Intelligence, Surveillance and Reconnaissance (ISR) data where “direct” covers reception of the UAV payload data by the unmanned control system when it has direct communication with the UAV;
- Level 3 – Control and monitoring of the UAV payload in addition to direct receipt of ISR and other data;
- Level 4 – Control and monitoring of the UAV, less launch and recovery; and
- Level 5 – Control and monitoring of the UAV, plus launch and recovery.

The implementation of a normative set of references dedicated to ground robotics remains pending since there is no established standard in this field. The standard JAUS was promoted as a UGV standard, in particular to have generic control/command stations, but it seems to have been put on stand-by for several years now [1].

JAUS and STANAG 4586 are both message-based standards [1]. Consequently, duplication of effort exists in terms of capabilities and the domains targeted by both standards. For example, both JAUS and STANAG 4586 continue to introduce functionality in order to support Unmanned Surface Vehicles (USVs). However, STANAG 4586 is still primarily focused on UAVs and JAUS is primarily focused on UGVs.

Consequently, JAUS currently supports some different capabilities than STANAG 4586 in terms of representing the environment surrounding unmanned systems and providing functionality for avoiding obstacles. Another technical difference between the standards is that JAUS treats unmanned systems as generic assets, whereas STANAG 4586 employs Vehicle Specific Modules (VSMs) within the standard.

The NATO group IST-089 task was to define a set of standards for UGVs. It identified data formats used within MAJIC and agreed on a subset of these standards:

- STANAG 3277 – Air Reconnaissance Request / Task Form;
- STANAG 3377 – Reconnaissance Exploitation Report message;
- STANAG 3596 – Air Reconnaissance Requesting and Target Reporting Guide;
- STANAG 4545 – SAR, EO, and IR Data (NSIF);
- STANAG 4559 – NATO Standard Image Library Interface;
- STANAG 4607 – Ground Moving Target Indicator Format, GMTIF;
- STANAG 4609 – Motion Imagery;
- STANAG 4676 – NATO ISR Tracking Standard, DRAFT;
- STANAG 5516 – Link 16 Data Exchange, Track management.

Furthermore:

- All group members agreed on using the Robot Operating System (ROS) for operating the robots; and
- All group members agreed on using Battle Management Language (BML) as a minimum interface.

2.2 ROS

The Robot Operating System (ROS) is a flexible framework for writing robot software. It is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behaviour across a wide variety of robotic platforms.

The robot systems' middleware/software should be built upon ROS or at least have ROS interfaces.

2.3 BML

For single robots up to heterogeneous multi-robot systems consisting of air, land and sea-systems (individually equipped with many sensors and actuators) there are potentially many military tasks. The standardized BML was specially developed for commanding military units and can be used to command single as well as a multitude of robots provided each robot is programmed how to interpret and execute commands within his capabilities, and depending on whether it is acting alone or in a group.

Two examples of BML commands (which could be either given to a single robot or a group of robots) are given below:

OB advance Tasker Taskee Route-Where Start-When (End-When) Mod Why Label.

OB reconnaissance Tasker Taskee At-Where Start-When End-When Mod Why Label (Figure 2-1).

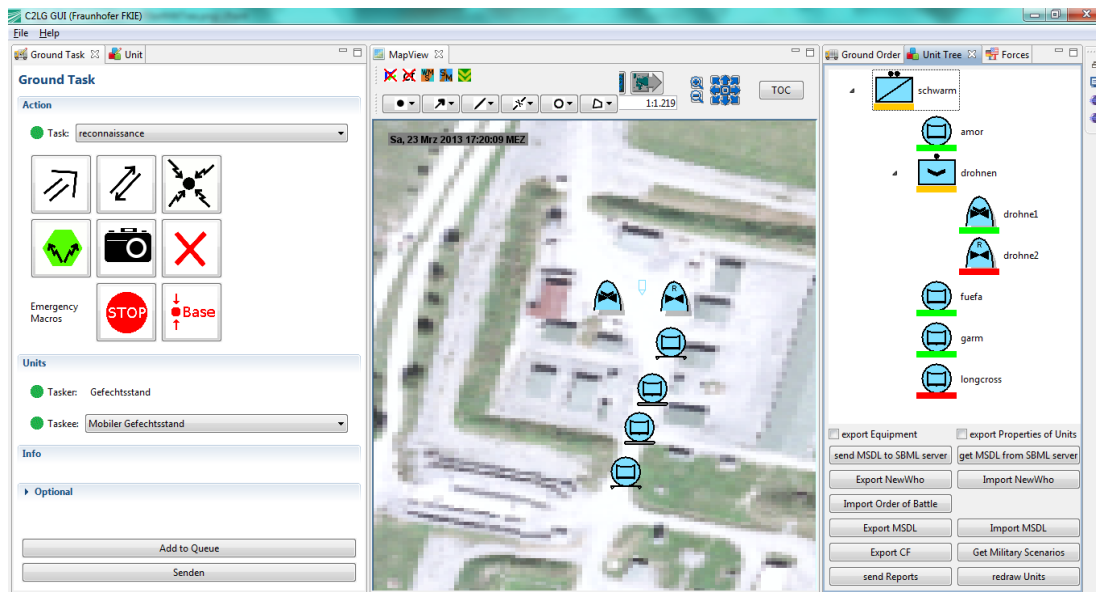


Figure 2-1: The BML-GUI Which is Developed by the Fraunhofer Institute FKIE Allows the Commanding of a Multi-Robot System by a Single Operator, and the Display of the Data Collected by the (Sensors of) the Robots.

2.4 PLATFORM / PAYLOAD

Although (not yet) part of an officially proposed standard we are also looking into defining some hardware standards for UGVs, especially for UGV payload, e.g., connection only over IPv4 based Ethernet using RJ45 cable and 24 V power supply, and standard payload size(s). With such standardized payloads different UGV platforms can be modularly equipped with the same exchangeable payload (mission-specific sensors and actuators are e.g., CBRNE reconnaissance, water jet for firefighting, manipulator for EOD, etc.) (Figure 2-2).



Figure 2-2: Robot Platform, 'Garm' with a Standardized Payload (In This Case a Simple Transport Box).

2.5 TEST TRACKS

Also (not yet) part of an officially proposed standard we have also started into looking at definitions of standard test tracks for mobility (TDSuS at Fraunhofer FKIE) and autonomous navigation (Swiss army training ground in Thun). In the case of autonomous navigation a single test (e.g., a run at a robotics competition) will not be enough because the performance may vary much with the weather conditions, and because – especially in an unstructured environment – the systems are generally not yet very reliable, many runs are needed for test results with good statistics.

2.6 M-ELROB

The military ELROB (M-ELROB), which takes place every two years (2014 in Warsaw was the fifth time), enables people to get a glance at the latest R&D in the area of unmanned outdoor/off-road ground systems. The different scenarios have been developed in closest co-operation with military users and reflect the up-to-date requirements of the forces.

One goal of the NATO group IST-107 RTG-052 on Standards Promoting Interoperability for Coalition UGVs (the follow-on activity of IST-089) is to promote, establish and validate the standards listed above within M-ELROB; the standards are introduced gradually (demanding an increasing number of different standards, e.g., data file formats, robot interfaces, battle management language, etc.).

So far five levels of interoperability have been defined (Table 2-1).

Table 2-1: Levels of Interoperability.

Level 1	Level 2	Level 3	Level 4	Level 5
ROS	BML	Standards	MAJIC (CSD)	Internet access

In 2014: All electronic data transfer/exchange in ELROB activities are done via WebDAV. The character en-coding is UTF-8 (8-bit UCS/Unicode Transformation Format), the geographic coordinate system is Universal Transverse Mercator (UTM) coordinate system, the geodetic reference system World Geodetic System (WGS) 84, time is UNIX Time/POSIX Time, graphics Portable Network Graphics (PNG) and/or JPEG (ITU-T T.81, ISO/IEC IS 10918-1 and, if needed, ITU-T T.84), and the sensor data collected during the competition is shared among all participants in Common Shared Data Sets (CSDS) in the ROS bag format.

Chapter 3 – UGV INTEROPERABILITY PROFILE (IOP) AND TESTS OF OCU INTEROPERABILITY AT TRAINING GROUND IN RENA, NOR

A. Tiderko, and Dr F.E. Schneider
Fraunhofer Institute for Communication
GERMANY

ABSTRACT

The Control Interoperability Profile (IOP) document, in conjunction with the Unmanned Ground Vehicle (UGV) IOP Overarching Profile, defines guidelines for the specification of message-passing interfaces and protocol behavior between UGV Platform payload(s) and Operator Control Unit (OCU) input/output devices, application environment and Human Control Interface (HCI).

3.1 IOP OCU STANDARDS

The OCU is the interface through which the UGV is controlled. OCU interfaces enable hardware and software interoperability providing operator(s) with command, control and status reporting capabilities as specified in RSJPO Capabilities Plan(s) and supporting Control IOP documentation. Relevant and/or applicable physical, electrical, and logical interoperability attributes are addressed:

- 1) Capabilities that are physically attached to one another with interfaces such as OCU mounting brackets, internal and external storage devices, display mounting systems, controls, connectors;
- 2) Control device interfaces such as joystick(s), touch devices, knobs, buttons, switches, etc. with interfaces such as USB, serial, fiber optic, Firewire;
- 3) Fixed displays and wearable devices with interfaces such as HDMI, DVI, VGA;
- 4) Audio input and output interfaces; and
- 5) Storage devices with interfaces such as USB, Serial ATA; 6. miscellaneous data interfaces such as RS232, RS485, RJ45 [24].

3.2 ROS-JAUS BRIDGE

To make the existing software IOP compliant, the ROS nodes had to be extended to become full-featured JAUS components. Translation between ROS and JAUS messages (ROS-JAUS-bridge) was done individually inside the particular ROS nodes. One example is the teleop node. Whenever the teleop node receives standard ROS messages it initiates the JAUS message sequence to gain control over the robot and send drive commands. Concepts which are not covered by ROS were implemented in dedicated new nodes [25].

3.3 UGV HANDOFF BETWEEN OCUS

IOP now supports the UGV handoff between OCUs, so UGVs and OCUs don't need to be rebooted to connect with a different pairing. To ensure proper control of UGVs access control is as defined in the UGV Interoperability Profile (IOP); OCUs will have to request and be granted control prior to sending any commands to the UGV platform. The UGVs are limited to granting control to only one OCU at a time and are not responding to any commands from OCUs not granted control. The UGVs go into a safe operating

mode when transitioning into a non-controlled state. Access Control is granted and denied based on the highest Authority Code provided by all the requesting controllers. To ensure that no OCU forcibly takes control of the UGV platform from another OCU, these OCUs would use the same Authority Code of when requesting control, together with an access control timeout. When an OCU drops or loses control, the UGVs broadcast availability status so that other OCUs are aware of its availability [25].

3.4 CHE-DEU COMMON PLATFORM

Germany and Switzerland were using the UGV platform ‘Garm’ (see Figure 3-1) and its OCU. In Garm all the interoperability standards which have been proposed in the RTG-149 (and its predecessors) are established in this platform; it has also been running in different scenarios at all the M-Elrob competitions since 2012.



Figure 3-1: Robot Platform, ‘Garm’, with a Reconnaissance Payload.

3.4.1 Interoperability Tests in Rena, Nor

In June 2018 the group performed a 5-day interoperability test in Norway, in and around a small village on the Rena military training ground. There were four different UGV platforms (see Figure 3-2) and five different OCUs. All the OCUs could successfully control all the UGVs, and the handoff of the control of a particular UGV from one OCU to another went smoothly and with no ambiguities. DEU and CHE were using the UGV platform ‘Garm’ and its OCU together. For the report refer to the DEU part of this document.



Figure 3-2: Parade of Robotic Platforms Participating at Interoperability Test in Rena.

3.5 OUTLOOK

We would like to see further standardization in the following topics:

- 1) *Enhancements to the IOP Obstacle Reporting Solution.*

When sensors determine an object, i.e., human, animal, vehicle, etc., this information should be added into the location data already defined in the current solution [25].

- 2) *Additional Standards for Payloads and Dynamic Payload Integration.*

Additional payloads should be tested to identify any issues with the IOP documentation for payload control and integration on platforms, e.g., manipulators, CBRNE sensors, effectors.



Chapter 4 – POLISH CONCEPTUAL OCU

J. Głowska, and R. Kozik

Industrial Research Institute for Automation and Measurements PIAP
POLAND

ABSTRACT

This chapter describes the Polish group's contribution to activities included in: NATO IST-149 / RTG-071 "Capability Concept Demonstrator for Interoperability within Unmanned Ground Systems and C2". Polish research and industry representation in the group was possible due to involvement of Industrial Research Institute for automation and Measurements PIAP (based in Warsaw).

The PIAP Institute, a mobile robot developer and manufacturer, joined NATO IST-149 in order to verify and validate the interoperability concept for the mobile robot control system.

4.1 SCOPE OF COOPERATION

Allocated resources allowed for the participation of PIAP researchers in several meetings conducted within the scope of NATO IST-149 / RTG-071:

- Spring Meeting 26 – 27 April 2017, Lilleström, Norway.
- Combined Meeting of NATO ToE on UGV and NATO IST-149 RTG, Fall 2017 – "Hackathon", Sep 11 – Sep 15, 2017, Koblenz, Germany.
- Combined Meeting of NATO ToE on UGV and NATO IST-149 RTG, Spring 2018 – "Plugfest" – May 14 – May 18, 2018, Koblenz, Germany.
- NATO Team of Experts UGV and NATO IST-149 RTG experiment, 11 – 15 June 2018, Rena, Norway.

The Polish party cooperated with the international team by bringing a demonstrator of a new mobile robot OCU concept (hardware and software) and participating in software integration activities with skilled engineers throughout the activities.

The goal was to achieve and present the interoperability of the mobile robot's control system by controlling different units from one control panel. In addition, a set of integration and engineering activities required to achieve interoperability was studied, analyzed, validated and verified during the process.

4.2 POLISH CONCEPTUAL OCU

4.2.1 Description

The Polish conceptual OCU consisted of a docking station with joysticks and buttons and a tablet computer inserted into it. It is supposed to operate as a mobile (remote) control station for UGVs of small and medium size (10 – 1000 kg). The preliminary design is presented in Figure 4-1 and Figure 4-2.

4.2.2 Hardware Setup

The tablet and docking station are powered from the internal battery placed inside the tablet with the possibility of recharging it via an external power source. The OCU contains connectors enabling connection to the tablet of external peripherals such as wired communication (RJ45), wireless communication system,

additional battery, gamepad and an additional screen. The operator can use a tablet touch screen and two joysticks located on both sides of the docking station.

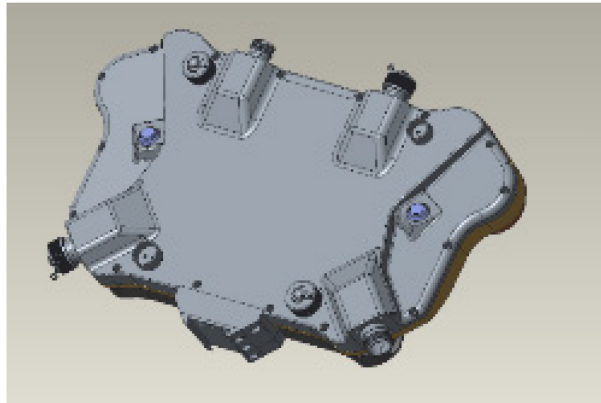


Figure 4-1: The Polish Conceptual OCU Consists of a Docking Station with Joysticks and Buttons.



Figure 4-2: A Tablet Computer is Inserted into the Docking Station.

4.2.3 Software Setup

The software side of the Polish conceptual OCU was realized to be interoperable within IOP v2 framework. The main available resources included:

- Platform state and control.
- Mobile base movement.
- Waypoint driving.
- Visualisation of geo-positioning on built-in mapping service.
- Simple manipulator operation.
- Video configuration and displaying.

The Polish conceptual OCU implementation has been based on OpenJAUS SDK (<http://openjaus.com/>) that has simplified the design and testing phases and allowed for the immediate application of any corrections or modifications in the way the OCU cooperated with platforms. No custom ROS-IOP bridges were used on the OCU side.

4.3 CONDUCTED TESTS

4.3.1 Tests with Robots

The Polish conceptual OCU was successfully tested with several platforms:

- Garm robot (Fraunhofer, Ruag) (Figure 4-3).
- Waypoint driving.
- Teleoperation (Figure 4-4, Figure 4-5, Figure 4-6).

4.4 GENERAL FINDINGS

One of the most important achievements during the testing was the ability to swiftly switch between the platforms with different configurations and capabilities. The modular software design of the Polish OCU allowed for easy reconfiguration of the Human-Machine Interface to better represent the chosen robot.

Despite using different a software development approach (based on OpenJAUS SDK for IOP) than the platforms (primarily using custom ROS-IOP bridges), the cooperation within IOP-specified functionalities was achieved without any significant technical challenges.



Figure 4-3: GARM Robot, Teleoperation.



Figure 4-4: Garm Robot, Teleoperation with OCU.



Figure 4-5: GVR-Bot (TARDEC), Teleoperation.



Figure 4-6: Telerob (WTD 41), Teleoperation.



Figure 4-7: Olav (FFI), Teleoperation.

4.5 VIDEO INTEROPERABILITY

The IOP uses discovery services that allow it to find and subscribe to existing video stream servers – mainly RTSP services offering h264 compression.

However practical implementation of that principle posed several challenges that were not easily solvable due to:

- Limitations and simplicity of the encoding sides:
 - Most of the robots used during the tests used simple closed solutions like Axis Q7411 or similar (<https://www.axis.com/nl-nl/products/axis-q7411>).
 - Used encoders did a good job in providing a proper stream but lacked the RTSP server capabilities that are expected in current evolution of streaming services.
- Video playback solutions built into OCU need to properly set up the video streaming services in order to set up a connection with as short a delay as possible and without any need for a buffering process which could compromise the overall video streaming performance (see Figure 4-8 from <https://www.chromium.org/audio-video>).
- The vendor of the robot platform and OCU is mainly concentrated on developing a solution that works on both ends and meet the requirements but is not looking into the interoperability of the video service.
- Current evolution of video playing solutions used in Polish conceptual OCU is based on the popular video decoders used by the Chromium web-kit engine (<https://www.chromium.org/audio-video>).

