ADS-B Based Separation Support for General Aviation

Damiano Taurino¹², Giuseppe Frau¹, Valentina Beato¹ and Alessandra Tedeschi¹

¹Deep Blue Consulting&Research
Piazza Buenos Aires, 20
Rome, 00198 Italy
+39-6-8555208
{damiano.taurino, giuseppe.frau, valentina.beato, alessandra.tedeschi}@dblue.it

²Engineering Department
University of Florence
Via Santa Marta, 3
Florence, 50139 Italy

ABSTRACT
In this paper we present a study for the development of a new tool to support separation for General Aviation based on ADS-B technology. The tool aims at developing a device to support General Aviation pilots in the process of conflict detection and resolution as well as in the flight routing. The introduction of such a tool, based on ADS-B technology and a Game Theory collaborative algorithm, represents a new approach in ensuring high performance levels in GA.

Keywords
General Aviation, VFR, Conflict Detection and Resolution, Separation Assurance, ADS-B, Satisficing Game Theory.

INTRODUCTION
Recent statistics [1] have shown that the civil air traffic is undertaking a constant growth since the last years. This involves also General Aviation (GA) and traffic from small and light aircrafts.

With GA one refers to all but commercial and military aviation, and thus business aviation, touristic non-scheduled flights, private flights and air schools flights, so including a wide range of aircraft types that fly under very different conditions and rules, both IFR (Instrumental Flight Rules) and VFR (Visual Flight Rules).

This category/class is considered of strategic relevance for the achievement of the mobility and capacity standards foreseen for the year 2020 by EUROCONTROL itself [2]. Indeed to increment the flight coverage throughout Europe and let also small destinations be connected, GA, and business aviation in particular, are the perfect candidates.

Interesting data comes from statistical analysis for business aviation (BA). For this category of flights it has been shown [3] that for the year 2009 the half of the yearly traffic took place among airports with less than 50 flights per day. This result highlights the importance of an increase of the coordination capacity in high traffic conditions also for small airport facilities and thus also for not controlled airspaces.

In this scenario it is crucial to set the basis for an effective exploitation of GA capabilities and thus to ensure for this category the same safety standards as for scheduled flights. R&D activity should be dedicated to develop new technologies and tools to reduce the piloting difficulties and complexity, to reduce in this way the pilot’s workload, and simultaneously to enhance the pilot’s situation awareness. This task is even more challenging when considering that GA pilots are usually less trained than commercial aviation pilots.

The SeparA consortium (see acknowledgment section) suggests a new contribution to address some of the safety and efficiency issues in GA. It indeed aims at developing a device able to deliver to the pilot real time information about surrounding traffic, restricted areas and to suggest routes. This paper presents the details of the tool designed in the SeparA project and the rationale behind the technical and technological choices.

In the first section of the paper we will briefly outline the key issues of the project, in the second section the device architecture will be described, the third section will be devoted to the algorithm implemented by the SeparA project, its customizations and some experimental results. The fourth and the fifth section will describe the Human Machine Interface (HMI) issues and its design process, while the sixth section will present the validation and testing sessions. Then some conclusions are presented.

THE SEPARA PROJECT
The project aims at developing a device to support pilots of GA in the process of conflict detection and resolution while ensuring an efficient flight routing. The operative scenario envisaged in the SeparA project considers different traffic conditions in a non-controlled air zone with different categories of GA flights flying according to Visual Flight Rules (VFR), all equipped with an ADS-B device. During a VFR flight the pilot is uniquely responsible for separation from other traffic and for avoiding restricted areas or forbidden airspaces infringement. In the preliminary phases of requirements collection comments about this critical tasks were
collected from GA pilots. It emerged that visual separation from traffic and forbidden airspace may be a major issue in a future scenario of huge traffic increasing. Under the described conditions (VFR flights and ADS-B equipped aircraft) the SepA project will deliver the pilot a tool for enhancing his situation awareness and to support the decision making process during the piloting manoeuvres. The SepA device indeed will provide the pilot a view of the surrounding airspace signalling the presence of other aircraft and potentially conflicting ones within the timeframe of 3 to 15 minutes. The SepA device will suggest manoeuvres computed by an algorithm derived from the Game Theory (see following sections for details).

THE SYSTEM ARCHITECTURE

The tool designed within the SepA project is based on the Automatic Dependent Surveillance Broadcast (ADS-B, see [4]). As envisioned by EUROCONTROL in the Cascade Programme (see [5]), in the future of ATM the mode-S technology, and in particular ADS-B, will be mandatory for all flights in the European sky. The presence of an ADS-B transceiver as communication module will allow the broadcasting and the collecting of flight information (flight ID, position, speed, heading and intent) to and from surrounding aircrafts, see Figure 1. The communication module delivers data to aircraft control unit and dedicated SepA CPU. The latter receives data about the aircraft itself (coming from onboard sensors) from the aircraft control unit. In the SepA CPU unit an ad-hoc developed algorithm elaborates the traffic data and delivers a “satisfying route” in the philosophy of a fully collaborative behavior from all the involved aircraft. The route is updated constantly according to the time evolution of surrounding traffic and thus displayed and continuously refreshed on visual interface.

Figure 1: Sketch of the system architecture of the tool developed by the SepA project.

THE ALGORITHM

Separation is assured by a dedicated innovative algorithm. This solution is inspired by the emerging framework of Collective Intelligence (COIN, see[6]). This formal framework is designed to address situations where there is no centralized control, where there is a clear global objective function to be optimized and where the problem can be modelled by means of a collection of agents. A global and distributed solution that relies on several independent agents, provides many advantages: the risk of catastrophic failures for the system is largely subdued. In a centralized approach, a single failure occurring in the central computation unit might “leave in the dark” the whole system. On the contrary, the intrinsically distributed and adaptive nature of Collective Intelligence would allow for a higher degree of robustness and dependability for the whole system. The failure of a single node does not entail the failure of the whole system, and the various agents can more or less quickly adapt to the new configuration. Distributed computing also could allow for more efficient and performing computation of solutions to the optimisation problem. Although the solution provided might be only suboptimal, it seems to outperform centralised approaches, because achieved with cheap computational resources for the single node and available in real time.

Our algorithm is based on a recent technique, proposed in [7]: Satisfying Game Theory (SGT). SGT key feature is that the algorithm seeks an “adequate” solution rather than the optimal one that, in complex problems like Separation Assurance in Air Traffic Control, may be not well defined. Basically, SGT defines the objective by means of two utility functions: selectability and rejectability, which respectively represent benefits and costs for each agent (in our case each aircraft) in making a choice. There might exist dependence between utility functions of different agents that implements an “altruistic” behaviour. This approach has already proven to be effective in dealing with several real-world problems (e.g. packet routing[10], transport logistics[11], automated car driving [7] Airborne Self-Separation for commercial aircrafts [12]).

Furthermore the SGT algorithm was successfully customized and implemented for UAVs environment within the ARCA project, funded by EUROSTARS Programme of the European Commission [13]. In the ARCA project the SGT algorithm is used to enhance Unmanned Aircraft Vehicles (UAV) capability to take autonomous decisions, by providing a fully distributed, robust and efficient solution to the technological challenge of Collision Detection and Resolution. This solution was designed to achieve safe separation between UAVs themselves in segregated areas and between UAVs and commercial aircraft in non segregated airspaces.

In the case of separation assurance problem for GA each aircraft is considered as an agent which acts according to the logic represented in the following flow chart.

The algorithm is performed with a fixed frequency (it may be 2 times each second that is ADS-B frequency). At each time step the algorithm chooses a satisfying heading direction in terms of separation assurance and performance optimization. The direction can be selected from a fixed set.
Figure 2: Flow chart representation of SGT Algorithm.

Required input values are current position, destination, actual heading, flight time, delay and intent of each viewable aircraft including the current concerned aircraft.

First step of the algorithm is to rank viewable agents (including itself). This can be done according to several methods; in our case we assigned higher rank to aircraft closer to destination or having longer flight time and delay. A special case, however, must be taken into account for the non-collaborative agents, in our case represented by commercial airplanes: those agents are assigned to the highest rank.

At this point the algorithm builds the so-called aircraft's "Priority Set" composed by the agents who have higher priority than the concerned one.

Three tasks that can be done simultaneously are performed now.

Rejectability of all the possible options is computed in order to evaluate the risk of collision for each particular direction. This is achieved computing the closest point of approach of all the agents in the Priority Set. Shorter times to the closest point of approach will lead to higher rejectability of that particular option.