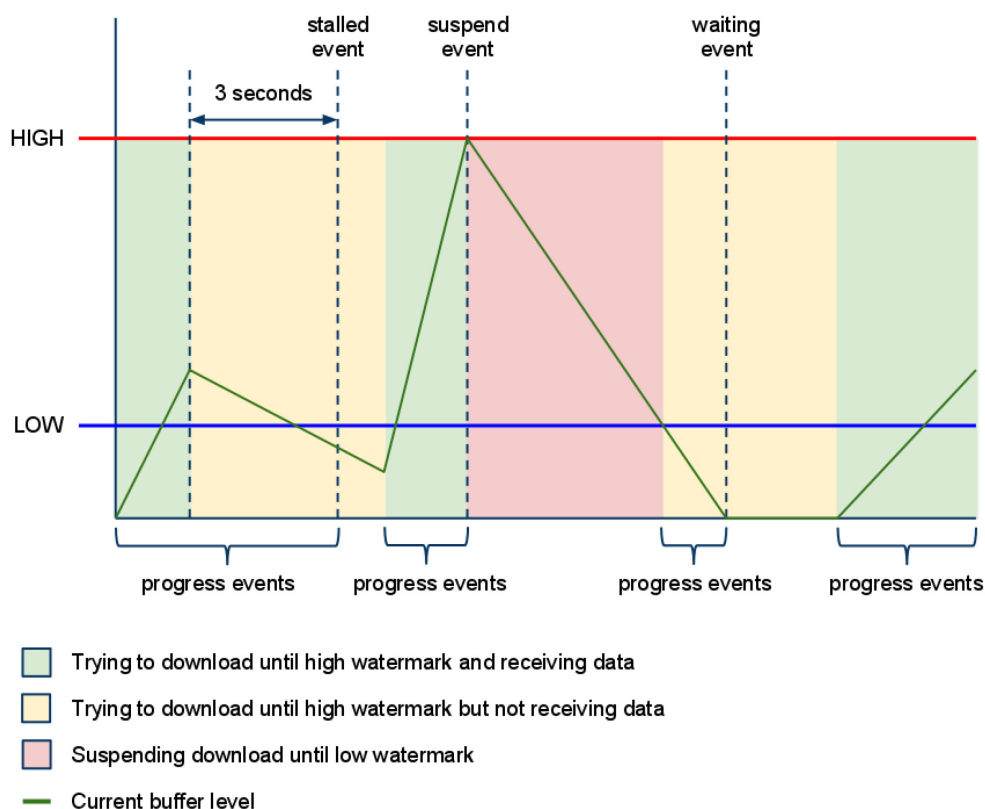


Figure 4-8: Chromium Uses a Combination of Range Requests and an In-Memory Sliding Window to Buffer Media (<https://www.chromium.org/audio-video>).

In order to support as many video receiving devices as possible the video streaming servers should use the most simple (and thus most quick) encoding option: H.264 baseline profile (<http://blog.mediacoderhq.com/h264-profiles-and-levels/>):

Primarily for low-cost applications that require additional data loss robustness, this profile is used in some videoconferencing and mobile applications. This profile includes all features that are supported in the Constrained Baseline Profile, plus three additional features that can be used for loss robustness (or for other purposes such as low delay multi-point video stream compositing). The importance of this profile has faded somewhat since the definition of the Constrained Baseline Profile in 2009. All Constrained Baseline Profile bitstreams are also considered to be Baseline Profile bitstreams, as these two profiles share the same profile identifier code value.

Baseline Profile (BP, 66) – https://en.wikipedia.org/wiki/H.264/MPEG-4_AVC.

4.6 SUPPORTED IOP FUNCTIONALITIES

4.6.1 Messages

IOP:

urn:jaus:jss:mobility:GlobalPoseSensor
urn:jaus:jss:iop:PathReporter
urn:jaus:jss:environmentSensing:StillImage
urn:jaus:jss:environmentSensing:DigitalVideo
urn:jaus:jss:environmentSensing:RangeSensor
urn:jaus:jss:mobility:LocalPoseSensor
urn:jaus:jss:iop:Measurement

4.6.2 Commands

IOP:

urn:jaus:jss:mobility:GlobalWaypointDriver
urn:jaus:jss:mobility:GlobalWaypointListDriver
urn:jaus:jss:mobility:PrimitiveDriver

4.7 CONCLUSIONS AND EXPERIENCE GAINED

4.7.1 Technical Findings

- Despite using different software development approach (based on OpenJAUS SDK for IOP) than the platforms (primarily using custom ROS-IOP bridges) the cooperation within IOP-specified functionalities was achieved without any significant technical challenges.
- The most important challenge that was discovered was the video stream compatibility between different platforms and OCUs.

4.7.2 Interoperability Findings

- IOP was proved to be working as a basis for controlling UGVs of different sizes with one OCU design approach.
- IOP was proved to be working with multiple UGVs and with switching between them.
- The OCU working on the IOP v2 basis was able to efficiently communicate with UGVs over different networking media (wired and wireless).

Chapter 5 – CONTRIBUTION OF THE UNIVERSITY OF OULU

J. Röning
University of Oulu
FINLAND

ABSTRACT

The responsibility of the University of Oulu was to provide an Unmanned Aerial System (UAS) for the experiment capable to work with the developed Operator Control Units (OCU) of other teams. The integration work was done in the Combined Meeting of NATO ToE on UGV and NATO IST-149 RTG Spring 2018 – “Plugfest”, organized in Koblenz, Germany, a month before the IOP Experiment.

5.1 DESCRIPTION OF THE SYSTEM

The provided low-cost quadcopter was built using a commercially available Flight Control Unit (FCU) with GPS localization, a custom video link solution utilizing a Raspberry Pi mini-computer, a 300 Mbps 2.4 GHz Wi-Fi dual antenna router, a gimbal for a Raspberry Pi camera and a 433 MHz telemetry radio. Hardware wise, the overall cost of the quadcopter was about 400 – 500 €.

The FCU and the rest of the system was integrated with ROS Kinetic running on a remote laptop computer, referred here also as Ground Control Station (GCS). The ROS-IOP bridge implementation used was the one provided by Fraunhofer FKIE [26], which was reconfigured so that it could be used with the UAS. In ROS, the MAVROS node was used for monitoring and controlling the UAS via FCU with Micro Air Vehicle Link (MAVLink) protocol.

Table 5-1: UAS Specifications.

	Name	Custom Quadcopter
Drone Specifications	Picture	Figure 5-1, Figure 5-2
	Type	Quadcopter, center body with four arms
	Configuration	4 rotor, X
	Temperature range	0 – 40 C
	Motors	920 KV, 255 W each
	Battery	4-cell Lithium-Polymer (LiPo)
	Propellers	Carbon fiber 10”
	Frame	S500, glass fiber and carbon fiber reinforced plastic
	Controller	PixFalcon, Micro PX4
	GPS	Yes
	Range Finder	LightWare LW20, up to 100 m
	Optical Flow	PX4Flow clone
	RC	Turnigy 10CH 2.4 GHz AFHDS 2A

	Name	Custom Quadcopter
	Telemetry	3DR 433 MHz / RFD 868x long range.
	Ground control	APM Mission Planner / ROS running on a laptop.
	Automatic flight	GPS Waypoints.
	Fail-safe system	Yes, return home.
Imaging	Camera	Raspberry Pi camera or any low delay USB 2.0 camera.
	Gimbal	TAROT gimbal.
	Video link	Raspberry Pi Zero or 3 model B, Asus USB-N14 2.4 GHz Wi-Fi in User Datagram Protocol (UDP) broadcast mode.
Dimensions	Length	0.4 m
	Width	0.4 m
	Height, excluding antennas	~20 cm
	Aircraft Weight (1 4S 8 Ah LiPo battery, gimbal, camera)	~2 kg
	Battery Weight	Depends on capacity and payload requirements.
	Max additional payload Weight	~0.3 kg
	Max Gross Take-off Weight	~2.3 kg
Speed	Max speed	25 m/s
	Cruise speed	5 m/s
	Maximum allowed altitude	120 m
	Endurance	Depends on battery capacity and propeller model, roughly 17 minutes.



Figure 5-1: Custom S500 Quadcopter.

5.2 CUSTOM COPTER'S SYSTEM OVERVIEW

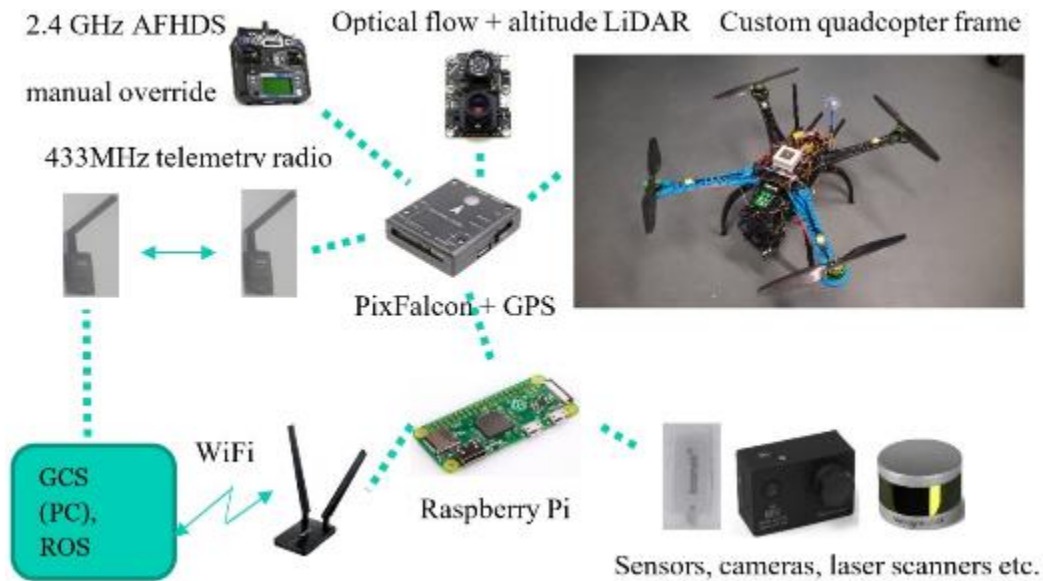


Figure 5-2: Custom Copter System Overview.

5.3 “PLUGFEST” OUTCOME

Marko Kauppinen and Rajesh Raveendran were responsible of integration and testing the ROS-IOP bridge for the UAS. Initial integration process was quite difficult as we had not participated in the earlier workshops organized in NATO IST-149 RTG. However, with the help of other teams, we managed to get the ROS-IOP bridge working in our system so that it could have been used to control the UAS. In the end, the ROS-IOP bridge integration is not too difficult, although to get it working properly, we would have needed to have access to an OCU much earlier to develop the system so that it would have been fully compliant with all the requirements.

The biggest issues encountered during integration and development during the “Plugfest” were related to the fact that the available ROS-IOP bridge compliant OCU’s were primarily intended for mobile ground platforms. Therefore, the control options offered by the IOP bridge were not directly compatible with our system as UAS cannot be operated in the same fashion as an UGV system.

Therefore, the ROS-IOP bridge for a UAS needed some hacks for adding a feature of supplying predetermined flight paths to the copter. Letting the operator take control of the copter directly via OCU is currently not an option. This is because the OCU control interfaces and the implementation as a whole was currently intended primarily for UGVs, without consideration for UAS control. In addition, to operate a UAS in direct control mode from an OCU has very strict delay constraints with respect to the actuation control loop and for the video views. Therefore, there should be as much automation as possible on the UAS side to simplify the control process.

5.4 UAS CONTROL SOFTWARE ARCHITECTURE

The laptop used as GCS for the UAS had Intel i5-2520M CPU @ 2.50 GHz with an integrated graphics card and 8 GB of RAM. The UAS was controlled with the ROS Kinetic distribution, version 1.12.13. The FCU used APM Copter firmware V3.4.2. In this experiment, the UAS was generally tested in GUIDED mode and

in this mode, the copter can be guided dynamically to a target locations by sending waypoints through Mission Planner [27] software or from ROS through the MAVROS node [28] via a telemetry radio.

The UAS uses Micro Air Vehicle Link (MAVLink) protocol for communication, which comprises both mission protocol and parameter protocol. It is a binary telemetry protocol designed for resource and bandwidth constrained systems. From the ROS side, MAVROS package provides the communication interface for UAS and in addition to that, it provides UDP MAVLink bridge for 3rd party GCS software such as Mission Planner or QGroundControl. The complete documentation about the ROS `mavros_node`, topics, services and parameters are available in Ref. [28].

The GCS laptop also contains Interoperability Profile (ROS / IOP) Bridge package, which bridges JAUS Toolset (JTS) and ROS. This package gives the capability to connect and communicate with different OCU's. The overview of how the FKIE ROS / IOP Bridge package works is described in Ref. [29].

In order to connect to OCUs, the UAS has to be discovered by the OCU and vice versa. The `iop_component_fkie` package entails `iop_component` node which is used to create two components, namely `iop_platform_manager` and `iop_digital_video`, and each component is configured with different sets of services. Both components include the `iop_discovery_fkie` service, which is responsible for connecting the UAS and the OCU with `iop_address` – 112.1.15, where 112 is the `id_subsystem` and it is JAUS system specific. The `iop_digital_video` component comprises of the services that are used to provide the RTSP link to the OCU that can take control and access the UAS. Figure 5-3 depicts the list of services that are used by the OCUs.

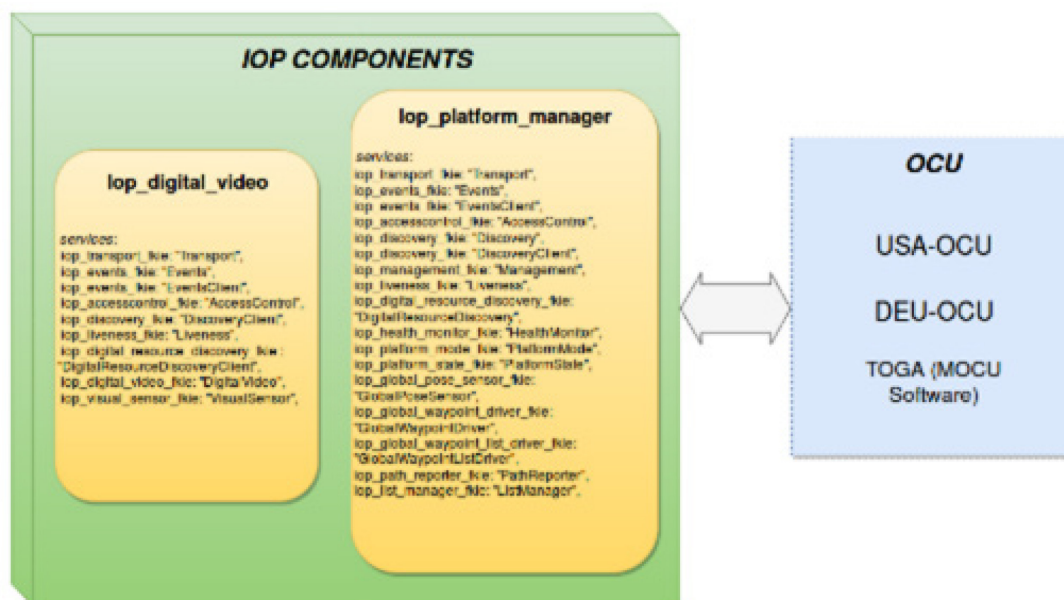


Figure 5-3: List of IOP Components and Services Utilized.

Figure 5-4 depicts the control architecture of the UAS. In this architecture, the PixFalcon – FCU executes the control command based on the flight parameter settings and the waypoints are sent from the IOP/JAUS bridge, which also publishes and subscribes the related messages from and to MAVROS connected to the FCU on the UAS via a telemetry radio. Since the IOP/JAUS system has no compatibility for MAVLink protocol as it is developed for UGV systems, the waypoints sent from OCU are in a path message, `nav_msgs/Path`. The FCU can execute waypoints according to MAVLink protocol only, which need to be provided in `mavros/Waypoints`. This message has not only GPS data but also contains the parameters from

MAV_CMD. A python code is implemented that parses the waypoints from OCU to MAVLink compatible waypoints. However, this is not a straightforward method as the MAVLink protocol control messages contain also vital flight parameters.

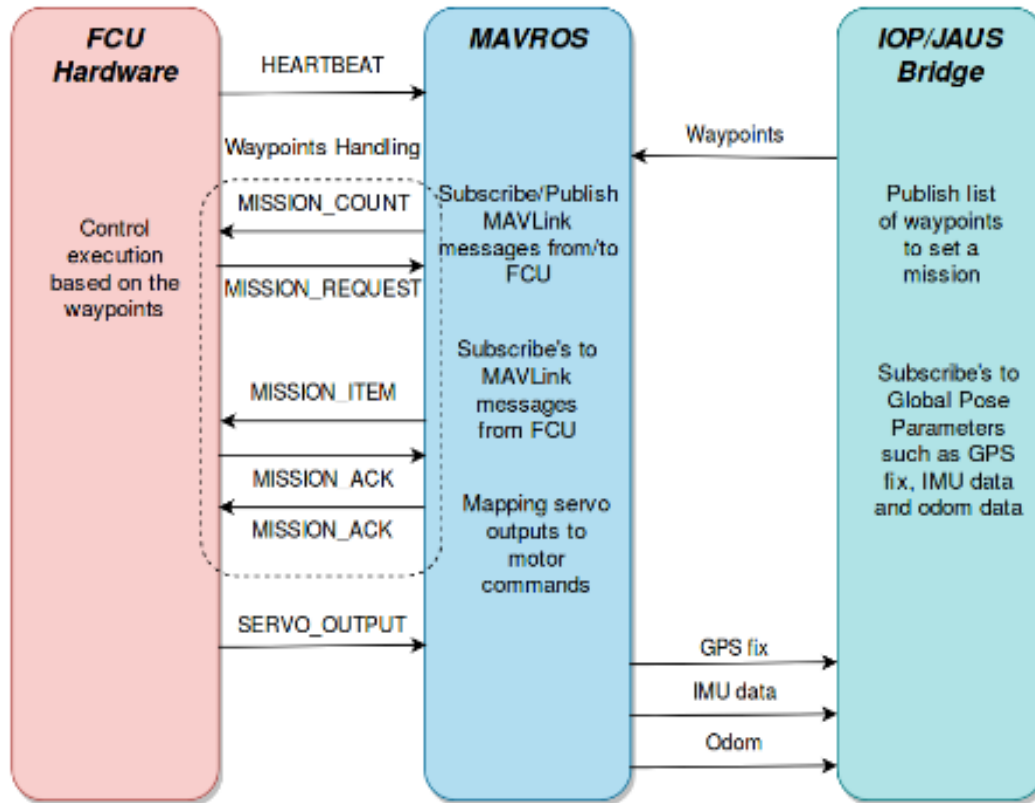


Figure 5-4: UAS Control Architecture.

5.5 VIDEO Streaming from UAS Over WI-FI to GCS

For live streaming, The UAS video is transmitted using the EZ-Wifibroadcast installation [30] on two Raspberry Pi's, one for sending on the UAS and one for receiving video on the GCS laptop. The video stream sent over the video link is H264 encoded with Forward Error Correction (FEC) and sent over the Wi-Fi link using UDP protocol. The stream is read in ROS from the UAS using the ROS gscam node [31], which entails GStreamer [32], a multimedia framework that is used to produce raw image topic for ROS of the camera in UAS via the Wi-Fi link shown in Figure 5-2.

With the GStreamer framework, it is possible to build up multimedia “pipelines” consisting of source, sink and filters. In gscam node, the parameter “GSCAM_CONFIG” sets the environment variable required for GStreamer. In this system, “udpsrc port=5600 ! H264parse ! avdec_h264 ! Videoconvert” environment variable is set as the pipeline definition for the GStreamer. The raw image is then published in ROS under the topic */raspicam/raw_image*.

In order to provide live streaming to OCU, *image_to_rtsp* ROS node is used. This node converts the image topic, */raspicam/raw_image* to RTSP and mounts it in */rgb* camera topic. The *iop_digital_video* component provides the service that contains the RTSP URL – “*rtsp://192.168.0.103:8554/rgb*” that is created by the *image_to_rtsp* node, for the OCU that can take control and access the UAS.

Finally, Figure 5-5 shows the ROS node graph of the overall system.

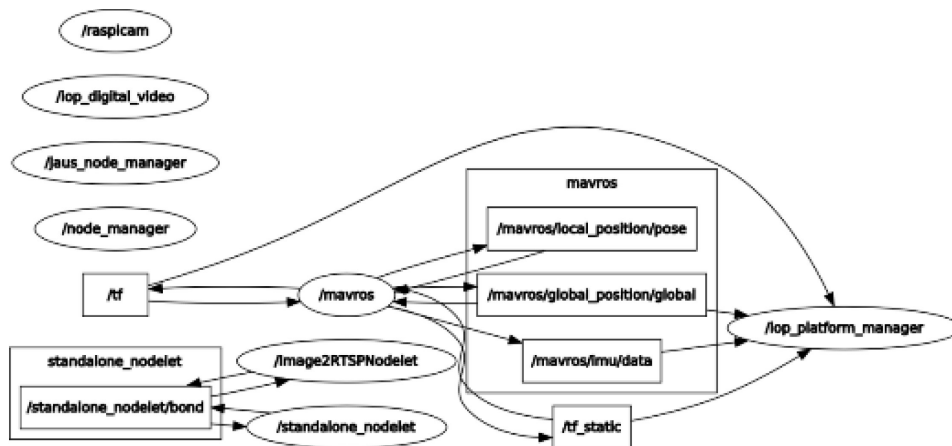


Figure 5-5: ROS Node Graph of the Remote GCS Laptop.

5.6 OFFLINE UAS TESTING WITH SITL

Software In The Loop (SITL) [33] is used for the purpose of routing the MAVLink protocol through simulation to test the controller and mission planning offline (Figure 5-6). The simulator is an open-source project from Ardupilot, as is the APM firmware used in the actual UAS FCU. Figure 5-7 depicts the architecture of the SITL from Ref. [34].

The UAS control algorithms can be tested using SITL in any map region by using *Microsoft satellite map*. With the help of the simulator, it is possible to run copter, rover and flight simulation of platforms that use Ardupilot firmware. Both the SITL simulated copter and the UAS have the same flight control modes and parameters available. Therefore, it is easy to test missions before any actual flight with the UAS. Figure 5-6 shows the SITL environment running in the GCS.

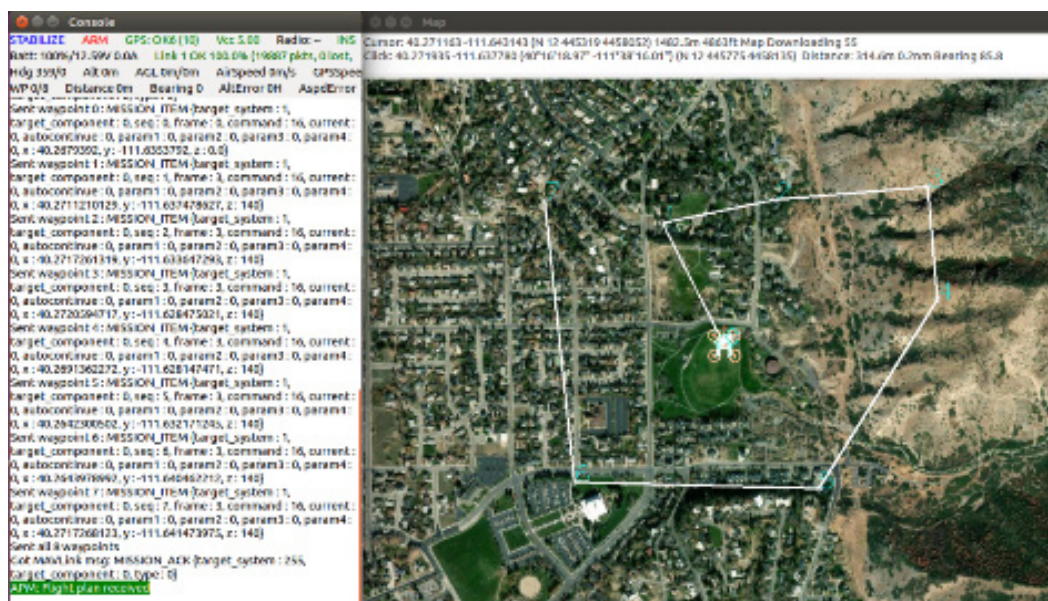


Figure 5-6: Mission Planning / Simulation with SITL.

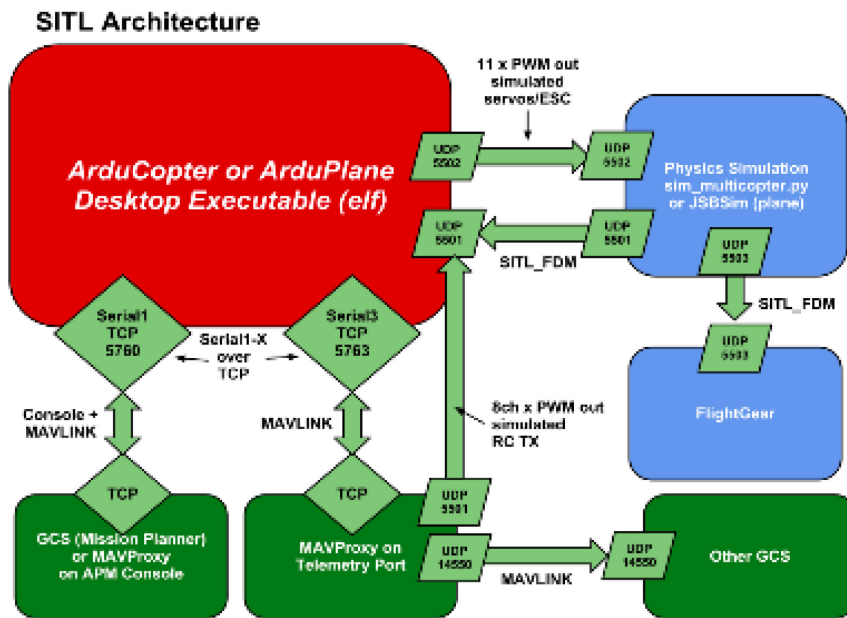


Figure 5-7: Architecture of SITL.

5.7 UAS CRASH BEFORE THE IOP EXPERIMENT IN RENA NORWAY

Unfortunately, just before the IOP Experiment, we experienced a major crash, Figure 5-8, during testing due to a failed motor bearing, causing the copter to shut down abruptly in more than 10 meters altitude. The analysis of the logs indicated that the FCU probably interpreted the failed bearing as a sign of crashing into something and triggered a fail-safe, shutting down the drone in mid-flight. This led to UAS frame being broken and the destruction of most of the motors, which we could not replace in time for the IOP Experiment in Rena, Norway.



Figure 5-8: Quadcopter After Crashing.

Since the accident, we have built two new drones with some redundancy against catastrophic motor failure, a hexa- and an octocopter, Figure 5-9. In these drones, if a motor fails it should not result to the copter immediately crashing to ground. These drones are also more stable during flight and have more lifting capability. They still use Pixhawk / PX4 based FCUs, so the ROS-IOP bridge implementation used with the quadcopter UAS should be compatible with these platforms.



Figure 5-9: Hexa- and Octocopter.

Chapter 6 – US OVERALL SUPPORT

P. Bunker
US Army CCDC-GVSC
UNITED STATES

ABSTRACT

US Army Tank-Automotive Research Development Engineering Center (TARDEC) supported RTG-149 by providing a Ground Vehicle Robotics Robot (GVR-Bot), a controller and the Robotics and Autonomous Systems – Ground (RAS-G) Interoperability Profile (IOP). The IOP is being used to promote interoperability of present and future Unmanned Ground Vehicle (UGV) systems larger than 5 Kg in a NATO Combined / Joint Service Environment. Interoperability is required because it will significantly enhance the war fighting capability of the forces. Interoperability will increase flexibility and efficiency to meet mission objectives through the sharing of assets and common utilization of information generated from UGV systems. NATO adoption of UGV IOP as a standard will allow for controllers to control other country's UGVs, providing additional functionality to NATO combatant commanders and reducing training time. In addition, countries can take advantage of UGV IOP compliant radios and payloads, potentially reducing costs and increasing functionality of an individual country's UGV systems.

NATO nations will be able to have a baseline document to add functionality based on emerging requirements. US UGV IOP is being updated with new functionality approximately every two years and will be offered to NATO countries along with their development for inclusion in updated standard documentation. Countries can utilize the standard for procurement and have interoperability with other countries compliant systems and subsystems.

6.1 IOP

This initiative is to identify and define interoperability standards to be organized and maintained within a Robotics and Autonomous Systems – Ground (RAS-G) Interoperability Profile (IOP). This IOP will be employed by Unmanned Ground Vehicle (UGV) acquisition managers in the acquisition of future Programs of Record, the upgrade of fielded systems, and the evaluation/acquisition of Commercial-Off-The-Shelf (COTS) products. Throughout this document, the terms RAS-G and UGV are used interchangeably.

A primary goal of this initiative is to leverage existing and emerging standards within the Unmanned Vehicle (UxV) community such as the Society of Automotive Engineers (SAE) AS-4 Joint Architecture for Unmanned Systems (JAUS) standard and the Army Unmanned Aircraft Systems (UAS) Project Office IOPs. The Advanced Explosive Ordnance Disposal Robotic System (AEODRS) Architecture Description Documents (which uses SAE AS-4) has been leveraged significantly. The end goals of this effort are:

- Facilitating interoperability among new UGV initiatives and legacy systems;
- Facilitating interoperability between controllers and UxV robotic system(s);
- Facilitating collaboration between UGV and UAS systems;
- Providing a path forward to standardized interoperable technology solutions; and
- Promoting payload and on-board subsystem modularity and commonality across the portfolio of UGV systems.

6.2 INTEROPERABILITY ATTRIBUTES

The RAS-G IOP has been designed to support interoperability on a variety of missions and objectives, vehicle classes/types, controller classes/types, payload classes/types, physical/software architectures, and interaction with external systems (e.g., networks, C2). Since every interoperability requirement will not be applicable to every system, the IOP provides a mechanism to independently specify these requirements in a composable, hierarchical manner, using Interoperability Attributes. In this way, Interoperability Attributes applicable to the specification and design of a system can be identified and subsequently utilized to filter applicable requirements from the RAS-G IOP, supporting system design, development, conformance and validation testing, initial operational test and evaluation, and fielding.

Project Management (PM) offices have the responsibility for identifying the Interoperability Attributes applicable to each acquisition program. The prime systems developer has the responsibility for implementing the RAS-G IOP in accordance with the specified Interoperability Attributes, and the conformance and validation tester has the responsibility for developing and executing conformance tests, based on those Interoperability Attributes.

For each individual acquisition program, the Project Manager (PM) representatives will review the IOP package and select which Interoperability Attribute values will apply, based on system requirements. The requirements associated with those Interoperability Attributes will then become part of the contractual requirements for that program.

6.3 INTEROPERABILITY ATTRIBUTE REFERENCE GROUPS

Many of the capabilities defined by these attributes are further refined or extended using related groups of more specialized attributes, referred to as attribute reference groups. These groups may be optional or required, and are either selectable independently, as a group, or in a mutually exclusively manner. Table 6-1 lists the different combinations and the related implications for each grouping type.

Table 6-1: Attribute Reference Group Selection Types.

Optional	Selection Type	Description
True	All	Either all attributes or no attributes from the attribute reference group MUST be selected. These attributes are linked together.
False	All	All attributes from the attribute reference group MUST be selected. These are mandatory attributes.
True	Any	As many of the attributes as desired (including none) from the attribute reference group may be selected. These are optional attributes.
False	Any	At least one of the attributes from the attribute reference group MUST be selected. These attributes are each optional, as long as one is selected.
True	One	No more than one attribute from the attribute reference group can be selected. These are mutually exclusive, though not required, attributes.
False	One	One and ONLY one attribute from the attribute reference group MUST be selected. These are mutually exclusive attributes where a choice is required.

6.4 IMPLEMENTATION OF STANDARDS

The RAS-G IOP defines standards as well as guidance with respect to implementation in order to promote interoperability as required by the acquisition program managers. Standards requirements are identified in a variety of areas, including electrical, mechanical, video, audio, communications, and messaging. However, due to the broad scope and expected operational usage of UGVs, not all standards will be applicable to every system.

Acquisition programs requiring conformance to this IOP will specify their interoperability requirements such that system developers can identify and conform to the applicable sections of the IOP.

System developers shall adhere to the standards and guidance as mandated within their procurement contracts relating to this IOP. Robotic system end items shall be tested and validated in accordance with the conformance and validation clauses contained within this overarching profile, as well as those contained in applicable companion IOP documents.

6.5 CONTROL AND STATUS MESSAGES AND JAUS PROFILING

To the degree possible, the RAS-G IOP utilizes the SAE AS-4 JAUS standards to define the interfaces between the OCU and the UGV as well as among on-board UGV subsystems. The *RAS-G IOP SAE AS-4 JAUS Profiling Rules* document provides guidance with respect to the profiling of the JAUS standard, in order to define the manner in which interfaces are applied/interpreted as a means of limiting ambiguity and maximizing interoperability among disparate vendors.

6.6 CUSTOM SERVICES, MESSAGES, AND TRANSPORTS

A set of “custom” RAS-G IOP services has been defined to provide mission capabilities scoped within the RAS-G IOP domain that are either not currently available in the SAE JAUS standards or require SAE JAUS extension by the SAE AS-4 Committee. These messages are documented in the *RAS-G IOP Custom Services, Messages, and Transports document*. To the extent possible, Custom Services, Messages, and Transports will be avoided, and when deemed necessary will be taken to the SAE JAUS committee for consideration of inclusion in a future public release. If/when Custom Services, Messages, and Transports have been accepted and published within the applicable SAE AS-4 JAUS standards, they will be removed from the *RAS-G IOP Custom Services, Messages, and Transports document*.

6.7 SYSTEM CONTEXT

From a system perspective, the RAS-G IOP is defined to address interoperability at multiple levels within varying systems configurations. The context diagram, presented in Figure 6-1, depicts this concept by showing the UGV and a controller interfacing to external UxV and external command and control (C2)/battle command systems. This concept can be instantiated in a number of ways from a basic OCU/UGV configuration to more complex configurations (i.e., an OCU controlling multiple UGVs with UAV feeds and/or a communication relay/extension). Interoperability can be applied to various aspects of these configurations as required by the system product manager, to include:

- OCU/UxV(s) – radio/data interfaces;
- Intra-OCU – between and among OCU hardware and software elements;
- Intra-UGV – between and among UGV subsystems/payloads and platform; and
- OCU/UGV to External C2 Systems – exchange of command and control, battlespace, and audio/video information.

For the purposes of IOP, only the circled portion of this architecture is within scope. This includes the hardware and software interfaces to define interoperability and modularity between a platform and an OCU, between a platform and its payloads, between a radio and a platform or OCU, and between an OCU and its human operator.

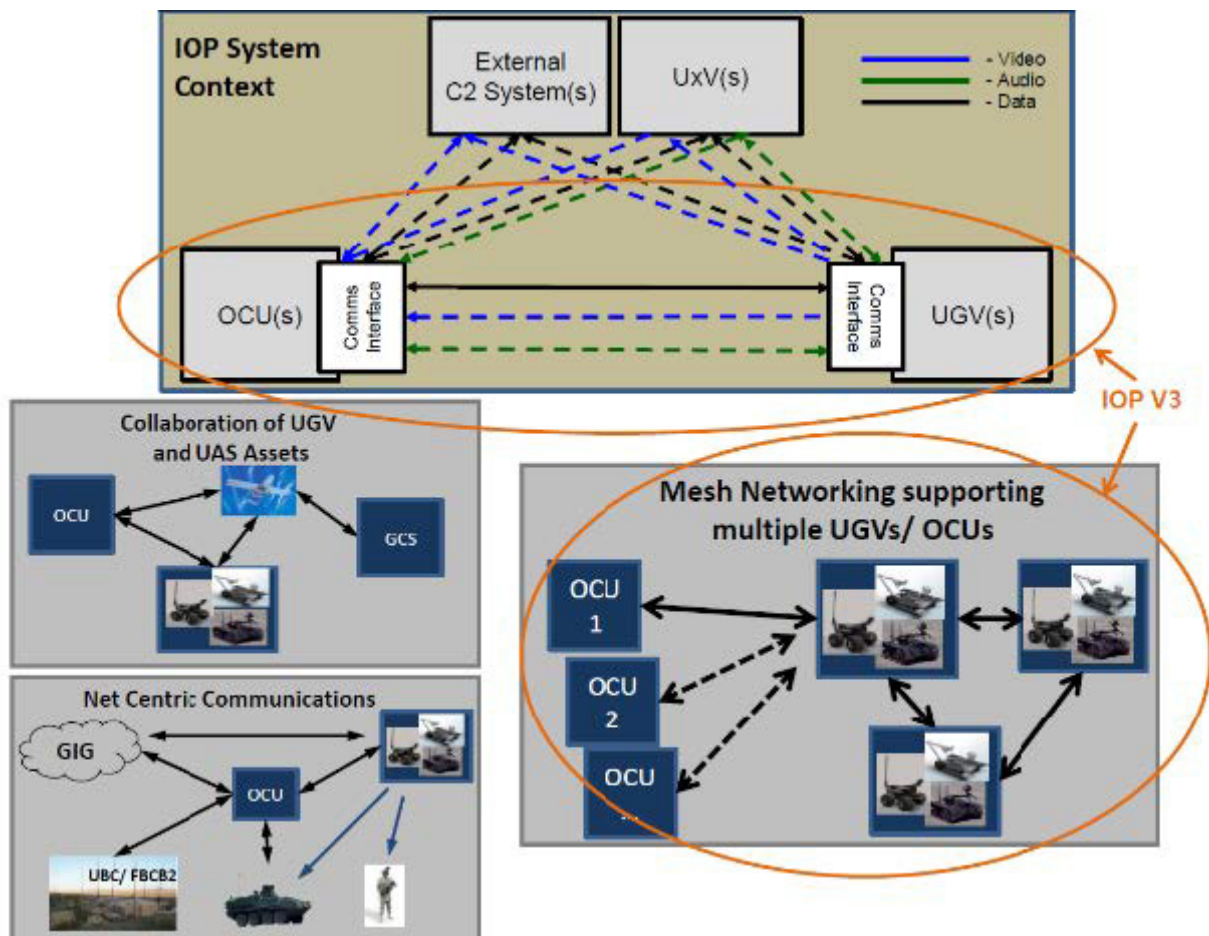


Figure 6-1: RAS-G IOP Context.