Base Selectability is computed for each possible direction available for the aircraft. This part of the utility function reflects the willing to go straight to the destination so directions who lead more quickly to the destination will have higher Base Selectability.

Parent Selectability is computed taking into account the selectability of the higher ranked agents: this function implements the collaborative behaviour of the aircraft as it uses Priority Set aircraft intent in order to allow higher ranked agents to follow a more convenient trajectory.

The last steps are the computation of the convex combination of Base Selectability and Parent Selectability for each direction and the searching for the option that maximizes the difference between selectability and rejectability.

The mathematical details of the algorithm and some simulation results are described and presented in [8].

To further explore the capabilities of the satisficing algorithms, in [9] was investigated the effects of including aircraft that do not follow behavioral rules of the SGT algorithm. These non-compliant aircraft are assumed to transmit the same information as other aircraft, but they never defer to other aircraft and fly directly to their destinations, such as commercial aircraft in our case. Our algorithm offers a natural method of dealing with non-compliant aircraft, i.e., other aircraft move them to the top of their priority set after observing their failure to defer to others. When non-compliant aircraft are included in the scenario, the overall system performance is reduced. The non-compliant aircraft themselves have perfect individual efficiencies, but conforming aircraft are forced to make more deviations from their preferred paths, and conflicts are resolved without the full cooperation of all participants.

**SGT customizations for general aviation**

Several important considerations have to be taken into account in order to customize the SGT approach within the General Aviation environment.

The most important enhancement provided to SGT algorithm is its generalization to the 3D case. We analyzed a similar issue in ARCA project for UAVs [13] but in that case aircraft can change freely their altitude during the flight, i.e. they are not required to maintain Flight Levels, so a simple modification in the algorithm is needed: two directional options are added (up and down) and SGT manage them like the others during route calculation and decision making.

In the General Aviation case aircraft have to maintain Flight Levels, which are standard altitude used to manage air traffic. In this case each aircraft, at each time can decide to remain in its Flight Level or change Level.

In this scenario each Aircraft performs 2D SGT calculation for traffic flying in its Flight Level. If traffic condition is particularly difficult (namely, difference between selectability and rejectability is below a threshold) aircraft can climb or descend one Flight Level.

Figure 3: Decisional flow of SGT for level change
Level changing could lead to a loss of separation in the new level so it is performed only if it is considered a safe manoeuvre.

The aircraft calculates possible future positions of all the neighbours (considering the 3D neighbourhood as a sphere), basing on: time needed by itself to perform level changing; actual position, speed and directional options of other agents. If there is not any risk of collisions with other aircraft level changing is considered safe (see figure 3). Level changing is decided in a very conservative way because algorithm exploit all the possible configurations before performing it.

In figure 4 an example of level change is shown. In a) the aircraft compute SGT in 2D for their flight level. Being the utility function below a given threshold for each directional option (no one is enough convenient), in b) aircraft evaluate change level options. The change represented in b) is not safe because can lead to a missed separation (in red the separation area). In c) upper UAV decide to go straight (i.e. the best option in 2D evaluation) since go up to another level is unsafe too. The lower one choose to go down to a safe level.

Some other enabling key features have been detected and are under development at the moment:

- definition and implementation of restricted airspaces, bad weather areas and/or VFR forbidden airspaces;
- introduction of a list of waypoints to be followed within the SGT solution;
- customization for physical characteristics of different General Aviation aircraft models (cruise speed, curvature radius, vertical motion speed).

**Algorithm performances**

In order to evaluate algorithm performances in terms of separation assurance and global optimization some computer based simulations were performed considering some critical scenarios.

In these simulations, the 2D case is analyzed in depth, deliberately forcing high aircraft densities at the same altitude and deactivating the level change option in order to stress the algorithm by increasing the complexity of interaction between the involved traffic.

The here presented target scenario is the so-called ‘Choke Point’: a predetermined number of vehicles is distributed around a circle and each vehicle has its destination at the exact opposite side of the circle. This scenario represents a well known challenge for any conflict resolution algorithm. The second considered scenario is the ‘Random Flight’ scenario where airplanes appear at random points on a square. They are assigned a random destination point on the opposite side of the square itself.

In analyzing results, we consider different performance measures: the most important is the absence of missed separations, called Separation Assurance. We compute the number of missed separations with varying traffic densities and in different traffic scenarios.

We also take into account the System Efficiency which is defined as the degree to which an aircraft is able to follow its ideal flight path. In order to compute it for the whole system, we perform the overall average

$$SE = \frac{1}{N} \sum_{i=1}^{N} \frac{t_i}{t_i + t_{di}}$$

where $t_i$ is the ideal flight time for aircraft i and $t_{di}$ is added delay time.

In the following table algorithm performance based on 30 measurements for the Choke Point scenario are presented:

<table>
<thead>
<tr>
<th>Aircraft number</th>
<th>Missed Separations</th>
<th>System Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Std. Dev.</td>
</tr>
<tr>
<td>3</td>
<td>0,000</td>
<td>0,000</td>
</tr>
<tr>
<td>5</td>
<td>0,000</td>
<td>0,000</td>
</tr>
<tr>
<td>10</td>
<td>0,000</td>
<td>0,000</td>
</tr>
<tr>
<td>15</td>
<td>0,833</td>
<td>0,278</td>
</tr>
</tbody>
</table>

**Table 1**: SGT performances in ‘Choke Point’ scenario (30 measurements, circle radius = 60 nautical miles, speed = 120 knots, directional option = [−10°, 5°, 0°, 5°, 10°])

---

*Figure 4: An example of SGT for level change*
SGT is a deterministic algorithm, so the different measurements are obtained by changing the initial distribution of priority values. The considered aircraft’s performances are typical for small General Aviation aircraft and are here assumed as significant for the domain.

We compared data here presented to the ones collected during experimentations for the UAV domain in [13]. Our results score better in terms of Separation Assurance, providing a missed separations free solution up to 10 aircraft. This better performance can be understood considering the lower speed and higher manoeuvrability here assumed for the involved aircraft. It is important to remark that the Choke Point scenario is quite far from a real operational scenario: it is an artificial configuration expressly designed to stress the algorithm to analyze its behavior in very critical conditions.

In order to test the level changing functionality, that is the capability to safely change altitude when certain traffic conditions are met (see previous section), some tests were performed. We designed traffic scenarios in which difference between selectability and rejectability is below a threshold for a specific aircraft and we observed through all simulations that:

- if level changing is considered safe, i.e. there is no risk of collisions with other aircraft, the aircraft changes flight level;
- otherwise it chooses the best 2D option, remaining at the same flight level.

This behaviour assures that the introduction of level changing does not affect Separation Assurance.

Some experimentations are under progress at the moment about System Efficiency or other reasonable performance metrics, in order to evaluate overall routes optimization in the 3D case.

**HMI**

An important factor to be taken into account for the success and usefulness of a decision support system for separation assurance in General Aviation environment is the quality of its Human Computer Interface and its integration within a real glass cockpit. New generation General Aviation cockpits are really sophisticated and performances are typical for small General Aviation aircraft and are here assumed as significant for the domain.

We compared data here presented to the ones collected during experimentations for the UAV domain in [13]. Our results score better in terms of Separation Assurance, providing a missed separations free solution up to 10 aircraft. This better performance can be understood considering the lower speed and higher manoeuvrability here assumed for the involved aircraft. It is important to remark that the Choke Point scenario is quite far from a real operational scenario: it is an artificial configuration expressly designed to stress the algorithm to analyze its behavior in very critical conditions.

In order to test the level changing functionality, that is the capability to safely change altitude when certain traffic conditions are met (see previous section), some tests were performed. We designed traffic scenarios in which difference between selectability and rejectability is below a threshold for a specific aircraft and we observed through all simulations that:

- if level changing is considered safe, i.e. there is no risk of collisions with other aircraft, the aircraft changes flight level;
- otherwise it chooses the best 2D option, remaining at the same flight level.

This behaviour assures that the introduction of level changing does not affect Separation Assurance.

Some experimentations are under progress at the moment about System Efficiency or other reasonable performance metrics, in order to evaluate overall routes optimization in the 3D case.