6.8 REFERENCE ARCHITECTURE

The architecture presented in Figure 6-2 builds upon the system context and defines key integrated elements of the UGV and OCU along with internal and external interfaces. These elements are defined in the IOP in a generic (abstract) manner and may be realized within a system in a variety of ways. For example, although all UGVs will have a conceptual platform controller to provide platform and mobility processing, the platform controller may be realized as a single computer, a single line replaceable unit (LRU) with multiple computers or a set of LRUs with distributed processing and internal interfaces. The terms defined and presented in the architecture diagram are defined to provide a consistent terminology and point of reference for utilization throughout the IOP.

As depicted in the figure, systems within the reference architecture are comprised of integrated elements, interfaces, and potential interfaces. An integrated element is an element that will exist (at least conceptually) within the specified system; an interface defines an inter or intra-system linkage between

integrated elements; and a potential interface is an interface that may or may not exist depending upon the required functionality as specified by a system's Interoperability Attributes.

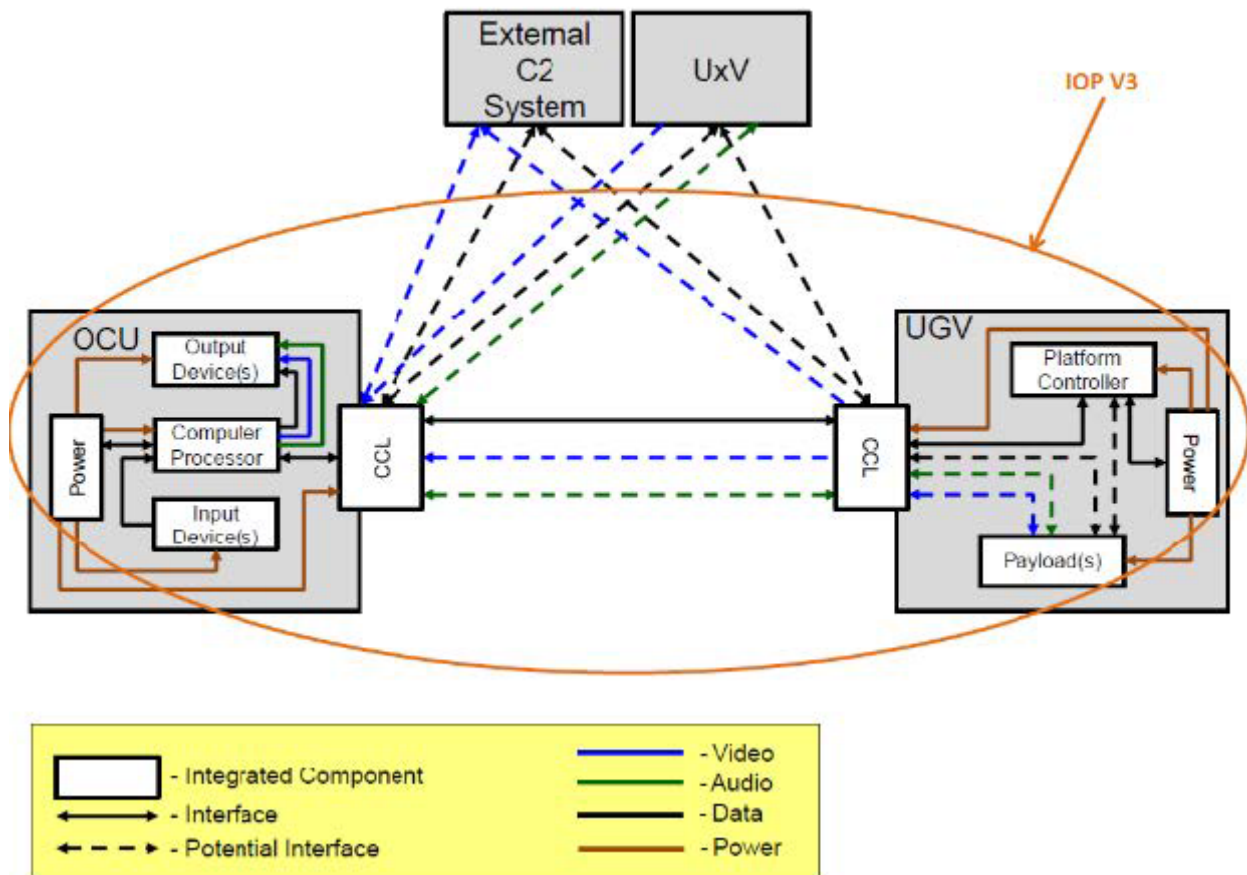


Figure 6-2: RAS-G IOP Reference Architecture.

6.9 OPERATOR CONTROL UNIT (OCU)

The OCU system of the reference architecture provides the human operator a capability to issue commands to and receive input back from one or more UGVs. Specific OCU-level requirements are included within the Control IOP.

The Input and Output Device element(s) are generic and could be adapted to describe a traditional controller implemented with joysticks, keyboard, mechanical switches, and a display, or other types of controllers such as those implemented with a glove, video game controller, monocle, voice, smartphone, or tactile belt.

6.10 COMMON COMMUNICATIONS LINK (CCL)

The Common Communications Link (CCL) element is an abstraction of the networking and/or point-to-point communications solution required for command and control of UGVs. It provides the OCU with the ability to transmit and receive data to/from a UGV, external C2 System, and/or another UxV.

6.11 UNMANNED GROUND VEHICLE (UGV)

The UGV system of the reference architecture consists of a Platform Controller element integrated with zero or more Payload elements and interfaced externally to an OCU and/or UxV and External C2 Systems via a CCL element. The Platform Controller element is conceptually responsible for providing platform management (e.g., system diagnostic monitoring, system safety, BIT/FIT) and mobility. The Platform Controller element receives data and provides status through the CCL element. The Payload element(s), if present, may be communicated with external consumers via an external connection through the CCL or may be integrated directly to the Platform Controller element. An example of a Payload element with a direct connection to the CCL might be an SAE AS-4 JAUS compliant payload that can be discovered and communicated to directly by an OCU. An example of a Payload element with a connection to the Platform Controller element might be a payload with an SAE AS-4 JAUS interface resident on the Platform Controller where commands to the SAE AS-4 JAUS interface are translated to the payload in a non-SAE AS-4 JAUS format. In addition, an SAE AS-4 JAUS compliant payload capable of interfacing directly to an OCU may still have an interface to the Platform Controller in order to provide subsystem status and to register for platform data (e.g., navigation, state/mode). The Power element provides power to all UGV integrated components.

6.12 EXTERNAL COMMAND AND CONTROL (C2)

The External C2 system of the reference architecture provides for the interfacing of an OCU and/or UGV with an external system, such as a battle command system. Sharing of payload data into external ground or air-based systems or networks will be addressed in a future versions of the IOP.

6.13 OTHER UNMANNED VEHICLES (UXVS)

The UxV system of the reference architecture provides for the coordinated interaction of an UxV system with the OCU and/or UGV. UxV in this context represents a generic unmanned system to include another UGV, a UAV, or an unattended sensor. While many UGV/UxV capabilities can be achieved by utilizing a combination of IOP requirements and requirements of existing UxV standards, explicit interface requirements for communicating with UxVs outside of the ground domain will be addressed in a future versions of the IOP.

6.14 UGV LOGICAL INTERNAL ARCHITECTURE

In addition to the reference and system architecture, the RAS-G IOP applies to the logical internal architectures for both the OCU and the UGV, specifying the general hardware and software interfaces relevant to interoperability within the OCU and UGV systems. The term logical refers to the fact that these elements may be implemented in varying physical configurations and that the physical configuration is not specifically relevant to the manner in which these elements interface with one another within the system. An example UGV logical internal architecture is presented in Figure 6-3.

The solid elements depicted in the figure represent elements that are always present within the UGV system. The dashed elements represent potential elements that may be present within the UGV system. Similarly, the solid lines between elements represent defined element interfaces while dashed lines represent potential element interfaces. Within the RAS-G IOP, potential element/interface requirements are subset via Interoperability Attributes to provide for the specification of a basic UGV configuration that can be augmented to accommodate specific interoperability requirements related to the interface with external system(s). A brief description of the controller elements is presented in the following paragraphs.

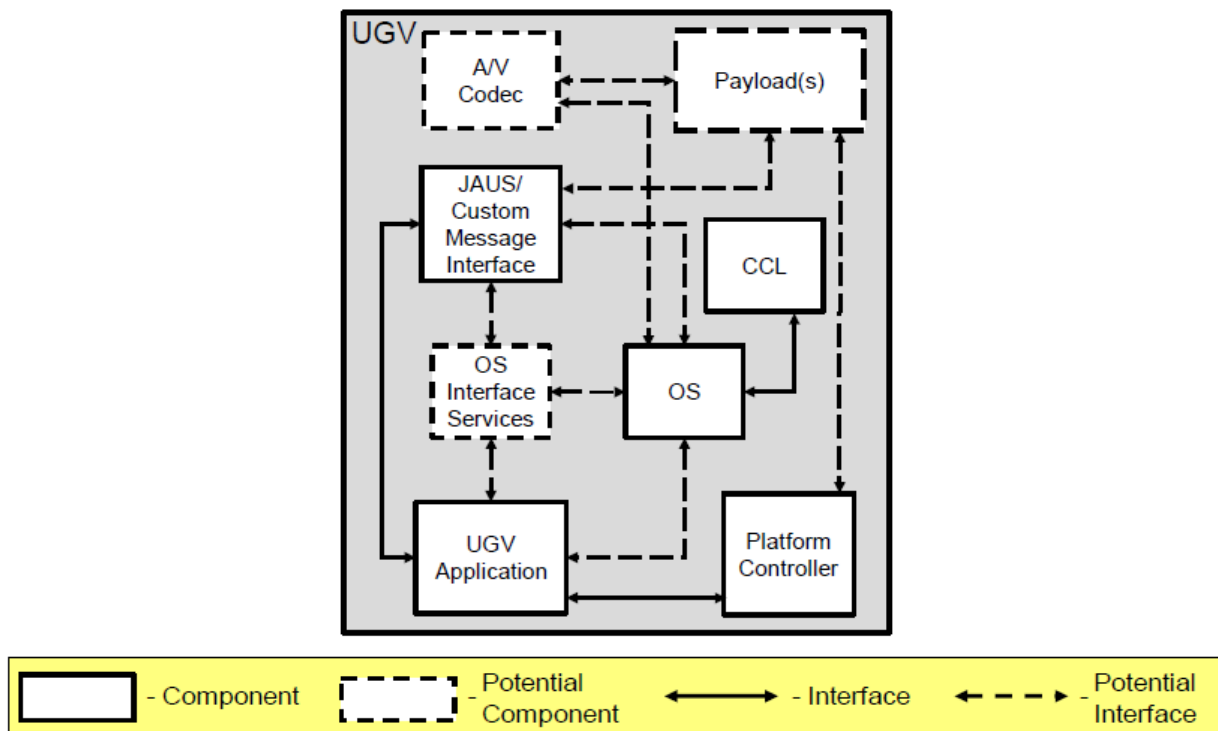


Figure 6-3: Example UGV Logical Internal Architecture.

OS (Element) – The OS represents the controller operating system. In general, requirements related to the operating system will not be addressed within the IOP and will instead be defined via acquisition requirements or design decisions.

OS Interface Services (Potential Element) – The OS interface services specify Inter-Process Communication (IPC) mechanisms required to implement interoperability interfaces between software (application) elements. This could be an interface between the controller application and an external system(s), the controller application and a vendor plug-in, and/or an interface utilized by the JAUS system to provide an underlying transport.

A/V Codec (Potential Element) – The A/V Codec represents the element within the system that decodes/encodes digital audio and video data streams. The codec formats required within the system will largely be dependent upon the requirements associated with specific payloads and/or the communications data link.

JAUS/Custom Message Interface (Element) – The JAUS/Custom message interface represents the element responsible for interfacing with the application to encode, marshal, transmit and receive, unmarshal, and decode JAUS messages between the controller and the robotic system(s). This element may be incorporated with or separate from the application software component.

UGV Application (Element) – The UGV application specifies the element responsible for the logical execution of the UGV system. In general requirements related to the application will not be addressed within the IOP and will instead be defined via acquisition requirements or design decisions.

Platform Controller (Element) – The platform controller specifies the element responsible for hosting and execution of the UGV application. This element may be realized in a variety of ways from a small micro-controller up to a multi-LRU configuration in accordance with the platform size and mission

capabilities/requirement(s). In general, internal requirements related to the platform controller will not be addressed within the IOP and will instead be defined via acquisition requirements or design decisions. External and interfacing requirements associated with the platform controller will be defined and presented within the IOP.

CCL (Element) – The CCL represents the communications data link between the controller and the robotic system. The requirements governing this interface are specified within the *RAS-G IOP Communications Profile*.

Payload(s) (Potential Element) – The Payload element specifies the element responsible for conducting mission specific functions/capabilities alone or in concert with other payloads and/or the UGV platform. This element may be realized in a variety of ways and configurations in accordance with the platform size and mission capabilities/requirement(s). Internal requirements related payloads will not be addressed within the IOP and will instead be defined via acquisition requirements or design decisions. Payload interfaces to/from the platform (e.g., electrical, mechanical), as well as interfaces between the payload and external systems (e.g., data formats, message protocols) are defined within the *RAS-G IOP Payload Profile*.

6.15 OVERVIEW OF IOP USAGE PROCESS

The IOP will be used by both the Project Manager (PM) community and the vendor community. For convenience, individuals within the Materiel Developer (the PM) will be referred to as “MATDEV”. Private industry (or academia) developers of systems, payloads, radios, technologies, or systems engineering/integration expertise will be referred to as “vendors”. For a given UGV acquisition program, the process begins with the MATDEV reviewing the operational requirements as articulated by the User community.

The MATDEV will then use systems engineering processes to transform these operational requirements into performance requirements. During this process, the MATDEV will conduct a formal review of the IOP documentation and select the attributes defined within the set of IOPs corresponding to the capabilities of the system. Many Interoperability Attributes will have one or more associated requirements, which will be inserted into the PM’s contractual requirements for the program’s Request for Proposal (RFP). Selection of the appropriate Interoperability Attributes will often require the MATDEV to conduct formal trade studies to inform the decision. There are also a number of Mandatory Interoperability Attributes across all of the IOP documents, whose requirements will be imposed on all systems. These Mandatory Interoperability Attributes are further described within this document, the *JAUS Profiling Rules document*, and the *Communications, Control, and Payloads IOP documents*.

The vendors within the competition space will then implement the requirements that have been selected based on the Interoperability Attributes, in accordance with the IOPs, and particularly in accordance with the *JAUS Profiling Rules IOP* and the *Custom Services, Messages and Transports document*.

An example of this process would be that the MATDEV receives a new requirement for a system that has Leader/Follower capabilities, as well as requirements for sensing Chemical, Biological, Radiological, Nuclear (CBRN) threats. In this case, the MATDEV will know that in addition to specifying the Mobility Interoperability Attribute with its associated “Core Mobility”, “Drive Timeout”, and “Emergency Stop Safety” requirements, they will also know that they must select, at a minimum, the Leader/Follower Interoperability Attribute. Similarly, the MATDEV will also know that it must also specify the CBRN Sensor Attribute, in addition to other common payload requirements. The requirements associated with each of these attributes would then become requirements imposed on the vendor to be compliant with the performance and/or product specifications.

6.16 IOP USAGE

For the MATDEV, the IOPs will be used as part of the requirements decomposition and allocation process prior to release of RFPs. It should be noted that the interface requirements in the IOPs represent a “product level specification”, as opposed to the typical “performance level specification” that PMs are typically responsible for. This has been done intentionally in order to provide sustained interoperability and modularity of systems throughout their full lifecycles. MATDEVs will build program-specific interoperability requirements into their RFPs prior to being released, based on the IOP. For the MATDEV, the Overarching IOP, Payloads IOP, Communications IOP, and Control IOP are of primary interest, since those contain definitions of the Interoperability Attributes, and the applicable requirements for each.

6.17 USAGE OF COMMUNICATIONS IOP

The Communications IOP will be used by the MATDEV and industry to define communications and radio related requirements for a CCL, to point to applicable sections in the other IOPs, and to define communications related Interoperability Attributes that may be selected by the MATDEV to impose interoperability requirements into acquisition contracts.

6.18 USAGE OF PAYLOADS IOP

The Payloads IOP will be used by the MATDEV and industry to define payload related requirements for both payloads themselves and UGV platforms, to point to applicable sections in the other IOPs, and to define payload related Interoperability Attributes that may be selected by the MATDEV to impose interoperability requirements into acquisition contracts.

6.19 USAGE OF CONTROL IOP

The Control IOP will be used by the MATDEV and industry to define desired common qualities of controllers. It is acknowledged that there are a variety of types of controllers that make sense for different missions, and the technology related to controllers is evolving rapidly, particularly based on advancements being made in the mobile/smartphone and gaming markets. The current Control IOP contains desired guidelines for user interfaces for conventional controllers but does not mandate explicit requirements. Controllers must be capable of communicating JAUS-based messages as defined in the IOP and must interface with the CCL as defined in the Communications IOP. The primary intent of the Control IOP is to promote an interoperable Human Machine Interface (HMI), which means that the relationship between the controller and the human operator must be modular based on minimized training for operation among different systems. If controllers can support the JAUS-based messages in the IOP, then interoperable messages will become the interface between the controller and the UGV platform.

For example, if a controller operator presses a keypad arrow to turn right, then an interpretable message command will be received and understood by the UGV platform. If another controller utilizes a joystick to turn right, then the UGV platform should receive a message similar as that sent from the first controller. User input to turn right on a smartphone type accelerometer device, a speech-based device, a motion-recognition device, or other innovative controller technology should all result in a similar functionality, interpretable JAUS-based message being received by the UGV platform.

6.20 ROS-M

The primary goal of the IOP is to define the physical, electrical, and logical interfaces between platforms, payloads, and devices for unmanned ground systems. This promotes interoperability between such devices,

without prescribing any internal implementation. This allows vendors and suppliers to use whatever tools, intellectual property, or proprietary implementations they wish. The IOP does not encourage one solution over another, so long as the interfaces are compliant.

As a result, the IOP can be seen as agnostic of, or compatible with, common software tool efforts such as the Robotic Operating System – Military (ROS-M). ROS-M is an on-going effort to promote ROS, maintained independently by the Open Source Robotics Foundation (OSRF), by developing and fostering an ecosystem of stabilized, validated, and approved ROS 2.0 packages. The goal is to promote rapid development through shared implementation of common robotic behaviors freely accessible as open source software.

Since the IOP emphasizes interfaces while ROS-M focuses on implementations, the two can be used together or entirely independently of one another. If both are used within an implementation, it will be necessary to convert or bridge between the IOP/JAUS messages used by the interface and the ROS/DDS topics used by the internals.