**HMI**

An important factor to be taken into account for the success and usefulness of a decision support system for separation assurance in General Aviation environment is the quality of its Human Computer Interface and its integration within a real glass cockpit. New generation General Aviation cockpits are really sophisticated and integrated systems, with several instruments and displays, designed to provide pilot(s) with a large number of information to support decision making processes during all the flight phases. Integrating new tools in such a complex and critical system requires a deep understanding of the mutual interactions occurring between human, the existent system and new technologies. In order to achieve a full and satisfactory integration we adopt User Centred Design approach to design the interface. Real General Aviation pilots are involved in the design process and their considerations, observations and needs will be analyzed through several methods (interviews, questionnaires) and indicators (workload, situational awareness) collected during simulations and tests performed in a realistic environment.

The information provided by the separation tool consists of the surrounding traffic, the precluded areas and the suggested manoeuvres to avoid both of them. At the moment two possible architectural solutions are under exam:

- integration of the tool as a new information layer within existent cockpit displays;
- a new dedicated portable device with its own display to be placed in the cockpit.

Both these solutions present pros and cons. The first one assures a complete physical integration of the new tool within the existent cockpit, the pilot is used to interact with the HMI and all the information are concentrated in a single (or two, depending on cockpit models) display. On the other side, cockpit systems are often “closed” ones and, because of manufacturer policies, it is not simple to integrate new third part functionalities within this kind of systems. Furthermore information provided by current glass cockpit is very rich and structured and it could be difficult to add a new layer.

The second solution allows the user to bring his own standalone device onboard. In this case information presentation can be clearer since the graphical environment is specifically intended for the application. In this case, however, a new display has to be added to the ones onboard, in an already tight space. The new display can be also awkward or a dangerous source of distraction for the pilot.

These solutions are under evaluation at the moment in order to select the most adequate. The aim of the project is to develop the chosen solution taking into account the pros and cons referred in this paragraph and the new considerations those will appear during the development phase. Nevertheless one of the aims of SeparA project is to provide a modular architecture easy to be integrated within existent onboard systems. In particular the component called “CPU with SGT Algorithm” in Figure 1 is completely independent from the HMI chosen for information presentation and it can be reused in different platform configurations.

**SYSTEM DESIGN**

The SeparA Decision Support System architecture and HMI has been designed following an iterative user-centred approach that highlights critical interactions among the humans, the technical components and the environment [14][15].

GA Pilots and Instructors has been actively involved from early phases of system design and development in order to:

1. Specify the Context of Use
   Identify different target users, describe their activities, and the conditions under which they will use the system.
2. Define Scenarios of Use
   Define relevant realistic narrative Scenarios of usage. Identify user detailed activities and tasks for each Scenario. Highlight potential critical interactions among all the socio-technical system components.
3. Specify user needs and system requirements
   Identify any operational and business requirements or user goals that must be met for the product to be successful.
4. Support the creation of design solutions
This part of the process is done in progressive stages, building from a rough concept to a complete design, with the involvement of Interaction Designers and Human Factors experts, Technology providers and end-users.

5. Evaluate designs
   Iterative evaluations - through experts walktrough, rapid prototyping sessions and usability testing with actual users – to refine the design and development processes.

TESTS AND VALIDATION
We developed an appropriate test and validation plan through the different development phases by setting up (and eventually iterating) different simulation stages:

- computer simulations used to obtain numerical results in order to estimate separation assurance and efficiency have been performed. This phase has actually been finished with satisfactory results. These results will be presented in the final paper.
- Hardware in the loop simulations will be performed by integrating the real hardware within a complex flight simulator in order to evaluate solution performances in a fully simulated aircraft environment.
- Human in the loop simulations will be performed in the flight simulator environment by involving General Aviation pilots in tools evaluation in terms of usability, situation awareness enhancing, quality of information presentation and other factors.

Moreover, the SeparA project is taking into account from the very beginning Safety, Standardization and Certification issues. A preliminary Safety Assessment of the SeparA device and concept has been carried out. Potential hazards and system faults have been identified and analysed. Proper mitigation means have been proposed from a technological, operational and procedural perspective. The proposed mitigation actions will be evaluated and refined in collaboration with Safety, Human Factors and domain experts, informing further system development. System compliance with regulation, standards and the SESAR Operational Concepts will be always analysed and assessed.

CONCLUSION
The General Aviation market is expected to perform a substantial growth in the next years. The tool provided by SeparA project aims to support GA pilots in their decision making process during all the flight phases, by providing information about surrounding aircraft and suggesting safe and efficient trajectories to maintain separation from them. This feature may be particularly useful and appreciated when flying in VFR mode in airspaces where separation for this kind of flight is not provided by Air Traffic Control. Based on ADS-B technology, the SeparA device aims to provide a valuable and affordable instrument to be deployed onboard of all General Aviation aircraft in the next years.

ACKNOWLEDGMENT
This work has been developed within the Sistema per il supporto alla separazione per l'aviazione generale (SeparA) project, funded by Italian agency Filas (Financial Investment Agency of the Regione Lazio).

REFERENCES
ABSTRACT
The SESAR System Wide Information Management (SWIM) concept proposes a global information management system to support all the data sharing among disparate and distributed systems of the Air Traffic Management (ATM) network. The overall ATM system functionality will be realized through the interaction of several ATM services (business services), ultimately supported by lower-level information management services provided by SWIM.

SWIM will consist on the set of information management services similar to the services provided by other middleware implementations and the distribution of all the information that will provide the automation of the future ATM system by giving a common operating picture to all participants (applications and systems both airborne and ground-based) allowing real-time data sharing, network-enabled operations and improving the air traffic operation.

Thus, SWIM is the net-centric service-oriented information management infrastructure that will support the future ATM systems and involves both low-level communication services that enable data interaction among heterogeneous distributed systems and high-level information management services, including those specific to the ATM realm (e.g. dissemination of meteo and aircraft performance data, trajectory prediction, trajectory management services, etc).

Taking into account that these systems are defined in the scope of SESAR as SWIM-enabled ATM systems, the specific SWIM implementation developed in the ATLANTIDA project has been designed for being used in UAS as platforms governed by SWIM-enabled ATM applications including, between its features, real-time middleware requirements, heterogeneous type of data, requirements about data rates, datalink and SWIM latencies, a good set of quality of services (QoS) policies and support for different hardware and software (HW/SW) platforms.

Keywords
UAS, ATM, RT-Middleware, SWIM, SESAR

INTRODUCTION
ATLANTIDA (Application of Leading Technologies to UAS for R&D in ATM) is a €30M project half funded by the Spanish Center for Technological and Industrial Development (CTDI) under its CENIT program. The initiative is conducted by a consortium of 17 aerospace industries and IT companies plus several universities and R&D organizations under the lead of Boeing R&T Europe. Integrasys is part of the ATLANTIDA consortium, contributing to the development of the SWIM system as the unique way for the integration of the large and complex number of applications participating in the overall scenario (Figure 1).

Figure 1. The ATLANTIDA project scenario

The ATLANTIDA initiative has invested a lot of effort for the deployment of a real demonstration scenario in which UAV/UAS can be integrated within an ATM scenario based on SWIM operations. The complexity of the ATLANTIDA scenario deals with the integration of disparate hardware and software (HW/SW) components among which it is worth to mention the avionics platform of the experimental UAVs, the on-board software applications embedded on experimental UAVs, the datalink which ensures air-ground communication capabilities and the software applications deployed on the ground platform.

The work presented here shows the specific approach, design and implementation of the lower-level layer of SWIM in the scope of the ATLANTIDA initiative. In this effort, SWIM is conceived as a real-time middleware...
infrastructure providing information integration services to distributed applications which are deployed on the ground as well as on airborne platforms embedded on UAVs used for experimental purposes.

This middleware infrastructure has been designed according to a principle of decentralization where the main objective is to implement a common SWIM layer that must be present in every single component of the global ATM system so all the different ATM system components may use it to communicate among each other, according to two different information exchange paradigms: request/reply (R/R) and publish/subscribe (P/S).

This SWIM layer is understood as a technology-agnostic layer providing abstraction from the underlying communication technologies selected for the particular implementation of the R/R and P/S communication paradigms, in our case CORBA[1] and DDS[2] respectively, technologies that have been proved successfully during all the ATLANTIDA project life cycle, with a common data definition and interoperation model based on the well-known Interface Definition Language (IDL) standard by the OMG.

The resulting SWIM of ATLANTIDA has been able to govern the communication procedures that occur among the different subsystems with different kind of communication procedures in different location (Ground/Ground, Air/Ground and Air/Air) enabling the integration within the project scenario of heterogeneous applications developed in different programming languages (C/C++, Java and ADA) and for different platforms (Linux, Windows).

Every ATM system in the scenario has been implemented based on the SWIM Application Programming Interface (API) in order to achieve the integration of the applications and the fulfilment of the ATLANTIDA simulation scenarios. The SWIM API has been evolving all through the project life cycle -four years- in order to achieve the requirements and integration needs of the different ATM systems. In this sense, the SWIM API development has been performed according to properly defined configuration management policies, being SWIM API version 5.0 the latest version that has been accomplished in the project.

The SWIM API version 5.0 provides the following set of services:

- Request/Reply (R/R) communication services based on a point-to-point architecture supporting point-to-point communications.
- Publish/Subscribe (P/S) communication services supporting point-to-multipoint communications.
- Interface Management based on a centralized Naming Service for managing the R/R communications
- Swim Message Service, with monitoring and reporting capabilities on the interactions among the SWIM actors.
- QoS communication policies configuring information handling:
  - P/S communications: data delivery policies (reliable, best effort), data volatility policies (durability, history), timing policies (deadline, liveliness, etc) and fault-tolerance policies.
  - R/R communications: timing policies (Relative RoundTrip Timeout) and policies for one-way operations (Sync Scope).

Having in mind that this paper is submitted to the industrial track of the ATACCS 2011 conferences, the following sections are focused on the description of work done, practical evaluations of methods, techniques and tools.

PROBLEM ASSESMENT

This chapter presents an introductory analysis regarding the assessment of the different technical challenges of the ATLANTIDA project, taking into account the System Architecture of the proposed ATM scenario and the different technologies that have been evaluated for the implementation of the SWIM infrastructure.

System architecture

Figure 3 presents an overview of the system architecture regarding the Air and Ground Segments of the ATLANTIDA project scenario.

In one side, the Onboard Platform of the real UAV (Airborne segment) is governed by a real-time operating system (RTOS) which is in charge of the execution of several SW modules, each of them with a specific functionality related to a concrete ATM service.

The SWIM platform will guarantee the communication procedures that take place among embedded SW modules (Air communication) and the communications procedures between the Ground SW modules and the Onboard SW modules (Air-Ground communications). Figure 3 also represents the HW subsystem of the onboard node (Single board computer SBC, I/O card, sensors, datalink, actuators) and the drivers that make possible the integration of all the hardware components in the software system.
On the other side, the Ground Segment is integrated by several nodes with heterogeneous HW and operating systems. The nodes are connected according to a standard Ethernet architecture. Each node may hold one or more SW applications with a specific functionality related to a concrete ATM service. The SWIM platform guarantees the communication procedures that occur among ground SW modules (Ground-Ground communication, G/G) and the inter-communication procedures that take place between the onboard and ground SW modules (Air-Ground communication, A/G).

Figure 3 includes a special node with the COMGND application, which makes possible the Air-Ground communications through the wireless datalink. Furthermore, Figure 3 includes another particular node which is referred as the Name Server: this node will be responsible for the deployment of the SWIM Naming Service.

Middleware platform
The ATLANTIDA Middleware (MW) matches to the concept of the integration between software components in both the Ground and Onboard infrastructures by providing hardware and operating system abstraction to the software applications.

The MW platform is the basis for the implementation of the SWIM infrastructure that enables the different communication scenarios established between the SWIM actors. It includes functionalities for delivering information, supports different information exchange patterns and includes point-to-point (“one-to-one”) and multipoint (“one-to-many”) communication capabilities.

The SWIM infrastructure is based on the following communication models: Publish/Subscribe (P/S) and Request/Reply (R/R). An analysis of the available middleware technologies was initially performed in order to assess and select the most suitable platforms that will hold the P/S and R/R communications paradigms. In this analysis we focused on the middleware technologies that were both (i) standard-based and (ii) conformance tested with the standards:

- CORBA (Common Object Request Broker Architecture) [1]
- DDS (Data Distribution Service for Real-Time Systems) [2]

The previous specifications are all defined by the Object Management Group (OMG). DDS is perfectly suitable for real-time scenarios and CORBA has also an extension [3] with support to real-time.

Request/Reply paradigm
Request/Reply is a message exchange pattern that involves a point-to-point interaction in which there are two different actors: the server and the client. The client sends a single request to the server and then it waits for a response. The server processes that request and then it sends a reply message to the client.

In the scope of ATLANTIDA the Request/Reply paradigm is implemented with CORBA technology. This technology has been widely used since the decade of 1990 for distributed applications based on client/server architectures. It is quite suitable for point-to-point communications, providing good response times for client/server connections with reasonable bandwidth.

Figure 4 presents the general architecture for a CORBA-based application including client and server. As it can be seen the subjacent infrastructure that lies below the CORBA infrastructure is the TCP/IP protocols. Besides, the Internet Inter-ORB protocol (IIOP) is the CORBA component that makes possible the interoperability among the different CORBA distributions.
In support to JAVA: ORB SUN (JDK version 1.6) [9]
In support to ADA: PolyORB version gpl 2007 [10]

Publish/Subscribe paradigm
Publish/Subscribe is a message exchange pattern in which there are mainly three different components: publishers, subscribers and topics. There may be several publishers that generate messages which are represented as topics. The subscribers indicate the topics in which they are interested and then they receive alerts when the selected topics are available for reception. This communication model can be defined as “one-to-many” since there may be several subscribers for a single publisher.

In the scope of ATLANTIDA the Publish/Subscribe paradigm is implemented with DDS technology. Figure 5 presents the general architecture for a DDS-based application including publisher and subscriber. As it can be seen, the subjacent infrastructure that lies below the DDS infrastructure is the UDP protocol.

Several distributions have been tested in order to give to support to programming languages Java and C/C++. No distribution has been tested yet in order to give support to ADA programming language. The following DDS implementations have been already tested:

- OpenDDS version 1.0 [4]
- RTI-DDS version 4.3e [5]
- OpenSplice 4.1. [6]

OpenDDS is an open-source implementation of the DDS standard and it is conceived as an extension of the TAO distribution. It is available in C/C++ and it only implements the basic functionality of the DDS standard.

The commercial distribution implemented by RTI is conformance with the DDS standard and its API is currently available both in C/C++ and JAVA. The OpenSplice distribution implemented by PrismTech is also conformance with the DDS standard and it is interoperable with RTI-DDS [11].

PRESENTATION OF THE SOLUTION
At the beginning of the project there was a great effort requirements specification and architectural design of the ATLANTIDA scenario based on an Architecture Control Document (ACD) which included a detailed description regarding:

- The ATM systems and its allocation to the software hosts of the deployment scenario
- The use cases that demonstrate the interactions among the ATM systems through the SWIM platform
- The topics and interfaces for the Publish/Subscribe and Request/Reply communication models, respectively.
- The specific R/R and P/S interactions among the SWIM systems
- The QoS configuration for the ATM systems

Once the architectural design was completed, the next step was to define the communication interfaces (R/R, P/S) in a low level language that could be directly applied to the implementation of the final ATM applications. The IDL standard language was selected since it was proved to successfully interoperate among the subjacent selected middleware technologies, i.e. CORBA and DDS.

The definition of the ATLANTIDA data model in the IDL language was a very important step that leaded to the development phase of the different ATM systems and the corresponding software components of the ATLANTIDA scenario.

At the beginning the ATM systems were developed according to the preliminary SWIM API versions that were already available in the first development phases of the project. Then the project development evolved according to an iterative pattern, progressively increasing the functionality of both the SWIM API and the ATM applications that were linked to it.

Architecture
The aim of this section is to introduce the software architecture of the SWIM infrastructure and the corresponding high-level applications that make use of the SWIM services. The description is focused on the internal architecture of the SW modules represented in Figure 3.

In general, a SW module will hold a specific software application being the MW platform the responsible for its integration—and interaction—with the rest of SW modules running in the overall domain or application scenario.

From now on, the MW platform will be referred to as SWIM module representing the conceptual implementation for the internal architecture of any SW module in the project as shown in Figure 6.
The SWIM Module is the software implementation that has been designed in order to achieve the integration of every subsystem that belongs to either the Ground or the Airborne Segments.

The following requirements and features have been considered:

- The design must be based on the principle of decentralization: the SWIM module must be present in every single subsystem of the ATLANTIDA scenario in form of dynamic or static library linked with applications. In this way the systems will be able to communicate to each other through the SWIM platform.

- The SWIM module must provide abstraction and transparency so that software applications can be distributed in several nodes with different operating systems. The following operating systems shall be supported by the SWIM Module: Linux-based RTOS, Linux distributions (UBUNTU, Debian, etc) and Microsoft Windows.