6.21 LOGISTICS AND SA

The IOP enables interoperability between robotic platforms, controllers, and payloads. While intended for use in a wide variety of configurations, the most commonly fielded systems to date generally involve a single human using a single controller to remotely operate a single platform. However, multiple experiments and projects demonstrate the applicability of IOP for coordination of multiple platforms with emphasis on supervisory control of autonomous and automated behaviors.

As robotic and autonomous ground systems grow in capability, they are more likely to be integrated in larger scale Situational Awareness (SA); Command and Control (C2); and Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance (C4ISR) systems. These systems often define interface standards of their own, likely with little overlap of those from the IOP. However, it is expected that these interfaces can be used simultaneously: that is, a controller, platform, or payload may be compliant to an IOP instantiation in addition to any other interface standards required by the project or program. Note that such compliance may require additional radios, physical interfaces, and logical communications.

6.22 SAE

The IOP, and particularly the logical interface definitions, makes extensive use of the JAUS services published by the SAE AS-4 standards body. Where necessary, the IOP extends published JAUS services by providing additional profiling rules and custom services to support a wider variety of projects and programs. In some cases, these custom services are being provided back to the AS-4 committee for possible adoption. The IOP is expected to maintain this close relationship with the AS-4 JAUS body going forward.

Furthermore, the Unmanned Systems Control Segment (UCS) Architecture has been brought under the AS-4 group. UCS uses a Model Driven Architecture (MDA) approach to define a Platform Independent Model (PIM) of the data being exchanged. While some examples and projects have been created to define a Platform-Specific Model (PSM) to demonstrate plug-and-play interoperability, this is not currently the main emphasis of the effort. Ultimately, AS-4 looks to align JAUS and UCS such that JAUS could be considered a specific PSM for an extended UCS architecture. This is likely a long-term process, and it is not expected that this alignment will have significant impact on the IOP.

6.23 GVR-BOT

6.23.1 Legacy PackBot

iRobot (Endeavour Robotics) PackBot is a fielded small robot used primarily for IED interrogation. The base platform is expensive to purchase and expensive to maintain. The design is “semi-modular” – payloads can be installed to the base chassis, but they can only be installed in specific locations, and the system must have the proper software installed and configured with any new payload. This has led a dependence on the manufacturer for integration of new payloads. US Army Robotic Systems – Joint Program Office funded a standardization effort to create a kit for retrofitting the PackBot to make them IOP compliant and reduce some of the other sustainment costs.

6.23.2 VR-BOT

GVR-BOT (Figure 6-4) is a continuation of the standardization effort to create a good government research platform with an Open Architecture that is IOP compliant.

The base capabilities include:

Weight: (Without Batteries and Flippers): 11.8 kg (26 lbs).

Size: 68.0 X 40.5 X 18.0 cm (26.8 X 15.9 X 7.1 in).

Top Speed: 2.0 m/s (4.4 mph).

Ambient Temperature Range: -20 to +50°C (-4 to +122°F).

Main Processor:

- Intel Atom E680T, 1.6 GHz.
- 1GB DDR2 soldered RAM.
- 3.6GB soldered NAND.

Power:

- Up to four BB-2590 Li-Ion batteries (requires one battery cradle per two batteries).
- Base chassis consumes approximately 30 Watts when powered up and stationary.

Payloads:

- Four payload ports.
- Three payload bays can physically contain existing PackBot 510 payload tub sized modules.
- Mounting rails for the fourth payload bay can be installed above the Front Electrical Housing (not included).

Communications:

- Communicates with any Wi-Fi enabled computer at 2.4 GHz frequency.
- Maximum communications range is ~250m.
- Internal router separates internal network from external communications.

Software Interface:

- Designed to meet the IOP V1 standard, internally (between payload components) and externally (to OCU).

Sensors:

- Internal Altitude Heading Reference System (AHRS) provides accelerometer, heading, and orientation information.
- Internal GPS Transceiver with external antenna port.

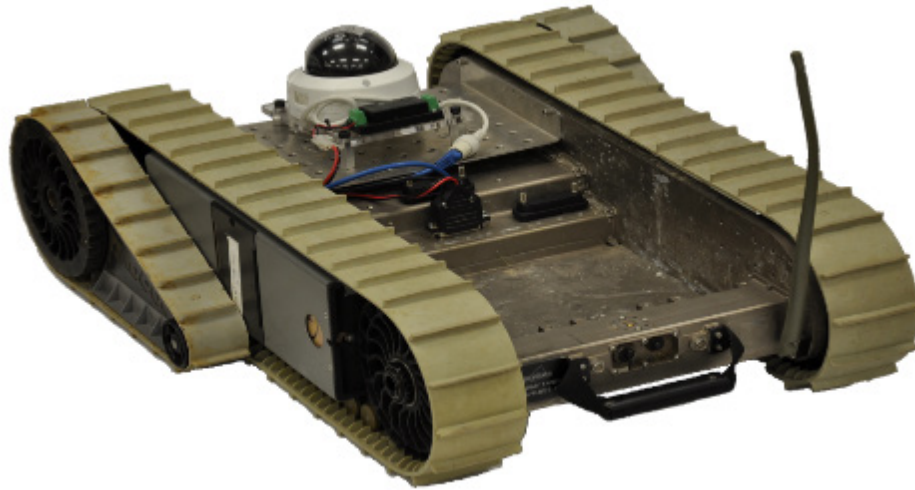


Figure 6-4: GVR-BOT.

6.24 MAPPING SENSOR

Using a scanning Lidar, an accurate digital map can be produced for digital mapping of rooms and buildings by driving the GVRBot around (Figure 6-5). Having a robust mapping and localization system allows for waypoint navigation, automated exploration, sensor fusion (chemical, biological, radiation, nuclear and temperature as fused overlays) and object classification. This will assist soldiers in search and cordon of potentially hazardous areas.

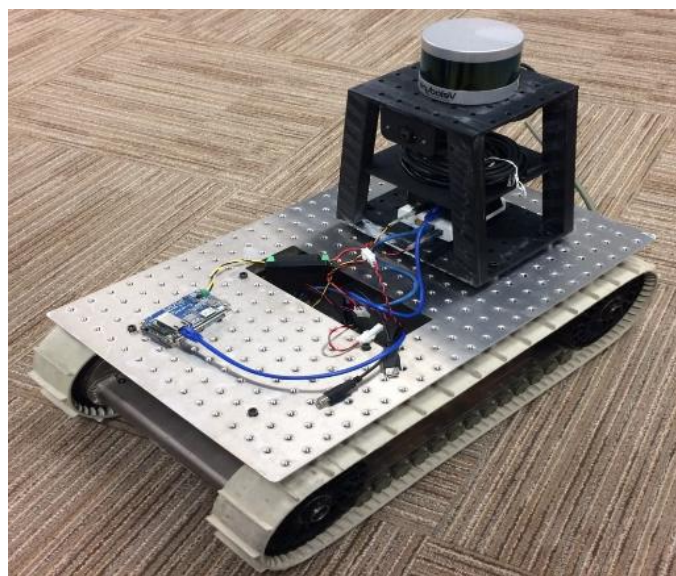


Figure 6-5: Mapping Sensor Mounted on GVR-BOT.

6.25 CONTROLLER

The system is based on a consolidated piece of hardware for control of unmanned systems. The basis is a commercial ruggedized 7" tablet computer with an Intel m5 processor (Figure 6-6). The tablet is wrapped in Hapflex (rugged plastic) enclosure to provide for realistic Human Machine Interface with rugged I/O. The total system weight is 4 lbs.



Figure 6-6: Controller.

The controller includes two joysticks, two covered switch guards, deadman grip switch, touchscreen and optional second screen (not shown in the current picture).

The controller was used in RTG-149 to control GVR-BOT, Telexmax, OLAV and GARM robots which was enabled using the IOP. When the controller was started it showed available IOP compliant robots on its screen. By selecting any of the available robots the controller was able to take control and tele-operate these robots using either one of the two joysticks.



Chapter 7 – NORWEGIAN DEFENCE RESEARCH ESTABLISHMENT

K. Mathiassen, and M. Baksaas
Norwegian Defence Research Establishment
NORWAY

ABSTRACT

The Norwegian Defence Research Establishment (FFI) hosted the trials at Rena from the 11th of June to the 15th of June. This chapter first describes the organization of the trials. Then it describes FFI's contribution in the trails, i.e., providing the mobile robot Olav and adapting the system to use the IOP-ROS bridge in order to be compliant with the standard, and setting up an operation center at the trial site.

7.1 TRIAL ORGANIZATION

The trial was held at the training site Rødsmoen, which is adjacent to Rena Army Camp. Rena Army Camp is one of the largest camps in the Norwegian Army and is located in Åmot municipality. The location is indicated in the map given in Figure 7-1. The trial was conducted at a close quarter's combat training ground at Rødsmoen.



Figure 7-1: Trial Location in Norway.

7.1.1 Trial Location – Close Quarters Combat Training Ground

A satellite image of the Close Quarters Combat Training Ground is shown in Figure 7-2. Marked 1 on the map is where the operation center was set up. It was a temporary housing unit with heating and electricity, and it was used for development and running the operation. Fraunhofer FKIE brought an operation center container, and this was placed at the point marked 2 on the map. The area marked 3 was used as a staging area for the robots before the operation started. At the point marked 4 the mock explosive device was placed.



Figure 7-2: Map Showing the Trial Area.

7.2 IOP IMPLEMENTATION ON ROBOT

FFI contributed with the robotic system Olav (Off-road Light Autonomous Vehicle) for the trials. First the system will be described, before the adaption of the system to use IOP is described.

7.2.1 Olav Description

In 2015, the Norwegian Defence Research Establishment started a project to develop an autonomous vehicle for off-road driving. Parts of that development are documented in Ref. [34]. In the on-going project Olav has two primary missions. The first is to navigate autonomously to a specific location in order to observe a sector. It should be able to observe the scene of interest for a prolonged period of time. The second mission is aiding a base protection team, where Olav can be sent out to check the perimeter of the base. Here, Olav will primarily drive autonomously, but sometimes remote operation is needed to do fine navigation of the vehicle.

Olav is based on a Polaris Ranger XP 900 EPS vehicle and has been modified to be controlled autonomously. It has been equipped with cameras and a LiDAR as perception sensors and an Inertial Navigation System (INS) for localization. In addition, a Protector Lite Remote Weapon Station (RWS) from Kongsberg Defence and Aerospace is mounted on the vehicle. The vehicle is shown in Figure 7-3. The vehicle uses the Robotic Operating System (ROS) as middleware, and thus run mainly on Ubuntu Linux.



Figure 7-3: The Robotic System Olav.

The control architecture of Olav is shown in Figure 7-4. A control station developed at the Norwegian Defence Research Establishment is used to define missions for the vehicle, and the missions are defined in a modified Coalition Battle Management Language (C-BML) [35]. The control station is denoted FFI GCS (Ground Control Station) in Figure 7-4. The mission is defined by specifying tasks the vehicle should execute, and each task is defined by specifying five key elements: *Who*, *What*, *When*, *Where*, and *How*. *Who* defines who will execute the order, *what* defines what kind of task the vehicle should perform, *when* is currently set to as soon as possible. *Where* defines the operation area of the task, or a route or location the vehicle should move to, and *how* defines additional information on how the task should be executed (for instance maximum speed of the vehicle).

A higher level decision making module has been implemented by adapting an existing framework used on Autonomous Underwater Vehicles [36] and is shown in Figure 7-4 as “decision making”. This module composes a complex mission into a task tree. A task can either be a parent task of a child task. A parent task has one or more child tasks as leaf nodes on the tree, while child tasks do not have leaf tasks. Each child task has one or more processes that are triggered periodically, and in the framework, these are called behaviours. The behaviours implement how the robotic system should achieve the goal of the task, and monitors if the task goal is achieved and the task is completed. When a task is completed the next leaf node in the tree starts to execute. The decision making module use a route planning service. The route planning service provides routes planned used geographical map data.

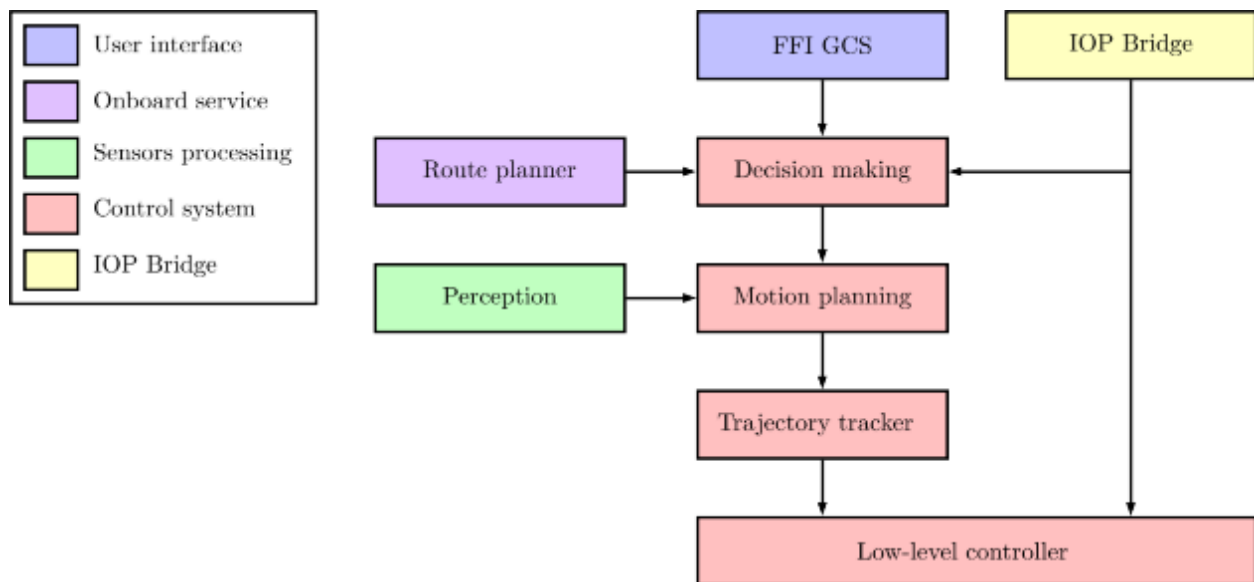


Figure 7-4: Control Architecture of Olav.

A perception module [37] has been developed that produces a map over the surrounding area of the vehicle. Cameras and LiDAR are the primary sensors when producing the map. The map estimates the driveability of the surrounding area and is used by a motion planning component. This component plans a driveable trajectory that avoids obstacles and at the same time follows the desired route. A route can be given by a user or planned, using an on-board route planning service. The motion planning component plans a new trajectory every other second and sends this to the Trajectory tracker component. This component ensures that the vehicle follows the trajectory by sending control signals to the low level controller. These signals are the steering angle set point and the desired forward velocity.

In addition, there is a localization component (an Inertial Navigation System) not shown in Figure 7-4. This component uses GPS and an inertial measurements unit to estimate the global position and orientation of the vehicle. It sends the position and orientation estimate to almost all the components and is therefore not included.

7.2.2 IOP Implementation on Olav

The system described above does not fit well into the direct tele-operation architecture used in this experiment. Therefore, it was necessary to make adaptations in the system in order for the system to be compliant with IOP. The main adaptation was to use the Fraunhofer FKIE IOP bridge for ROS [38], and to use Gstreamer [39] for providing Real-Time Streaming Protocol (RTSP) servers for publishing video in IOP. Both these additions will be described in greater detail in the following subsections.

In addition, the higher level decision module was modified to include a task for tele-operation. This task is started when someone requests control of Olav through IOP. This overrides the current mission, and sets it on hold, until the control is released, and the operator restarts the current mission in the GCS. In this way Olav can conduct a mission, and the operator can have the ability to override the vehicle by tele-operating it through IOP, and then restart the mission at a late stage. In Figure 7-4 the IOP is connected to the decision making module, to indicate that the bridge notifies that someone has requested control of the vehicle. It is also connected to the low level controller, as the IOP bridge bypasses the rest of the control system and controls directly the velocity and steering angle of the vehicle.

7.2.2.1 Using Fraunhofer IOP Bridge

Since Olav is already running ROS, all that is required to be IOP compliant is to install the Fraunhofer FKIE ROS/IOP bridge, configure it and map the correct topics to and from the bridge. The bridge is configured by creating a ROS launch file that will launch one or more ROS nodes. Each ROS node contains a set of JAUS services. Olav was configured on the basis of the example configuration in Fraunhofer ROS/IOP bridge.

The first ROS node is configured as the platform manager node. The JAUS services provided by this node are given below.

```
iop_transport_fkcie: "Transport",
iop_events_fkcie: "Events",
iop_events_fkcie: "EventsClient",
iop_accesscontrol_fkcie: "AccessControl",
iop_discovery_fkcie: "Discovery",
iop_discovery_fkcie: "DiscoveryClient",
iop_management_fkcie: "Management",
iop_liveness_fkcie: "Liveness",
iop_digital_resource_discovery_fkcie: "DigitalResourceDiscovery",
iop_health_monitor_fkcie: "HealthMonitor",
iop_platform_mode_fkcie: "PlatformMode",
iop_platform_state_fkcie: "PlatformState",
iop_primitive_driver_fkcie: "PrimitiveDriver",
iop_local_pose_sensor_fkcie: "LocalPoseSensor",
iop_range_sensor_fkcie: "RangeSensor",
iop_global_pose_sensor_fkcie: "GlobalPoseSensor",
iop_handoff_fkcie: "EnhancedAccessControl",
```

We will comment on the most important JAUS services in this node. The PrimitiveDriver services advertise that the robot can be driven by an Operator Control Unit (OCU). In the specification of IOP it is stated that this message is a six degree of freedom effort request to the vehicle, and that the effort is specified as a percentage of maximum effort. The bridge converts this effort into velocity by specifying a maximum velocity in each degree of freedom, and multiplying this maximum velocity with the percentage effort value to get the desired velocity. Olav is an Ackermann vehicle, and the primitive driver does not fit with such a vehicle since you cannot have angular velocity in yaw when you have zero forward speed. To overcome this issue, we convert the angular velocity to desired steering angle of the front wheels and use the desired velocity directly.

The GlobalPoseSensor sends the global position to the OCU. We had to convert the ROS messages from Olav's navigation system into the appropriate messages. The JAUS service in the bridge advertises two ROS topics of the type sensor_msgs::NavSatFix and sensor_msgs::Imu. The first is a message typically obtained from a GPS. However, we have a complete navigation system on Olav, so we report the position found by the navigation system. The second message is data from an Inertial Measurement System (INS), and here we forward the INS data. For the LocalPoseSensor we do not publish anything.

The EnhancedAccessControl JAUS service in the bridge publishes a ROS topic that says if someone is controlling the robot or not. This is used by the higher level decision module in Olav to interrupt the current mission. Whenever there is a change in this value the higher level decision module acts. If it goes from uncontrolled to controlled, the current mission is interrupted and a tele-operation task starts to run. Whenever it goes from controlled to uncontrolled the tele-operation task is removed, and the higher level decision module waits for the operator to start the current mission again.