- The main goal of the SWIM Module must be its capabilities to hold and encapsulate the different communication models that might take place among the SW modules that belong both to the Ground and Air segments. Specifically, the SWIM module shall implement the Request/Reply and Publish/Subscribe communication paradigms.

- The data model used to define the communication interfaces and data information exchanges that may occur among the SWIM systems must be common between all the ATLANTIDA systems. For this purpose, the IDL standard from OMG was selected, being used according to the syntax required for the implementation of SW applications.

The software applications that implement the communication interfaces defined in IDL are developed in C/C++ and JAVA.

As it is stated above, the SWIM module implementation is based on the well-known middleware technologies CORBA and DDS, which represents both the core components and subjacent technologies for it.

The middleware distributions used in the scope of the ATLANTIDA project has been:

- CORBA (Common Object Request Broker Architecture). TAO version 1.5a
- DDS (Data Distribution Service). RTI-DDS version 4.3e

Besides these core elements, an additional API has been developed in order to complete the full functionality of the SWIM module. This API has been referred as the SWIM API in the context of the project and can be represented as in Figure 6.

**Implementation**

The ATLANTIDA SWIM API consists in a set of software libraries that implement the functionality that has been agreed according to the requirements defined in the project:

- Communications service based on R/R and P/S communication models, implemented on CORBA and DDS technologies, respectively.
- Several QoS policies configuring information handling
- Interface management based on a centralized SWIM Naming Service
- Swim Message Service, which monitors and reports all the interactions among the SWIM actors.

This section will be focused on the description of the most advanced functionalities of the SWIM API, namely the SWIM Naming Service and the Swim Message Service.

**The Naming Service**

The Naming Service is encapsulated within the SWIM platform in order to enable the R/R communications model that is shown in Figure 7. The communication procedures between clients and servers are governed by the so-called Name Server, which is an administration entity that is executed as an additional service (daemon) in a ground node, as it is depicted in the architecture in Figure 3.

As it is stated above, the SWIM module implementation is based on the well-known middleware technologies CORBA and DDS, which represents both the core components and subjacent technologies for it.

The middleware distributions used in the scope of the ATLANTIDA project has been:

- CORBA (Common Object Request Broker Architecture). TAO version 1.5a
- DDS (Data Distribution Service). RTI-DDS version 4.3e

Besides these core elements, an additional API has been developed in order to complete the full functionality of the SWIM module. This API has been referred as the SWIM API in the context of the project and can be represented as in Figure 6.

**Implementation**

The ATLANTIDA SWIM API consists in a set of software libraries that implement the functionality that has been agreed according to the requirements defined in the project:

- Communications service based on R/R and P/S communication models, implemented on CORBA and DDS technologies, respectively.
- Several QoS policies configuring information handling
- Interface management based on a centralized SWIM Naming Service
- Swim Message Service, which monitors and reports all the interactions among the SWIM actors.

This section will be focused on the description of the most advanced functionalities of the SWIM API, namely the SWIM Naming Service and the Swim Message Service.

**The Naming Service**

The Naming Service is encapsulated within the SWIM platform in order to enable the R/R communications model that is shown in Figure 7. The communication procedures between clients and servers are governed by the so-called Name Server, which is an administration entity that is executed as an additional service (daemon) in a ground node, as it is depicted in the architecture in Figure 3.
The Name Server is responsible for the following tasks:

- Register the different servers by means of a specific naming hierarchy (register service).
- Listen to the client requests and send them the correct reference location for the server that matches the client request (look-up service).

Once the client has received the server reference location (look-up service), the client is ready to invoke the server operations defined in the corresponding interface (invoke service).

**The Swim Message Service**

As it is explained previously, the SWIM actors interact with each other through the SWIM platform according to the following communication models: Request/Reply and Publish/Subscribe. The main goal of the SWIM Message Service is to monitor the different interactions that might take place among the SWIM actors of a certain scenario. Furthermore, the SWIM Message Service will be useful in order to determine if a certain SWIM system is or not active within the execution of a particular scenario.

The design of the SWIM Message Service is based on the specification of the different events that might be monitored with this infrastructure. These events are identified based on the entity which is responsible for originating the event. The entity can be referred as a software module which belongs to a certain SWIM system, and the software modules can be classified as client, server, publisher or subscriber.

The following is a specification of the different types of events that might be monitored by the SWIM Message Service:

- **State Events**: they determine if a specific software module (client, server, publisher or subscriber) is or not active within the execution of a certain SWIM system in a particular scenario.
- **Interaction Events**: they determine if a specific software module (client, server, publisher or subscriber) is interacting with other software modules at a certain moment of the execution of a particular scenario. The interactions among the software modules can be either Request/Reply or Publish/Subscribe. The software modules might belong to different SWIM systems.

The implementation of the SWIM Message Service is based on the deployment of an monitoring software application within each of the SWIM systems of a certain ATM scenario. This software application will be able to capture the state and interaction events that might occur within the execution of each SWIM system. Once an event is captured within a SWIM system, the monitoring software application is able to report this event by means of a specific message (topic) sent through a publisher module.

**RESULTS**

The main results that can be highlighted from the ATLANTIDA SWIM implementation are:

(i) Its flexibility for number of systems successfully integrated,
The heterogeneous type of data used ranging from small data structures to large amounts of raw data representing meteo images, etc (see Table 1),

The range of platforms supported (from Linux-based RTOS to non-critical Windows-based systems) and

The dynamism introduced regarding the analysis of different nature carried out: data rates, performance metrics, latencies, and data monitoring and systems interaction recording, etc (see Figures 9 and 10).

Table 1. SWIM Compliance matrix of technologies, programming languages and HW platforms supported

<table>
<thead>
<tr>
<th>CORBA</th>
<th>JAVA</th>
<th>C++</th>
<th>ADA</th>
<th>DDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>CORBA</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>JAVA</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>C++</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADA</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The architecture benefits

The ATLANTIDA project has been results driven, addressing the developments to real flights and pursuing the R&D in ATM, 4-D trajectories automation, data gathering and analysis.

The specific SWIM architecture developed has been consolidated as a real-time communications middleware platform which enables the integration of the distributed applications within the ATLANTIDA scenarios for both ground and airborne systems.

The resulting SWIM implementation is present in every ATLANTIDA system, fulfilling the main milestones of the project:

- Simulation of ATM scenario in the lab: the SWIM component is present in the synthetic aircraft, the virtual aircrafts, the hardware-in-the-loop (HIL) aircraft and the ground systems

- Flight Campaigns with UAVs: the SWIM component is present in the avionics platform embedded on the UAVs and also in the Monitoring and Control Systems of the ground infrastructure.

Besides of this, as the final aim of the ATLANTIDA project was to demonstrate several simulation scenarios in which the ATM Systems could perform their corresponding functions based on the SWIM, a great integration effort has been invested in order to achieve a uniform scenario with every system integrated with the SWIM version 5.0, leveraging from the Swim Message Service and thus enabling the integration of external applications which are able to register and analyze the data obtained from the ATM simulations.

The resulting architecture on the final scenario was based on the following main segments (Figure 11), where each of the segments comprises several ATM systems which are connected through the SWIM platform according to the SWIM API version 5.0.

- The ground ATM systems, which are deployed in the so-called TIMEO platform of the ATLANTIDA lab.
- The synthetic aircraft which emulates a real UAV, which is deployed in the so-called CRITIAS-TEST platform of the ATLANTIDA lab.
- The virtual aircraft instances created within a simulation framework (SIM subsystem), which are deployed in the TIMEO platform of the ATLANTIDA lab.
- An additional aircraft based on a Hardware-in-the-loop (HIL) architecture. This aircraft includes several ATM airborne systems which are deployed in the CRITIAS platform of the ATLANTIDA lab.
- The resulting architecture on the final scenario was based on the following main segments (Figure 11), where each of the segments comprises several ATM systems which are connected through the SWIM platform according to the SWIM API version 5.0.
Figure 11 depicts the deployment distribution of the ATM ground and airborne systems of the ATLANTIDA final scenario. The ground systems are shadowed in grey color and the airborne systems are shadowed in green (synthetic aircraft), blue (HIL aircraft) and yellow (virtual aircrafts).

One of the main technical challenges that have been faced within the ATLANTIDA initiative is the integration of the software applications being developed by the different partners (industries, universities, etc) involved in the ATLANTIDA project. A lot of coordination effort has been invested within the consortium in order to achieve the integration based on a common information model based on the Interface Definition Language (IDL) and a common Application Programming Interface (namely “SWIM API”) which enables the provision of the SWIM services to the final ATM applications.

REFERENCES
1. Catalogue of OMG CORBA/IIOP specifications:
2. Catalogue of OMG Data Distribution Service (DDS) Specifications:
3. OMG Real-Time CORBA specification:
   http://www.omg.org/technology/documents/formal/real-time_CORBA.htm
4. OpenDDS:
   http://www.opendds.org/
5. RTI-DDS:
7. CORBA and DDS Integration:
8. TAO - TheAceORB:
   http://www.theaceorb.com/
   http://www.cs.wustl.edu/~schmidt/TAO.html
9. SUN ORB
   http://java.sun.com/javase/6/docs/technotes/guides/idl/index.html
10. PolyORB:
    http://polyorb.objectweb.org/
11. Interoperability among DDS
A Flight Trajectory Generator For a PC-based Analysis Tool

Jaime García Sáez
Air Traffic Management Directorate
INECO
Madrid, Spain
+34 914521398
jaime.saez@ineco.es

ABSTRACT
This document bases on an innovation project by INECO aimed at developing and integrating a flight simulator in ATM (Air Traffic Management) simulation platforms. The paper focuses on the work done so far in the validation and checking of instrumental procedures. It also documents a method for generation of aircraft flight trajectories based on average flight conditions.

Keywords
Procedure, checks, ARINC rules, radio electric coverage, dynamic model, average path.

INTRODUCTION
This paper focuses on the flight trajectory generation facility available within the Flyability Simulator project, although these introductory words are intended to present the whole project. The main goal of the Flyability Simulator is the development and installation of a flight simulator that truly reflects the aircraft performances and studies the flyability of the instrumental procedures designed within the Aeronautical DG (Directorate General) of INECO. Increased deployment and use of Area Navigation (RNAV) leads to more diversified navigation routes and to improved utilization of available airspace. This is a competitive tool to generate flight trajectories, providing aircraft models (Flight Dynamic Model and Flight Control System, autopilot, engines) and a simulation environment to test instrumental procedures.

Computer simulations of air traffic are a major source of quantified estimates of system benefits. So, this tool is also intended for ongoing SESAR initiatives and any other project requiring simulated trajectories for disparate purposes.

There is a need to have a tool able to check against a set of known coding rules (ICAO norms in doc 8168, ARINC spec 424) and verify the arrival, departure, approach, missed approach and en route instrumental procedures with a wide range of testing, taking into account different aircraft models, different loads and different weather conditions. A practical and usable model requires sufficient modeling detail to allow evaluations of the impact of alternative route designs and navigational procedures while limiting simulation run times. To this respect, the project has well-equipped premises with workstations and a visual system providing high quality performances of the application.

The simulator is currently adaptable and configurable in some key performance parameters and it will be integrated with other ATM simulation platforms (route, tower and control) in the short time. Furthermore, it is widely known that allowing for variability in aircraft operational or navigational performance or external conditions may necessitate repeated searches for desired solutions. Consequently, rapid generation of flight trajectories is essential in order to maintain adequate simulation run times. The tool will also be improved in order to run environmental studies such as noise, carbon dioxide emissions and so on, due to the increased concern about the effects of air traffic on people and environment. For instance, it can help in the testing of CDA trajectories. Again, there are collaborations with projects like DORIS, RETACDA and CDAs in PMI to build a Flight Data Analysis module within the scope of the Flyability Simulator project.

Technically, the simulator checks the functions of a procedure under different conditions before its approval, testing it against a low cost and wide set of studies. It has real time requirements so the mathematical models are complex (especially when calculating the average path flight) and they intend to faithfully reproduce the inputs to the real system. With this support tool, some uncertainties in the running of some services are considerably reduced to a minimum (for instance, delays in procedure publication as an important service quality factor) and the airport authorities may rely in this decision-support element to know before real operation the effectiveness of some measures and improve the resource usage. Though the certification of this tool is not a goal itself, it could be considered later on if the need is detected.

SIMULATION OVERVIEW
The Need
The trajectory traced by an aircraft in an instrumental flight is based on a complex and previously designed schema. One of the most important stages is the design phase of instrument flight procedures.
The modeling approach presented in this paper defines a small set of trajectory model parameters for vertical and horizontal planes.

There is a concern that simulation-based validation of new and/or updated procedures is a crucial step in Procedure Design. This step covers the following operations, included in the Simulator:

- Checking for completeness of data (already implemented).
- Checking about criteria applied: minimum distance between waypoints, climb/descent gradient, abatement noise constraints, etc. (analysis done, in development).
- Validates distances to/from waypoints (already implemented).
- Validates courses between waypoints: Angle between two consecutive segments is within limits. Generally, the maximum angle should be less or equal to 120 degrees, except in approach maneuvers where the maximum angles are 90 degrees in intermediate fix (IF) and 45 degrees in final fix (FAF) (analysis done, in development).
- SID: check the coherence between established altitude constraints and published minimum ascent gradient (analysis done, in development).
- Generation of average flight (implemented for DF (direct to a fix), TF (track to a fix), CF (course to a fix) path terminators and FlyBy/FlyOver waypoints). Implementation of CA (course to an altitude) is pending.
- Line of sight and power coverage studies: aircraft is in direct view from a radio help element. In other words, the aircraft is covered by a terrestrial station (to be implemented in conjunction with COVER tools, developed by Ineco).
- Checking about path terminator sequences and parameters (already implemented).
- Finally, checks flyability through a simulation engine (models implemented (validation pending), integration in Simulator project pending).

These steps will be further described in subsequent sections of this document.

The goal of that phase is the operational safety, taking into account how to clear obstacles in a safe way. For this purpose, according to the navigational sensor, each phase of a flight uses a set of protection surfaces through the trajectory and a set of values for obstacles clearance. The surfaces are defined in the most unfavorable case, assuring that all trajectories flown by different kind of aircrafts are within operational margins.

The norms followed and implemented in the flight procedure design Editor Tool are mainly described in the documents 8168 – OPS 611 (Aircraft Operations Volumes I / II).

Once the procedure is designed and after some subsequent stages, the maneuvers are published and some problems may arise since then, such as:

- The trajectories flown by the aircrafts do not adjust to the initial requisites. For instance, there is no way that aircrafts may adjust to the procedure in some turns to avoid entering the airspace above a particular town. Then, the process should start from scratch, incurring in higher costs and time.
- Some specified maneuvers cannot be flown by part of the fleet. That is, a number of aircrafts cannot meet the procedure requirements totally or partially.
- Some maneuvers cannot be validated due to a lack in signal quality and/or radio electric coverage.

The forecast of the aforementioned problems is very important from a social, political and economical perspective, so that a previous simulation is necessary for checking procedures in real situations and reducing the time needed to publish the final procedure.

Finally, the goal is a tool providing a flight replica in the most accurate and realistic way, maintaining the integrity of the specifications and covering different aircraft models and meteorological patterns, which is one of the goals of the Flyability Simulator.

**Functionalities**

Generically, the flyability simulator will have, at the end of the development (Q3 2011), the set of functions described under this point. All functions fulfill the main goal, which is the determination of the flyability of a procedure. In the design of a flight procedure, one of the first steps is the compilation of information related with aeronautics, cartography and obstacles. There are different types of elements:

- Point: The element is defined by mean of coordinates.
- Vector: The element is defined graphically by a series of points.
- Geo-referenced raster Files, such as satellite images, maps and so on.

The representation of those elements in the flight simulator is relevant from the point of view of decision-makers. The user can add tags to the elements and choose the type of information that should be showed.

Before a procedure can be run under different inputs, there is a lot of checking activities to be done. This depends on the navigational mode selected – conventional or RNAV -. Any incident detected or non-conformities with the design rules are reported to the user.

**Trajectory Model**

The conceptual model views a flight trajectory as an estimate of the anticipated behavior of an aircraft in space and time based on:

- Characteristic flight performance of the aircraft: This is done by means of Flight Dynamics Model (FDM) and Flight Control System (FCS).
- ATC requirements: In addition to aircraft operational considerations, key procedural speeds often take into account general ATC requirements.
Conversion of Designed Procedure to Piloting Procedure: Actually the Flyability Simulator relies on a Procedures Editor Tool to load the procedure into the system. Piloting procedures provide key aircraft operational speeds and general timing information detailing when speed transitions are typically initiated.