The second ROS node is configured to publish the video feeds to IOP, and the JAUS services provided by this node is given below.

```
iop_digital_resource_discovery_fkie:"DigitalResourceDiscoveryClient",
iop_digital_video_fkie: "DigitalVideo",
iop_visual_sensor_fkie: "VisualSensor",
iop_liveness_fkie: "Liveness",
iop_discovery_fkie: "DiscoveryClient",
iop_accesscontrol_fkie: "AccessControl",
iop_events_fkie: "Events",
iop_events_fkie: "EventsClient",
iop_transport_fkie: "Transport",
```

The important JAUS services here are VisualSensor and DigitalVideo. The VisualSensor service defines a name and potentially some parameters for the visual sensor. The DigitalVideo service defines the URL of the video provider. For Olav's case two visual sensors are defined. One for a forward facing camera and one for the RWS video feed.

7.2.2.2 Using Gstreamer to Provide RTSP Stream

Gstreamer is a powerful tool for creating multimedia pipelines, for instance for sending compressed video over a network. IOP requires that all video should be made available using RTSP servers and the video should be encoded using H.264 encoding.

In Olav there are several video sources. There are three cameras mounted on the roof of the vehicle. In addition, there are video feeds generated by the perception module, showing parts of the image processing or displaying how the vehicle perceives the world. All these feeds are available in the perception framework. In addition, there is a video feed from the RWS on Olav, which is made available by a frame grabber on Olav.

The perception framework was extended to export video as an RTP stream with H.264 encoding through UDP, locally on Olav. This was implemented using the gstreamer C API, and using the appsrc plugin [40]. This plugin allows the application (in this case the perception module) to send images to gstreamer. Gstreamer then creates a video of the images and encodes the video. An example pipeline generated by the perception module is given below, and a detailed explanation of the pipeline steps (each step is separated with a ! sign) is given in Table 7-1.

```
appsrc name="mysource" caps="video/x-raw, framerate=1/25, width=1920,
height=1080, format=BGR" ! videoconvert ! x264enc bitrate="500"
tune=zerolatency speed=preset=ultrafast ! video/x-h264,profile=main !
rtph264pay ! application/x-rtp,payload=96 ! udpsink host=localhost
port="9999"
```

Table 7-1: Detailed Explanation of the Streaming Pipeline in the Perception Module.

Pipeline Step	Description
appsrc name="mysource" caps="video/x-raw, framerate=1/25, width=1920, height=1080, format=BGR"	This describes the source of the pipeline. In this case an application supplies images to the pipeline, and the format is in the caps parameter. BGR specifies the color format, Blue, Green and Red in this case.

Pipeline Step	Description
videoconvert	This step converts the video to the appropriate format for the next step in the pipeline.
x264enc bitrate="500" tune=zerolatency speed- preset=ultrafast	This step encodes the video to H.264, using a low latency preset.
video/x-h264,profile=main	This step specifies the capabilities (caps) of the stream. This help gstreamer to link to steps correctly, so that the output of the previous step matches the input of the next step. This step does not perform any action to the stream.
rtph264pay	This step encapsulates the stream using the stream using the Real-time Transport Protocol (RTP).
application/x-rtp,payload=96	This is another capability step and does not do any action on the stream.
udpsink host=localhost port="9999"	This step sends the stream out on the network using UDP. In this case it is only sent locally on the computer (because the host is specified as localhost).

The stream produced by the perception module needs to be made available by a RTSP server. Gstreamer has a RTSP server library with an example server [41]. We used the example server and provided that with a pipeline. This pipeline needs to get the stream from the perception module and make it available for the server. We would also like to avoid decoding the video and encoding it again, because this would increase the time delay of the video. The pipeline used in the server application is given below, and a detailed explanation of the pipeline steps is given in Table 7-2.

```
udpsrc address=localhost port=9999 ! application/x-rtp,payload=96 !
rtph264depay ! h264parse ! video/x-h264, profile=main ! rtph264pay
name=pay0 pt=96
```

Table 7-2: Detailed Explanation of the Streaming Pipeline for the RTSP Server Publishing the Perception Module Video Feed.

Pipeline Step	Description
udpsrc address=localhost port=9999	This describes the source of the pipeline. Here we get UDP packets from localhost, i.e., the UDP packets sent by the perception module.
application/x-rtp,payload=96	This defines the capabilities of the UDP source.
rtph264depay	This step receives the RTP stream and provides the H.264 video to the next step in the pipeline.
h264parse	This step parses the H.264 video, in order to know the format of the video. Note that it does not decode the video.

Pipeline Step	Description
video/x-h264, profile=main	This defines the capabilities of the video.
rtph264pay name=pay0 pt=96	This step encapsulates the stream again using RTP. The name pay0 is necessary in order to have the server application catch the stream and make it available.

To make the RWS video stream available we use a slightly different pipeline, as the source of this video feed does not originate from the perception module. The frame grabber is available similarly as a web camera in Linux. The pipeline below is used in the server application, and a detailed explanation of the pipeline steps is given in:

```
v4l2src device=/dev/video0 ! x264enc bitrate=500 tune=zerolatency speed-
preset=ultrafast ! video/x-h264, profile=main ! rtph264pay name=pay0
pt=96
```

Table 7-3: Detailed Explanation of the Streaming Pipeline for the RTSP Server Publishing the RWS Video Feed.

Pipeline Step	Description
v4l2src device=/dev/video0	This pipeline takes a hardware video device as source, in this particular case a frame grabber.
x264enc bitrate=500 tune=zerolatency speed- preset=ultrafast	This step encodes the video to H.264, similarly to how this was done in the perception module.
video/x-h264, profile=main	This defines the capability of the encoded video.
rtph264pay name=pay0 pt=96	This encapsulates the video using RTP and provides this to the server application.

Using the above pipelines Olav provides two RTSP servers in the IOP framework.

7.3 EXPERIMENT SETUP

FFI's main responsibility in setting up the experiment was to provide the central switch for the network, where everyone connected their robots and OCUs, connecting Olav to the network and setting up the operation center. This is further described in the following subsections.

7.3.1 Network Setup

The network used in the experiment used a subnet for communication and avoided the usage of routers to route messages between different subnets. This made net network setup much easier. Each participant was responsible for connecting their device (robot or OCU) to the central switch.

To mask away the internal network on Olav we used a router with NAT setup. We set up the router to forward multicast messages, so that IOP traffic could go through the router. In addition, we set up port

forwarding to forward connection request to the RTSP server to the correct computer on Olav. Figure 7-5 shows the network setup on how Olav is connected to the network. The orange components are on the internal net on Olav, while the blue are on the common IOP network. The main components on Olav's internal net are the Olav controller computer, which runs the control software and IOP bridge, the perception computer, running the perception module and the RTSP server for the forward camera, and the RWS controller, which control the RWS and has a secure link to the RWS control station in the operation center.

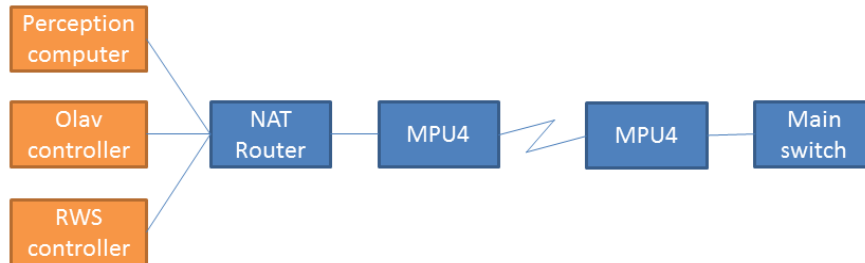


Figure 7-5: Network Setup from Olav's Perspective.

The NAT router is connected with the common IOP network through the mesh radios MPU4 from Persistent systems. These radios provide up to 20 Mbit/s bandwidth and form a network layer 2 mesh. During the experiment one radio was used as relay. This reduced the available bandwidth by a factor of two.

7.3.2 Operation Center Setup

An operation center was set up and is shown in Figure 7-6. Here two operators use the OCUs from TARDEC and PIAP. In addition, the RWS control station is set up (to the right in the figure). On the wall a projector displays a video feed from each robot. The video feed was set up using gstreamer, in order to have a low latency viewer for the streams. Each stream was started with the following Linux command

```
gst-launch-1.0 rtspsrc latency=0 location=$URI ! rtph264depay !
avdec_h264 ! autovideosink
```

Where \$URI is the RTSP server URI (for instance rtsp://192.168.0.2:8555/test). The RTSP server URI was obtained from the IOP messages, and each viewer was started manually.



Figure 7-6: The Operation Center.

A screen showing the location of all the robots in a map was also set up. Mapviz [42] in ROS was used to show the map and the positions. Four ROS OCUs were started using the Fraunhofer ROS/IOP bridge. We used the example OCU provided in Ref. [43] and made a launch file launching four of these OCUs with different IOP addresses. Then we used the ROS RQT GUI provided with the bridge to be an observer for the four robots.

7.4 EXPERIENCES FROM TRIALS

Before the experiment IOP was implemented on Olav using the Fraunhofer ROS/IOP bridge. TARDEC had also developed a bridge, but since the Fraunhofer bridge was available at an earlier stage this was used. Then we avoided changing bridge in the middle of the integration. The availability of a bridge made it much easier and faster to get an IOP compliant robot. And this is an important point in order to get more robots to be IOP compliant. However, we found that it is important to have some knowledge of the IOP standard and the bridge you are using. All robots may have differences in their control architecture, and Olav has some differences compared to the robots used by Fraunhofer. We extended the bridge to also publish if someone was controlling the robot or not. This is important as Olav has two modes, autonomous mode and tele-operated mode. When Olav is in autonomous mode and then someone starts to control Olav through IOP, we would like the decision module to be interrupted and hand over the control to the user. This was not possible when using the bridge originally. Fraunhofer updated their bridge based on our changes and needs.

Another issue is that IOP is a large standard, so before the experiment all involved parties agreed on a set of IOP messages that should be supported. This made it easier to get the robots to be IOP compliant but came at a cost. The PrimitiveDriver was used to control the robots, but this may not be the best way to control an Ackermann vehicle.

In our opinion the trials were very successful. All OCUs managed to control all the robots. However, some difficulties came up during the experiment. One of them was interference between different radios. This was not a problem associated with IOP, but rather on how IOP data could be shared in the network. After going through the different frequency ranges for the different radios, we managed to avoid this problem by changing the channel used by one of the radios. However, this is just a symptom for a larger problem, how to connect all the robots in a network. Ideally all the robots should use the same radios and form a mesh network back to the operation center. This might not be possible in practice, for several reasons. In our case we did not have a sufficient amount of radios, and each team was responsible for their own radio communication. In other cases, the robotic system could be delivered with a radio.

Also, routing could potentially be a problem when connecting the robots together. Many of the robots have their own internal network, and it might be available for the entire network. This could pose a problem when connecting robots together, as the internal networks might be on the same subnet and have conflicting IP addresses. This was solved in this experiment by placing a NAT router on the robot to mask the internal network away from the common network.

The last issue that appeared was that some of the OCUs made assumptions that were not part of the standard. For instance, the OCU provided by TARDEC only accepted robots with the name "Packbot." This caused us to have four Packbot robots in the experiment and made it harder to know where each robot was. Another issue was that one of the controllers only accepted one video stream, even though two were available. This was easily solved by reducing the number of available streams.

For the experiment we defined a scenario that we played out a number of times. We had two operators controlling the four robots. In our opinion IOP provided a convenient way of controlling multiple robots. Additionally, having all the video sources available and the positions of all the robots is very valuable for the operators when they are cooperating in performing a task. Testing with military personnel is needed to verify these results. The experiment motivated FFI to continue to participate in the IOP work in NATO.

Figure 7-7 shows all the participants and robots involved in the experiment.



Figure 7-7: All Experiment Participants and Robots.

7.5 MEDIA COVERAGE AND DISSEMINATION

The Communications Department at FFI came to the last day of the trials and made videos and a popular science article about the trials. The article was published on FFI's webpages, and at forskning.no, which is a Norwegian online paper reporting on Norwegian and international research news. The links to the two online papers are found below:

- <https://forskning.no/forsvarets-forskningsinstitutt-partner-krig-og-fred/slik-kan-nato-robotene-samarbeide/1195257>.
- <https://www.ffi.no/aktuelt/nyheter/slik-kan-nato-robotene-samarbeide>.

In addition, the Communications Department at FFI made two videos from the event. One video describes the trials, and one has an interview with Paul Bunker (the leader of the Team of Expert UGV group under NATO NAAG LCG/LE). The links for the videos are given below:

- <https://vimeo.com/277425404>.
- <https://vimeo.com/276862970>.

A group from Rena Army Camp, including a person from the project advisory group of the UGV project at FFI, came on the last day for a small demonstration. The group demonstrated the scenario they had rehearsed and explained the capabilities of the robots and the main objective of the trials.



Chapter 8 – FRAUNHOFER INSTITUTE FOR COMMUNICATION, INFORMATION PROCESSING AND ERGONOMICS (FKIE)

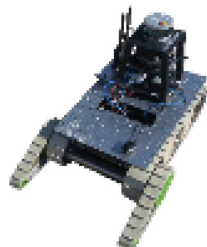
A. Tiderko and Dr F.E. Schneider
Fraunhofer Institute for Communication
GERMANY

8.1 INTEROPERABILITY TRIAL PARTICIPATION

The interoperability trial was held at a training site supported near to Rena in Norway. Six nations with different hardware (OCUs and UGVs) participated at this trial. Germany was represented by BAAINBw and FKIE. For scenario participation, a robot of type Telemax was provided by BAAINBw and FKIE, supported by a robot control station mounted into a container shelter of type “Fernmeldekabine 2B”. Furthermore, an IOP-compliant bridge software for ROS-based control software was provided by FKIE and was also used by other trial participants in their hardware. This particularly concerned the robot “GARM” which is the product of cooperation between RUAG and FKIE. For the interoperability trial, the payload with all sensors on it and the control software was created by FKIE. Table 8-1 shows an overview of all hardware that was used on the trials in Rena.

Table 8-1: Participants’ Hardware for the Trial.

	OCU	Robot / Payload
FFI (Norway)		
FKIE (Germany)		
PIAP (Poland)		

	OCU	Robot / Payload
RUAG (Switzerland)		
TARDEC (US)		
WTD41 (Germany)		

8.1.1 Robot System “Telemax”

Telemax is a small robot build by Telerob. The approximately 80 kg robot (without battery) is designed for use in cramped confines. The 7-axis manipulator has a long reach and can lift weights between 3 and 20 kg depending on the manipulator position. With his four chain flippers it can easily overcome an obstacle and reach speeds of up to 10 km/h.

By default, Telemax has its own control panel. However, this control panel has no option to forward the sensor data and control commands to other control instances. To control Telemax through the IOP we should connect to Telemax directly. For this purpose, the robot has an Ethernet and a CAN interface. A control computer can be connected to these and the robot can be extended with assistance and autonomy functions.

The Autonomiebox is an extension, which is available as an accessory for the Telemax. This is made in the form of a battery and replaces one of the two batteries on the robot, see Figure 8-1. Internally, the Autonomy Box is equipped with an Intel i7 quad-core processor, an Inertial 6-Axis Measurement Unit (IMU) and a Wi-Fi hotspot. In addition, the Hokuyo 2D laser scanner and Xbox Kinect are mounted on the box, which can be used for local navigation. The Autonomiebox communicates with Telemax via CAN interface.

The internal cameras of the Telemax provide only an analogue signal. To get digital video for IOP an analogue-to-digital converter with Power-over-Ethernet (PoE)-injector was attached to the Autonomiebox (Figure 8-2).



Figure 8-1: Robot System Telemax.



Figure 8-2: Autonomiebox for Telemax.

The control software in the Autonomiebox is based on ROS. However not all functions of the robot are properly supported by the ROS Telemax driver, in the current state. Therefore, not all skills of the robot could be fully utilized during IOP tests in Rena. To get this system compliant with IOP, FKIE ROS/IOP bridge [38] for ROS was used. The ROS/IOP bridge on Telemax provides following services:

<i>urn:jaus:jss:core:Transport</i>	<i>v1.1</i>
<i>urn:jaus:jss:core:Events</i>	<i>v1.1</i>
<i>urn:jaus:jss:core:AccessControl</i>	<i>v1.1</i>
<i>urn:jaus:jss:core:Discovery</i>	<i>v1.1</i>
<i>urn:jaus:jss:core:Management</i>	<i>v1.1</i>

<i>urn:jaus:jss:core:Liveness</i>	<i>v1.1</i>
<i>urn:jaus:jss:exp:aeodrs:HealthMonitor</i>	<i>v1.4</i>
<i>urn:jaus:jss:environmentSensing:DigitalVideo</i>	<i>v1.0</i>
<i>urn:jaus:jss:environmentSensing:RangeSensor</i>	<i>v1.0</i>
<i>urn:jaus:jss:environmentSensing:VisualSensor</i>	<i>v1.0</i>
<i>urn:jaus:jss:mobility:GlobalPoseSensor</i>	<i>v1.0</i>
<i>urn:jaus:jss:mobility:LocalPoseSensor</i>	<i>v1.0</i>
<i>urn:jaus:jss:mobility:PrimitiveDriver</i>	<i>v1.0</i>
<i>urn:jaus:jss:iop:DigitalResourceDiscovery</i>	<i>v1.1</i>
<i>urn:jaus:jss:iop:PlatformMode</i>	<i>v1.1</i>
<i>urn:jaus:jss:iop:PlatformState</i>	<i>v1.1</i>
<i>urn:jaus:jss:iop:EnhancedAccessControl</i>	<i>v1.1</i>

Due to the limitations of the firmware in combination with ROS driver for Telexis not all functions of the Telexis could be used during the IOP tests. These include GPS, control of the flippers, manipulator and switching of the internal cameras. In the meantime, most of them are already integrated into ROS driver.

8.1.2 Payload for Robot System “GARM”

There are a large number of possible applications for unmanned platforms. Depending on different scenarios, the platform must be equipped with different sensors and tools. While planning the robot platform GARM a modular payload concept was introduced. As a result, the payload with sensors of a robot can be easily exchanged and adapted to the required application scenario. Figure 8-3 shows the reconnaissance payload used while IOP tests in Rena.



Figure 8-3: Payload for Reconnaissance Scenario.

The payload consists of the following sensors:

- Three Acti Ethernet cameras, a 360° camera to observe the surrounding area of the robot and two cameras for the lateral periphery, to allow a better overview in bottlenecks.
- Oculus: a pan-tilt camera with a zoom lens and a thermal camera.
- Searchlight with pantilting function.
- GPS receiver.
- Zelos-C mesh radio.
- PC with robot control software.
- Various Wi-Fi receivers to simulate ABC sensors.
- Connections for external laser scanners, which are attached to the robot platform.

The payload is connected to the robot platform via defined power and communication connections (usually RJ45), see Figure 8-4. The robot platform is controlled by the PC in the payload. It also provides assistance and autonomy functions adapted to the application scenario. The control software in payload is based on ROS.



Figure 8-4: Robot Chassis GARM from RUAG and Payload from FKIE.

For interoperability tests in Rena, all of the payload sensors were deployed via ROS/IOP bridge for access via IOP-compliant OCU. In addition, autonomous waypoint navigation was integrated, which was also IOP-compliant and successfully tested with both the FKIE's own IOP-compliant OCU and the polish POCU.

The ROS/IOP bridge on the payload provides following IOP services:

<i>urn:jaus:jss:core:Transport</i>	<i>v1.1</i>
<i>urn:jaus:jss:core:Events</i>	<i>v1.1</i>
<i>urn:jaus:jss:core:AccessControl</i>	<i>v1.1</i>
<i>urn:jaus:jss:core:Discovery</i>	<i>v1.1</i>
<i>urn:jaus:jss:core:Management</i>	<i>v1.1</i>
<i>urn:jaus:jss:core>ListManager</i>	<i>v1.1</i>
<i>urn:jaus:jss:core:Liveness</i>	<i>v1.1</i>
<i>urn:jaus:jss:exp:aeodrs:HealthMonitor</i>	<i>v1.4</i>
<i>urn:jaus:jss:environmentSensing:DigitalVideo</i>	<i>v1.0</i>
<i>urn:jaus:jss:environmentSensing:RangeSensor</i>	<i>v1.0</i>
<i>urn:jaus:jss:environmentSensing:StillImage</i>	<i>v1.0</i>
<i>urn:jaus:jss:environmentSensing:VisualSensor</i>	<i>v1.0</i>
<i>urn:jaus:jss:mobility:GlobalWaypointListDriver</i>	<i>v1.0</i>
<i>urn:jaus:jss:mobility:GlobalWaypointDriver</i>	<i>v1.0</i>
<i>urn:jaus:jss:mobility:GlobalPoseSensor</i>	<i>v1.0</i>
<i>urn:jaus:jss:mobility:LocalPoseSensor</i>	<i>v1.0</i>
<i>urn:jaus:jss:mobility:PrimitiveDriver</i>	<i>v1.0</i>
<i>urn:jaus:jss:manipulator:PanTiltSpecificationService</i>	<i>v2.0</i>
<i>urn:jaus:jss:manipulator:PanTiltJointPositionDriver</i>	<i>v2.0</i>
<i>urn:jaus:jss:manipulator:PanTiltJointPositionSensor</i>	<i>v2.0</i>
<i>urn:jaus:jss:manipulator:PrimitivePanTilt</i>	<i>v2.0</i>
<i>urn:jaus:jss:iop:CostMap2D</i>	<i>v1.1</i>
<i>urn:jaus:jss:iop:DigitalResourceDiscovery</i>	<i>v1.1</i>
<i>urn:jaus:jss:iop:MeasurementSensor</i>	<i>v1.1</i>
<i>urn:jaus:jss:iop:PlatformMode</i>	<i>v1.1</i>
<i>urn:jaus:jss:iop:PlatformState</i>	<i>v1.1</i>
<i>urn:jaus:jss:iop:PathReporter</i>	<i>v1.1</i>
<i>urn:jaus:jss:iop:EnhancedAccessControl</i>	<i>v1.1</i>
<i>urn:jaus:jss:ugv:PowerPlantManager</i>	<i>v1.0</i>

The function of some services, e.g., local navigation, should be configured for a specific robot to get an optimal behaviour. In this case, the reconnaissance payload was prepared for use on GARM as shown in Figure 8-4.

8.1.3 Control Station “Fernmeldekabine 2B”

The control software of Fraunhofer CMS is a ROS-based software. This consists of the plug-ins for *rqt* and *rviz*. This allows the GUI to be adapted to any robot and application scenario. The ROS/IOP bridge

developed at FKIE was used to also control the IOP-compliant robots. This bridge not only transforms ROS-based robots into IOP-compliant systems, but also provides the ability to use the IOP services and provide them for ROS-based GUI without the need to customize the existing software.

Figure 8-5 shows the *rgt* GUI as used to control the GARM. In addition to the video display for the cameras on the payload, control elements to control the pen-tilt units, the power supply and IOP specific control are visualized.

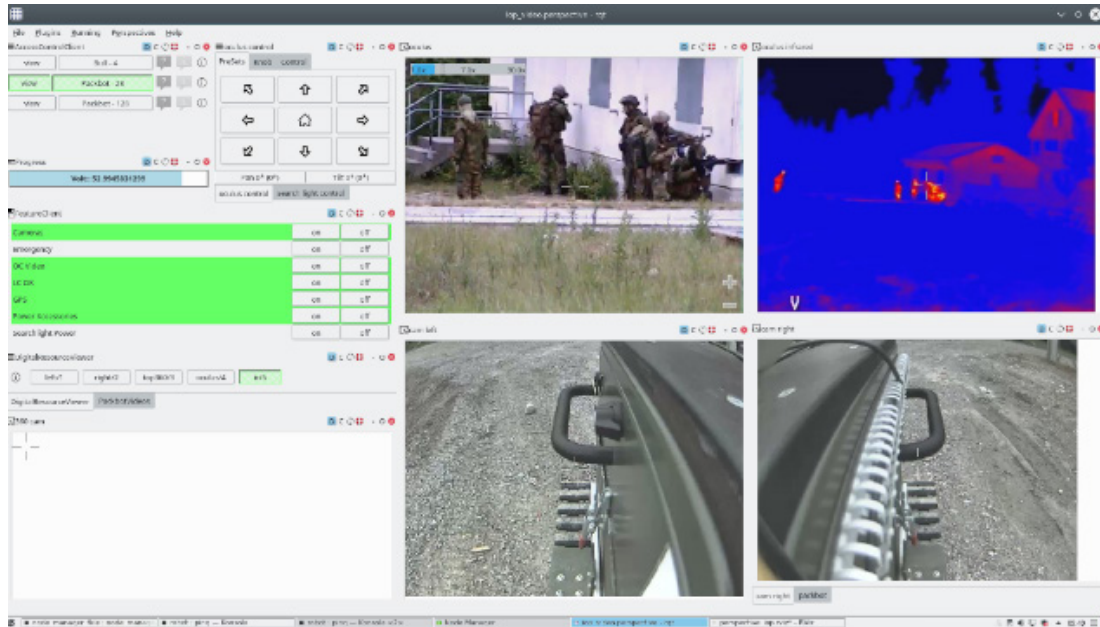


Figure 8-5: Part of Control GUI While Control GARM.

Rviz is used to visualize the map, position and laser data, see Figure 8-6.

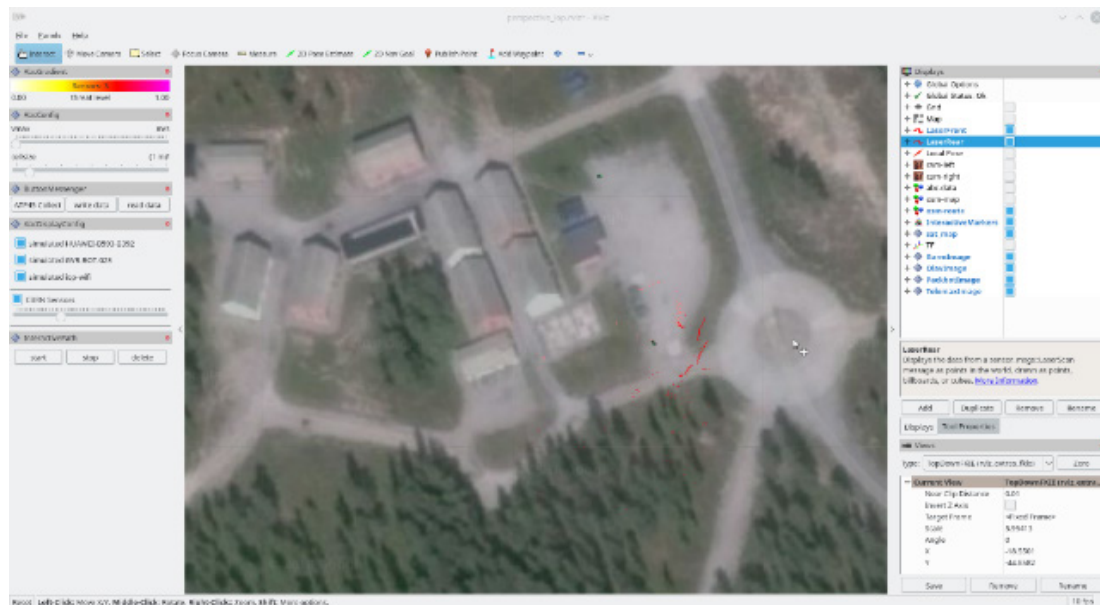


Figure 8-6: Visualization of Robot Position in Rviz.

In the same view, the positions of other IOP-compliant robots were also shown, see Figure 8-7.

For situation overview and control of the robots, a communications booth of type “Fernmeldekabine 2B” was used. This was set up with two control stations, each with two monitors and all necessary infrastructure, including radio modules. The communications booth and the interior view are shown in Figure 8-8 and Figure 8-9.



Figure 8-7: Multi-Robot View.



Figure 8-8: Exterior View of FKIE Control Station.



Figure 8-9: Inside View of FKIE Control Station.

8.2 COMMUNICATION INFRASTRUCTURE

Communication between the systems during the IOP tests was IP-based. The IOP does not prescribe which devices are used for this function. So, each robot system was equipped with its own radio system. The GARM communicates via the Zelos-C, the OLAV via MPU4, Telexmax via Ubiquiti Bullet 5 and the GVR-BOT via Silvus Streamcaster. To enable each OCU to access the robots, all receivers were connected to a common switch, mostly by detours. In order to avoid any address conflicts, the addresses were assigned in advance and a DHCP server was set up. The overview of the most important network participants is shown in Figure 8-10.

The infrastructure for controlling the robots was distributed to the communications booth (Fernmeldekabine) and the container. For space and demonstration reasons, the robots were controlled during the final demonstration from the container. The control system in the communications booth was used exclusively to monitor the systems. In this constellation the scenario was presented without communication difficulties.

In the first days of the integration phase, however, communication problems were frequently observed, which severely impaired the control of the robots in particular. Identification of the problems was very difficult for several reasons:

- **Different Radio Systems.** Each robot system was controlled by its own radio system. In addition, some systems also had separate radio emergency stop. Although not all radio systems shared the same frequency range, the interference between the systems could not be ruled out. For exact statements the appropriate equipment was not available.

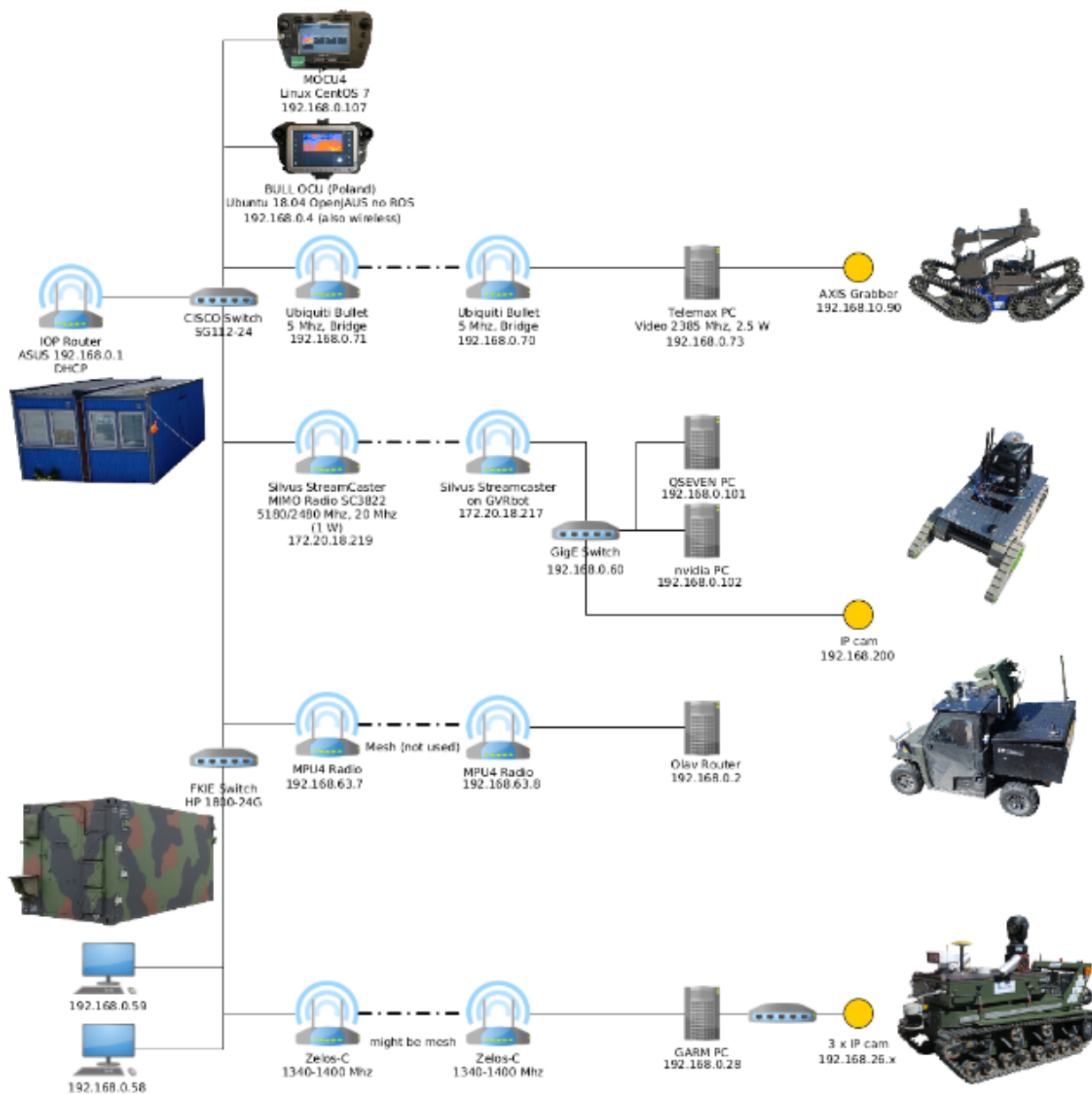


Figure 8-10: Network Topology.

- **Separation of IOP and Video Transmission.** The video transmission is done in IOP via common standards, e.g., Real-Time Streaming Protocol (RTSP). While all other sensor and control data are passed through the “JAUS Node Manager”, the transmission of the audio-visual data (streams) is “parallel” to it. Most current implementations of IOP-compliant software also lack the control of the current stream viewers. So, it can very quickly happen that several OCUs subscribe to the streams of the same robot and thus bring the wireless communication to the robot to overload. Since the video streams are transmitted outside the “JAUS Node Manager”, the prioritization mechanism of the IOP has little impact. As a result, the controlling OCU loses control of the robot.
- **No Monitoring Options for IOP Communication.** The current implementations of the “JAUS Node Manager”, which runs on every host and administrates the IOP communication, offer no possibilities to monitor the current network traffic and thus to provide helpful hints in case of

problems. Although it is not always sufficient to monitor only the IOP traffic. But, providing information about current participants and their activity, especially during development and integration, can provide evidence of problems. This feature can also be helpful in operation. In IOP all subscribers are subscribed to one multicast group. Thus, a subscriber with a misbehaviour that sends “too much” to this group is enough to cause disruption. Appropriate mechanisms to detect such and other disruptive activities at runtime would be very helpful in the “JAUS Node Manager”.

Currently, the tool Wireshark is used to analyse IOP traffic. It is a very powerful tool for recording and analyzing all network traffic on the computer. It has only one major disadvantage; it does not automatically detect the abnormalities and errors in necessary processes of IOP communication. This requires an IOP specialist.

8.3 CONCLUSION

The IOP field trials in Rena can be considered successful for several aspects. They have shown that the robot systems of NATO partners can be used interoperably in a common scenario. The IOP conformity of the robot systems facilitates integration into joint situation presentation. It is also easy to take control of the robot from central control station. Even software-level interoperability is enough to integrate a system into a scenario.

Weaknesses of the IOP were uncovered during the trials, and their improvement can be addressed in future versions of the IOP. These include, but are not limited to: network configuration, which is currently largely manual; network overload detection; and prioritization of IOP control commands.

For FKIE, the IOP field trial in Rena was also very informative for testing the ROS/IOP bridge. With an ROS/IOP bridge a ROS-based robot can provide IOP-compliant functions with very small effort. Since the ROS/IOP bridge was also used on the other systems the adaptation of the bridge to other ROS systems was also tested. This proved to be quite simple and flexible, even to build a common position overview of the IOP systems with a ROS-based GUI called *mapviz* [42] within a very short time.



Chapter 9 – REFERENCES

- [1] Freeman E., Robson E., Sierra K., Bates B. Head First Design Patterns. Hendrickson M., Loukides M., (eds.). O'Reilly Media; 2004.
- [2] Serrano D., Chrobocinski P., De Cubber G., Moore D., Leventakis G., Govindaraj S. ICARUS and DARIUS approaches towards interoperability. In: IARP RISE Workshop; Lisbon, Portugal. IARP; 2015.
- [3] Serrano D. Key initiatives for interoperability through standardization. Applicability to small unmanned vehicles. In: The NATO NATO STO Lecture Series SCI-271, 2015.
- [4] Studer R., Benjamins V.R., Fensel D. Knowledge engineering: Principles and methods. Data and Knowledge Engineering 25: pp. 161-197, 1998.
- [5] Prestes E., Carbonera J.L., Fiorini S.R., Jorge V.A.M., Abel M., Madhavan R., Locoro A., Goncalves P., Barreto M.E., Habib M., Chibani A., Gérard S., Amirat Y., Schlenoff, C. Towards a core ontology for robotics and automation. Robotics and Autonomous Systems 61(11): pp. 1193-1204, 2013. DOI: <http://dx.doi.org/10.1016/j.robot.2013.04.005>.
- [6] NATO: STANAG 4586, Edition No 2, Standard Interfaces of UAV Control System (UCS) for NATO UAV Interoperability [Internet]. Available from: <http://nso.nato.int/nso/zPublic/stanags/current/4586eed02a2.pdf>. Retrieved Nov 1, 2018.
- [7] Birk A., Pfingsthorn M. A HMI supporting adjustable autonomy of rescue robots. In: Bredenfield, A., Jacoff, A., Noda I., Takahashi, Y. (eds.) RoboCup 2005: Robot Soccer World Cup IX. RoboCup 2005. Lecture Notes in Computer Science 4020. Springer, Berlin, 2006.
- [8] Goodrich M., Olsen D., Crandall J.W., Palmer T.J. Experiments in adjustable autonomy. In: IJCAI workshop on autonomy, delegation, and control: Interacting with autonomous agents, 2001 IEEE International Conference on Systems, Man, and Cybernetics, 3, 2001.
- [9] Parasuraman R., Sheridan T., Wickens C. A model for types and levels of human interaction with automation. IEEE Transactions on Systems, Man, and Cybernetics — Part A: Systems and Humans. 30(3), 2000.
- [10] Lacroix S., Alami R., Lemaire T., Hattenberger G., Gancet J. Decision making in multi-UAVs systems: Architecture and algorithms. In: Ollero A., Maza I., (eds.). Multiple Heterogenous Unmanned Aerial Vehicles. Springer Tracts in Advanced Robotics, pp. 15-48, Springer, Berlin, 2007. DOI: 10.1007/978-3-540-73958-6_2.
- [11] Farinelli A., Iocchi L., Nardi D. Multirobot systems: A classification focused on coordination. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 34(5): pp. 2015-2028, 2004.
- [12] ISO. ISO standards website [Internet]. Available from: <http://www.iso.org/iso/home/standards.htm>. Retrieved Nov 1, 2018.
- [13] Pedersen J. Interoperability Standard Analysis (ISA). Standards Committee of the National Defense Industry Association (NDIA) Robotics Division. 2007.
- [14] NATO. STANAG 4586, Edition No 3, Standard Interfaces of UAV Control System (UCS) for NATO UAV Interoperability [Internet]. Available from: <http://nso.nato.int/nso/zPublic/stanags/current/4586eed03.pdf>. Retrieved Nov 1, 2018.