The aircraft type largely determines the main features of a flight trajectory. Performance data typically include speeds, climb/descent gradients and so on. The studies carried out with JSBSim model data as functions of aircraft gross weight and atmospheric conditions. There are also procedure constraints with regards to gradients and speeds that need to be included as input parameters. These data are read from a procedure chart. They are generally imposed by Air Traffic Control (ATC). For example, general speed restrictions apply to flight at altitudes below 10000 feet mean sea level (MSL), often limiting the on-course climb speed to 250 KIAS.

The route is defined by a sequence of navigation fixes that typically include charted radio navigational aids, airway intersections, radicals and distances relative to a fix, or latitude and longitude information of a waypoint. In order to evaluate this model, a lot of parameters are taken into account covering different areas (atmosphere, velocities, coefficients, position, propulsion, massprops, aerosurfaces, FCS and so on). This means slower computational rates in order to fully characterized vertical and horizontal planes achieving 6-Dof (Degrees of freedom) and tracking departures and arrivals/approaches, testing aircraft performances and not flight data plan. Overall, the flight is based on RNAV and flight modeling and takes into account control constraints.

Procedures Editor Tool
Next, there is a set of three snapshots of the Procedures Editor Tool:

- First: Waypoints Editor. There is an option to edit/add/remove WPs.
- Second: A detail of the path terminator combo is shown.
- Third: Main window of the Procedures Editor Tool.

RNAV Checkings
Theoretically, area navigation is possible with different combinations of sensors. Currently in Spain the only approved method is DME/DME though is foreseen that GNSS will be the main help system for navigation in departures and approach procedures where aircrafts fly in low altitudes so reception from terrestrial stations decreases.

In order to define a trajectory a set of points called waypoints is needed, as well as the way the aircraft will trace the segment defined by a couple of consecutive points called path terminators. All data used by a RNAV system is stored in a database. This database is coded according to the norms published by aviation industry. The main spec is ARINC 424 ‘Navigation System Database Specification’. The Procedures Editor Tool (already implemented) takes also into account the application of navigation specification by flight phase, that is, there is a correlation between the flight phase and the navigation specification according to the doc 9613 of ICAO. This norm applies to a specific RNAV called Performance-based Navigation or PBN.

Before examining the flyability of the procedure, a set of checkings is an essential requisite to validate the correct codification of the procedure and the theoretical signal quality. Therefore, the checkings defined are:

- Check along the trajectory the fulfillment of signal quality requirements in terms of accuracy, availability, service continuity and integrity, as well as radio electric coverage DME/DME or GNSS.
- Check there are no errors in coding the procedure, so that all waypoints have an assigned name, their geographic coordinates are defined, each segment has a path terminator and each path terminator has all required leg data fields.
- Check that rules coded in specs are followed:
  - The need of FlyOver waypoint.
  - Initial and final leg as a function of RNAV procedure.
  - Path terminator sequences allowed from current leg to next leg. To this respect, the type of path terminator allows in the Procedures Editor Tool is one of the following: IF, TF, DF, CA, CF, FA, FM, HM, RF, VA, VI and VM.
- Run the checkings defined for conventional navigation except the coverage ones.
Each segment needs a minimum distance for the aircraft to stabilize. This minimum distance depends on the waypoints (FlyBy, FlyOver), a turn in the flight path, altitude, speed, roll angle and path terminator type. The latter parameters need some calculations, such as:

- Minimum stabilization distance will vary for FlyBy and FlyOver waypoints, as seen in the diagrams below, where distances and radio turns are computed.
- Check the route between two waypoints according to the coordinates defined in the origin and final positions, based on orthodromic distances and spherical geometry.

In case there is a mismatch between calculated and defined coordinates and/or minimum distance cannot be set, the user will be informed.

Operations

Once the procedure has been designed and validated, it is ready to start running simulations under different conditions, using the JSBSim-1.0 simulation engine as core tool, which is an open-source, platform-independent, flight dynamics model in C++. Then, the trajectories obtained and associated parameters can be further analyzed as graphs (CSV Processor module) or in a visual subsystem (KML file, Flight Gear playback module). This is already implemented.

Trajectories are generated using data from an aircraft performance model, based on C++ libraries (aircraft dynamic model in JSBSim) with autopilots and events defined for each aircraft. It works with a 6-DoF (Degrees of Freedom) model. The FDM software library is reasonably easy to comprehend and due to the ease with which it can be configured, it has proven to be useful in flight simulation applications such as this project. Some models such as C172x, B747 and A320 have been already implemented, refined and tested, though validation activities have not been carried out yet.

Multiple trajectories can be displayed simultaneously in CSV processor view and 3D view (based on the functionalities provided by Google Earth API). Some filtering criteria have been also defined so only a subset of results is shown. This has also been implemented.

Meteorological conditions can be modified and are user-configurable (see table below), as well as simulation start conditions are selectable (Start altitude/Start position/Level flight/Velocities and angles and so on). This has been analysed but not implemented yet.

The flyability simulator provides the following operations:

- Translation of path terminator into maneuvers for each leg in the procedure. This drives the configuration of the script to be run in the simulation engine. There is a FDM and an autopilot with specific maneuvers for each aircraft modeled in the application.
- Test a procedure, selecting different aircraft models and a combination of input parameters such as maximum/minimum loads, maximum/minimum temperatures and winds in different directions.

The user makes his choices though there is a predefined and automatic set of tests ready to be used, as reflected in the table below.

<table>
<thead>
<tr>
<th>Test</th>
<th>Load</th>
<th>Temperature</th>
<th>Wind</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Max</td>
<td>Max</td>
<td>None</td>
</tr>
<tr>
<td>2</td>
<td>Max</td>
<td>Max</td>
<td>Max side</td>
</tr>
<tr>
<td>3</td>
<td>Min</td>
<td>Min</td>
<td>Max side</td>
</tr>
<tr>
<td>4</td>
<td>Min</td>
<td>Min</td>
<td>Max back</td>
</tr>
<tr>
<td>5</td>
<td>Min</td>
<td>Min</td>
<td>Max forth</td>
</tr>
</tbody>
</table>

- Test a procedure with operational constraints defined by the user in terms of climb gradient, altitude constraints and meteorological conditions (ISA standard or user-defined). For instance, noise abatement departure procedures. The norm 8168 volume I establishes two NADP methods to reduce the noise in the end of the runway and in far-away areas. Below there is the graph for the first method.
Visualize the trajectories. They are tagged and activated/deactivated by the user. Also, a search function to get the desired trajectories according to some user criteria is currently under development.

Represent the mean trajectory and the aircraft parameters at any point of the flight. This is not the average path flight already computed from the procedure description, but additional computing based on simulated trajectories output by the simulation engine.

Study the feasibility of turns, so a partial running of the procedure should be available. The user selects the start and end points of the simulation as well as the input conditions. This is to be done in later stages of the project. It’s not covered yet. Some theoretical analysis has been done and mockup developments are available in order to evolve the mathematical model. Right now, flyby and flyover waypoint types are considered and DF, CF and TF path terminators are calculated.

In a later stage of the project, deviations against real radar data to check the consistency of the simulations will be studied.

TECHNICAL APPROACH
This point explains the testing and analysis with different simulation engines and the requisites from the point of view of project goals.

Simulation Engines
In a first step, some open-source simulation engines like FlightGear (FG) and JSBSim have been deeply tested in order to evaluate their applicability to the Project. A general analysis was done regarding their characteristics and the value offered to the fulfillment of project goals. The analysis covered the following areas: basic features, ease of installation, usability, openness, and in-depth analysis to check the requisites needed in terms of simulation software. Those requirements are:

- Source code available.
- Configurable software.
- Model aircrafts
- Get aircraft parameters out of a simulation running
- Real time interaction with the simulation
- Generation of different scenarios (airport and surroundings)
- Well documented

Both simulation engines fulfill the basic requirements and allow the configuration and modeling through files and code. With regards to output data, JSBSim has more interactive characteristics than FG, but FG is better as simulation core engine. So, finally both are used due to the flyability simulator architecture is modular and different modules point to different goals. Therefore, JSBSim aims to visual subsystem and runs in batch mode through script generation. FG gets input data from the user and outputs verbose files with the trajectories.