REFERENCES

- [15] Incze M.L., Sideleau S.R., Gagner C., Pippin C.A. Communication and collaboration of heterogeneous unmanned systems using the joint architecture for unmanned systems (JAUS) standards. In: OCEANS 2015, pp. 1-6, Genova, Italy, 2015.
- [16] Rowe S., Wagner C.R. An introduction to the joint architecture for unmanned systems (JAUS). 2008.
- [17] SAE International. JAUS core service set [Internet]. 2008. Available from: <http://standards.sae.org/as5710/>.
- [18] SAE International. JAUS mobility service set [Internet]. Available from: <http://standards.sae.org/as6009/>. Retrieved Nov 1, 2018.
- [19] SAE International. JAUS environment sensing service set [Internet]. Available from: <http://standards.sae.org/as6060/>. Retrieved Nov 1, 2018.
- [20] SAE International. JAUS manipulator service set [Internet]. Available from: <http://standards.sae.org/as6057/>. Retrieved Nov 1, 2018.
- [21] QGroundControl. MAVLink micro air vehicle communication protocol [Internet]. Available from: <http://qgroundcontrol.org/mavlink/start>. Retrieved Nov 1, 2018.
- [22] Balta H., Bedkowski J., Govindaraj S., Majek K., Musialik P., Serrano D., Alexis K., Siegwart R., De Cubber G. Integrated data management for a fleet of search and rescue robots. *Journal of Field Robotics* 34(3), pp. 539-582, 2017.
- [23] Ferri G., Ferreira F., Djapic V., Petillot Y., Franco M.P., Winfield A. The euRathlon 2015 Grand Challenge: The first outdoor multi-domain search and rescue robotics competition – A marine perspective. *Marine Technology Society Journal* 50(4), pp. 81-97, 2016. <https://doi.org/10.4031/MTSJ.50.4.9>.
- [24] RS JPO, UGV Control Interoperability Profile (IOP), 2013.
- [25] Bunker P., Volk A. NATO LCG LE UGV ToE Interoperability Experiences, 2015.
- [26] Website: GitHub.com. fkie/io.core. https://github.com/fkie/iop_core. Retrieved Nov 1, 2018.
- [27] Website: ArduPilot. Mission Planner. <http://ardupilot.org/planner/>. Retrieved Nov 1, 2018.
- [28] Website: Ros.org. Mavros. <http://wiki.ros.org/mavros>. Retrieved Nov 1, 2018.
- [29] Website: GitHub.com. How ROS/IOP-Bridge works – An overview. https://github.com/fkie/iop_core/blob/master/doc/how_it_works.md. Retrieved Nov 1, 2018.
- [30] Website: GitHub.com. EZ-WifiBroadcast. <https://github.com/bortek/EZ-WifiBroadcast>. Retrieved Nov 1, 2018.
- [31] Website: Ros.org. Gscam. <http://wiki.ros.org/gscam>. Retrieved Nov 1, 2018.
- [32] Website: GStreamer. <https://gstreamer.freedesktop.org>. Retrieved Nov 1, 2018.
- [33] Website: Ros.org. SITL Simulator (Software in the Loop). <http://ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html>. Retrieved Nov 1, 2018.

- [34] Mathiassen K., Baksaas, M., Olsen, L.E., Thoresen, M., Tveit, B. Development of an autonomous off-road vehicle for surveillance missions. STO-MP-IST-127: Intelligence and autonomy in robotics. NATO Science and Technology Organization, 2016. ISBN 978-92-837-2068-3.
- [35] Seehuus R.A., Mathiassen K., Ruud E.-L.M., Simonsen A.S., Hermansen F. Battle management language for robotic systems, Experiences from applications on an UGV and an USV, modelling and simulation for autonomous systems (MESAS) 2018 proceedings, Springer International Publishing, 2018.
- [36] Syre Wiig M., Løvlid R.A., Mathiassen K., Krogstad T.R. Decision autonomy for unmanned vehicles. STO-MP-IST-127: Intelligence and autonomy in robotics. NATO Science and Technology Organization. 2016. ISBN 978-92-837-2068-3.
- [37] Haavardsholm T.V., Smestad R., Larsen, M.V., Thoresen M., Dyrdal I. Scene Understanding for Autonomous Steering. STO-MP-IST-127: Intelligence and Autonomy in Robotics. NATO Science and Technology Organization 2016 ISBN 978-92-837-2068-3.
- [38] Website: Github. Fraunhofer ROS/IOP bridge, https://github.com/fkie/iop_core. Retrieved Nov 1, 2018.
- [39] Website: Gstreamer. Open source multimedia framework <https://gstreamer.freedesktop.org/>. Retrieved Nov 1, 2018.
- [40] Website: Gstreamer. appsrc <https://gstreamer.freedesktop.org/data/doc/gstreamer/head/gst-plugins-base-plugins/html/gst-plugins-base-plugins-appsrc.html>. Retrieved Nov 1, 2018.
- [41] Website: Github. Gstreamer RTSP server example <https://github.com/GStreamer/gst-rtsp-server/blob/master/examples/test-launch.c>. Retrieved Nov 1, 2018.
- [42] Website: ROS Package Summary Mapviz. <http://wiki.ros.org/mapviz>. Retrieved Nov 1, 2018.
- [43] Website: Github. Fraunhofer ROS/IOP bridge iop_cfg_sim_stage https://github.com/fkie/iop_cfg_sim_stage_fkie/blob/master/launch/inc_iop_control.launch. Retrieved Nov 1, 2018.

REFERENCES



REPORT DOCUMENTATION PAGE			
1. Recipient's Reference	2. Originator's References STO-TR-IST-149 AC/323(IST-149)TP/958	3. Further Reference ISBN 978-92-837-2278-2	4. Security Classification of Document PUBLIC RELEASE
5. Originator Science and Technology Organization North Atlantic Treaty Organization BP 25, F-92201 Neuilly-sur-Seine Cedex, France			
6. Title Capability Concept Demonstrator for Interoperability within Unmanned Ground Systems and C2			
7. Presented at/Sponsored by Final report of NATO RTG IST-149.			
8. Author(s)/Editor(s) F. Schneider, K. Mathiassen, P. Bounker, J. Glówka, A. Tiderko, M. Baksaas, R. Kozik, J.J. Röning and G. De Cubber			9. Date June 2021
10. Author's/Editor's Address Multiple			11. Pages 114
12. Distribution Statement There are no restrictions on the distribution of this document. Information about the availability of this and other STO unclassified publications is given on the back cover.			
13. Keywords/Descriptors Interoperability; Interoperability Profiles (IOP); Joint Architecture for Unmanned Systems (JAUS); Unmanned Ground Systems (UGS); Unmanned Ground Vehicles (UGV)			
14. Abstract This report describes the research and experiment efforts of the NATO STO group IST-149-RTG Capability Concept Demonstrator for Interoperability within Unmanned Ground Systems and C2. Unmanned Ground Vehicles (UGVs) are becoming increasingly relevant in the modern battle-space. The main purpose of the group was to investigate possible standards for controlling UGVs from Operator Control Units (OCUs) and receiving data back, and tests them in a real world scenario. The efforts in the project have been two folded. The Belgium contribution is work done in the EU project ICARUS. The second effort was a combined effort in the group to have an experiment demonstrating interoperability between the UGVs and OCUs available within the group. Both efforts used the Joint Architecture for Unmanned Systems (JAUS) with the Interoperability Profile (IOP) to successfully enable interoperability between the systems. The trials showed that it is possible to extend the systems quite easily and achieve compliance with parts of the standard in a relatively short time. Both Fraunhofer FKIE and TARDEC had developed software to pass information from the IOP domain to the Robotic Operating System (ROS). As this was a successful trial, the next step would be to test interoperability using higher level functions.			





BP 25

F-92201 NEUILLY-SUR-SEINE CEDEX • FRANCE
Télécopie 0(1)55.61.22.99 • E-mail mailbox@cs.o.nato.int



DIFFUSION DES PUBLICATIONS STO NON CLASSIFIEES

Les publications de l'AGARD, de la RTO et de la STO peuvent parfois être obtenues auprès des centres nationaux de distribution indiqués ci-dessous. Si vous souhaitez recevoir toutes les publications de la STO, ou simplement celles qui concernent certains Panels, vous pouvez demander d'être inclus soit à titre personnel, soit au nom de votre organisation, sur la liste d'envoi.

Les publications de la STO, de la RTO et de l'AGARD sont également en vente auprès des agences de vente indiquées ci-dessous.

Les demandes de documents STO, RTO ou AGARD doivent comporter la dénomination « STO », « RTO » ou « AGARD » selon le cas, suivi du numéro de série. Des informations analogues, telles que le titre et la date de publication sont souhaitables.

Si vous souhaitez recevoir une notification électronique de la disponibilité des rapports de la STO au fur et à mesure de leur publication, vous pouvez consulter notre site Web (<http://www.sto.nato.int/>) et vous abonner à ce service.

CENTRES DE DIFFUSION NATIONAUX

ALLEMAGNE

Streitkräfteamt / Abteilung III
Fachinformationszentrum der Bundeswehr (FIZBw)
Gorch-Fock-Straße 7, D-53229 Bonn

BELGIQUE

Royal High Institute for Defence – KHID/IRSD/RHID
Management of Scientific & Technological Research
for Defence, National STO Coordinator
Royal Military Academy – Campus Renaissance
Renaissancelaan 30, 1000 Bruxelles

BULGARIE

Ministry of Defence
Defence Institute "Prof. Tsvetan Lazarov"
"Tsvetan Lazarov" bul no.2
1592 Sofia

CANADA

DGSIST 2
Recherche et développement pour la défense Canada
60 Moodie Drive (7N-1-F20)
Ottawa, Ontario K1A 0K2

DANEMARK

Danish Acquisition and Logistics Organization
(DALO)
Lautrupbjerg 1-5
2750 Ballerup

ESPAGNE

Área de Cooperación Internacional en I+D
SDGPLATIN (DGAM)
C/ Arturo Soria 289
28033 Madrid

ESTONIE

Estonian National Defence College
Centre for Applied Research
Riia str 12
Tartu 51013

ETATS-UNIS

Defense Technical Information Center
8725 John J. Kingman Road
Fort Belvoir, VA 22060-6218

FRANCE

O.N.E.R.A. (ISP)
29, Avenue de la Division Leclerc
BP 72
92322 Châtillon Cedex

GRECE (Correspondant)

Defence Industry & Research General
Directorate, Research Directorate
Fakinos Base Camp, S.T.G. 1020
Holargos, Athens

HONGRIE

Hungarian Ministry of Defence
Development and Logistics Agency
P.O.B. 25
H-1885 Budapest

ITALIE

Ten Col Renato NARO
Capo servizio Gestione della Conoscenza
F. Baracca Military Airport "Comparto A"
Via di Centocelle, 301
00175, Rome

LUXEMBOURG

Voir Belgique

NORVEGE

Norwegian Defence Research
Establishment
Attn: Biblioteket
P.O. Box 25
NO-2007 Kjeller

PAYS-BAS

Royal Netherlands Military
Academy Library
P.O. Box 90.002
4800 PA Breda

POLOGNE

Centralna Biblioteka Wojskowa
ul. Ostrobramska 109
04-041 Warszawa

PORTUGAL

Estado Maior da Força Aérea
SDFA – Centro de Documentação
Alfragide
P-2720 Amadora

REPUBLIQUE TCHEQUE

Vojenský technický ústav s.p.
CZ Distribution Information Centre
Mladoboleslavská 944
PO Box 18
197 06 Praha 9

ROUMANIE

Romanian National Distribution
Centre
Armaments Department
9-11, Drumul Taberei Street
Sector 6
061353 Bucharest

ROYAUME-UNI

Dstl Records Centre
Rm G02, ISAT F, Building 5
Dstl Porton Down
Salisbury SP4 0JQ

SLOVAQUIE

Akadémia ozbrojených síl gen.
M.R. Štefánika, Distribučné a
informačné stredisko STO
Demänová 393
031 01 Liptovský Mikuláš 1

SLOVENIE

Ministry of Defence
Central Registry for EU & NATO
Vojkova 55
1000 Ljubljana

TURQUIE

Milli Savunma Bakanlığı (MSB)
ARGE ve Teknoloji Dairesi
Başkanlığı
06650 Bakanlıklar – Ankara

AGENCES DE VENTE

The British Library Document
Supply Centre
Boston Spa, Wetherby
West Yorkshire LS23 7BQ
ROYAUME-UNI

Canada Institute for Scientific and
Technical Information (CISTI)
National Research Council Acquisitions
Montreal Road, Building M-55
Ottawa, Ontario K1A 0S2
CANADA

Les demandes de documents STO, RTO ou AGARD doivent comporter la dénomination « STO », « RTO » ou « AGARD » selon le cas, suivie du numéro de série (par exemple AGARD-AG-315). Des informations analogues, telles que le titre et la date de publication sont souhaitables. Des références bibliographiques complètes ainsi que des résumés des publications STO, RTO et AGARD figurent dans le « NTIS Publications Database » (<http://www.ntis.gov>).



BP 25
F-92201 NEUILLY-SUR-SEINE CEDEX • FRANCE
Télécopie 0(1)55.61.22.99 • E-mail mailbox@cs.o.nato.int



DISTRIBUTION OF UNCLASSIFIED STO PUBLICATIONS

AGARD, RTO & STO publications are sometimes available from the National Distribution Centres listed below. If you wish to receive all STO reports, or just those relating to one or more specific STO Panels, they may be willing to include you (or your Organisation) in their distribution.

STO, RTO and AGARD reports may also be purchased from the Sales Agencies listed below.

Requests for STO, RTO or AGARD documents should include the word 'STO', 'RTO' or 'AGARD', as appropriate, followed by the serial number. Collateral information such as title and publication date is desirable.

If you wish to receive electronic notification of STO reports as they are published, please visit our website (<http://www.sto.nato.int/>) from where you can register for this service.

NATIONAL DISTRIBUTION CENTRES

BELGIUM

Royal High Institute for Defence –
KHID/IRSD/RHID
Management of Scientific & Technological
Research for Defence, National STO
Coordinator
Royal Military Academy – Campus
Renaissance
Renaissancelaan 30
1000 Brussels

BULGARIA

Ministry of Defence
Defence Institute "Prof. Tsvetan Lazarov"
"Tsvetan Lazarov" bul no.2
1592 Sofia

CANADA

DSTKIM 2
Defence Research and Development Canada
60 Moodie Drive (7N-1-F20)
Ottawa, Ontario K1A 0K2

CZECH REPUBLIC

Vojenský technický ústav s.p.
CZ Distribution Information Centre
Mladoboleslavská 944
PO Box 18
197 06 Praha 9

DENMARK

Danish Acquisition and Logistics Organization
(DALO)
Lautrupbjerg 1-5
2750 Ballerup

ESTONIA

Estonian National Defence College
Centre for Applied Research
Riia str 12
Tartu 51013

FRANCE

O.N.E.R.A. (ISP)
29, Avenue de la Division Leclerc – BP 72
92322 Châtillon Cedex

GERMANY

Streitkräfteamt / Abteilung III
Fachinformationszentrum der
Bundeswehr (FIZBw)
Gorch-Fock-Straße 7
D-53229 Bonn

GREECE (Point of Contact)

Defence Industry & Research General
Directorate, Research Directorate
Fakinos Base Camp, S.T.G. 1020
Holargos, Athens

HUNGARY

Hungarian Ministry of Defence
Development and Logistics Agency
P.O.B. 25
H-1885 Budapest

ITALY

Ten Col Renato NARO
Capo servizio Gestione della Conoscenza
F. Baracca Military Airport "Comparto A"
Via di Centocelle, 301
00175, Rome

LUXEMBOURG

See Belgium

NETHERLANDS

Royal Netherlands Military
Academy Library
P.O. Box 90.002
4800 PA Breda

NORWAY

Norwegian Defence Research
Establishment, Attn: Biblioteket
P.O. Box 25
NO-2007 Kjeller

POLAND

Centralna Biblioteka Wojskowa
ul. Ostrobramska 109
04-041 Warszawa

PORTUGAL

Estado Maior da Força Aérea
SDFA – Centro de Documentação
Alfragide
P-2720 Amadora

ROMANIA

Romanian National Distribution Centre
Armaments Department
9-11, Drumul Taberei Street
Sector 6
061353 Bucharest

SLOVAKIA

Akadémia ozbrojených síl gen
M.R. Štefánika, Distribučné a
informačné stredisko STO
Demänová 393
031 01 Liptovský Mikuláš 1

SLOVENIA

Ministry of Defence
Central Registry for EU & NATO
Vojkova 55
1000 Ljubljana

SPAIN

Área de Cooperación Internacional en I+D
SDGPLATIN (DGAM)
C/ Arturo Soria 289
28033 Madrid

TURKEY

Milli Savunma Bakanlığı (MSB)
ARGE ve Teknoloji Dairesi Başkanlığı
06650 Bakanlıklar – Ankara

UNITED KINGDOM

Dstl Records Centre
Rm G02, ISAT F, Building 5
Dstl Porton Down, Salisbury SP4 0JQ

UNITED STATES

Defense Technical Information Center
8725 John J. Kingman Road
Fort Belvoir, VA 22060-6218

SALES AGENCIES

**The British Library Document
Supply Centre**
Boston Spa, Wetherby
West Yorkshire LS23 7BQ
UNITED KINGDOM

**Canada Institute for Scientific and
Technical Information (CISTI)**
National Research Council Acquisitions
Montreal Road, Building M-55
Ottawa, Ontario K1A 0S2
CANADA

Requests for STO, RTO or AGARD documents should include the word 'STO', 'RTO' or 'AGARD', as appropriate, followed by the serial number (for example AGARD-AG-315). Collateral information such as title and publication date is desirable. Full bibliographical references and abstracts of STO, RTO and AGARD publications are given in "NTIS Publications Database" (<http://www.ntis.gov>).