Based on a procedure description, the trajectory model then generates a 4D flight trajectory for the full route of a flight specifying time, latitude, longitude and altitude and other pertinent data of the flight at each defined point along the route. Trajectory points include the discrete events associated with a set of maneuvers within a procedure, as well as the events related to aircraft operational procedures. Aircraft operational procedures are grouped by the phase of flight they occur in. The simulator focuses on SID, STAR, APP and Missed APP instrumental procedures.

The model runs through discrete event simulations of flight trajectories using simulation time steps of variable length. The parameterization of inputs and outputs is up to the user and intended to fixed-wing aircrafts such as B747, for instance. The simulation assumes that all accelerations are executed at a constant rate.

It is noteworthy to point out that even though piloting procedures are modeled by sequentially establishing a number of discrete indicated airspeeds throughout the various phases of flight, the actual or true airspeed (TAS) of the model aircraft generally changes realistically as altitude is gained in a climb or altitude is lost during descent.

For example, during a climb at constant indicated airspeed, the true airspeed continually increases as the aircraft gains altitude. This increase is a result of changes in atmospheric conditions which give rise to greater divergence between indications of the airspeed indicator and the true airspeed of the aircraft. While the airspeed indicator is calibrated to indicate true airspeed at MSL and in standard atmospheric conditions, its indications remain below the true airspeed of the aircraft at a rate of approximately 1.5 to 2 percent per 1000 feet of altitude.

Flight parameters are defined for each aircraft type including accelerations, speeds, altitudes, climb/descent gradients, roll/pitch/yaw angles, etc. The current version of the model includes effects due to wind, temperature and weight.
The images show a departure from Madrid-Barajas airport. The second one includes a turn to a fore point while the first one starts in the runway and goes up to a certain altitude following a straight course.

The average flight path trajectory model was implemented using the programming language Java 1.6 and using aeronautical charts. Also JSBSim models and enriched sceneries for Flight Gear with new objects related to the procedure were developed in order to test the designed procedures with simulated aircrafts. First, required modeling input such as geographic airport and air navigational data are read from a database. To this respect, the runway elevation threshold, DER (Departure End of Runway) information and design constraints are entered by the user. Next, each leg in the procedure is calculated according to some parameters. Among them, path terminator and type of waypoint are the most relevant. After legs are calculated, the user is prompted to select the number of trajectory points per leg. Finally, a KML file is ready to visualize the trajectories flown by the simulator. Also technical analysis through a processor module is able to the final user.

**INNOVATIONS**

This final part of the paper focuses on the main innovations of the whole project and not just in the average trajectory generation tool already described.

The main innovation of the Project is the development of a flyability simulator that enhances the current simulation tools in order to ease the prevalidation activity on the design of instrumental procedures, determining the flyability of it. For this purpose, the designer will work closely with this simulator in order to provide the necessary input (aircraft data, navigation system data, instrumental procedure data, meteorological data) and checking the trajectories and parameters so that concluding the right design deployed in the tool. The completely brand new and out of the box tool set will speed the acceptance of the procedures from aeronautical bodies implied in this crucial issue as there is no other set of tools like this in the market as a benchmark carried out in the first phase of the project assessed.

To air transport players, such as Air Navigation System Providers, Airlines and National Authorities Bodies, this project provides environments (modeled in FG and JSBSim) and means (databases, desktop application) to set up new procedures and to train end users in order to pre-validate the procedures.

The validation of new procedures requires an effective test phase. Moreover, the experimentations need to be carried out within the most highly realistic environments while taking into account other non-functional factors such as time and money.

Thanks to the flyability simulator, a remarkable increase in the procedure design process shall be obtained as the functions deployed with the application will allow, on one side, the performance of reliable procedure analysis in ATM environments and, on the other side, the fulfillment of innovative ways to analyze and evaluate new operational scenarios such as Precision Area Navigation procedures, aircraft performances testing and so on.

Regarding other interesting areas, for instance, multi-queue departure sequencing and separation algorithms require repeated generation of flight trajectories. Even for this case, more complex separation requirements typically demand generation of greater numbers of model trajectories.

The tool can also check aircraft performances with regards to specific maneuvers, such as CDAs, for instance. Today, climb maneuver has speed restrictions and is done in several steps. Tomorrow, there will be a continuous climb, so that new climb paths reduce noise and fuel burn and using aircraft climb capabilities to the fullest extent.

The current concept of Instrumental Procedure sets pre-defined paths at pre-defined altitudes and speed and altitude constraints. In the future, shorter routes may be found by flying at optimum altitude and speed.

In the landing phase of the flight, today there isn’t enough coordination between airports and air-traffic controllers. Moreover, aircrafts are often asked to descent too early and/or put on holding. There are also pre-set speeds and stepped routes. On the contrary, the continuous descend approach is growing as it uses the aircraft gliding capabilities to the fullest extent, allowing for higher precision landings and lower noise impact.

The architecture of this tool is highly modular and flexible to deal with new requirements from SESAR (Single European Sky ATM Research) and/or NextGen initiative programmes.

The visualization tool will allow the analysis of different trajectory data sources, including radar data, simulated trajectories and computed trajectories.

Interoperability is also a concern in the project but it requires internationally agreed standards and, for that, SESAR will deliver the technical basis for defining them through ICAO (International Civil Aviation Organization) SARPs (Standards and Recommended Practices) and coordinated industry standards. The foreseen breakthroughs in this field are explained next.
Platform Interoperability
The main lines in interoperability are:

- Connection with simulation platforms where simulated traffic can validate new control techniques. The Flyability Simulator adjusts to the simulated traffic needs of those platforms. In this sense, there are ongoing works in SESAR initiative whose goals refer to validation of advanced operational concepts.
- GAIA initiative. There was an approach to this group and Flyability Simulator was introduced.
- Gate-to-Gate concept. Today information managed by ATC systems is not shared. So, ATC constraints the trajectory. In the future, ATM communication network will connect different systems so that flights are managed as a continuous event from planning through execution. The Flyability Simulator analyzes the cross-implications of getting together disparate actors in one environment.
- Connection with tools used in environmental impact studies such as noise and gas emissions. To this respect, as referred in the introduction section, the collaboration with three European projects consists in the analysis of different input sources such as in-flight reports, flight plans, FOQA data, flight schedule, radar and wind data, pilots and ATC feedback, AIP, then apply some validation criteria and quality checks to populate a database in order to run assessments (statistical operations, calculations, graphics), filtering criteria (RNP value and deviation from horizontal route, vertical RNP and deviation from the vertical path, analysis of CDAs) and obtain key indicators (deviation of parameters among profiles, fuel flow and fuel burn, number of lost opportunities against time).
- Connection with Green ATM lines: Measure noise and fuel burn in order to test new procedures and generate greener trajectories. In this sense, the Flyability Simulator shall be flexible enough to work with new ATM scenarios.

Flyability Simulator Architecture
This final point shows up a high level view of the Simulator architecture and its connection with external tools, such as Google Earth, Radar data source and COVER Apps (a suite of tools developed by INECO that provide coverage, power and line of sight studies facilities).

There is a Procedure Validation Tool (PVT), an Average Trajectory Generation Tool (ATGT), a Simulated Trajectory Generation Tool (STGT) and a Trajectory Visualization Tool (TVT). There are main entities connecting the tools, which are the Procedure and Trajectory. There are files generated in two main basic formats, such as CSV and KML. Finally, it is used a simulator engine such as JSBSim and Flight Gear in order to test procedures and graphically analyze the results.
Automation and Interoperability Challenges for Heterogeneous UAS Fleet Management

Nicolas Giuditta\(^1\), Marcel Quintana\(^2\), Lorena Fernández\(^1\), Esteban Gutiérrez\(^2\), Joaquim Oliveira\(^1\) and Santi Vilardaga \(^2\)

\(^1\) Deimos Space  
Ronda de Poniente, 19  
Tres Cantos, Spain  
+34 91 806 34 50  
nicolas.giuditta@deimos-space.com

\(^2\) CTAE – Aerospace Research and Technology  
Centre Tecnologia, 19  
E-08840 Viladecans, Spain  
+34 93 664 26 44  
marcel.quintana@ctae.org

ABSTRACT
The command, control, communications and intelligence (C\(^3\)I) (sub)systems and components of an unmanned aircraft system (UAS) is a highly customised solution. Such a dedicated design raises the complexity when integrating a single-aircraft-single-control platform into a fleet of different (i.e. heterogeneous) platforms.

Heterogeneity cannot be avoided in a large fleet of UASs that allows the use of several vendor solutions; meanwhile maintaining or even increasing automation (i.e. and its benefits) could only be achieved by preserving interoperability at C\(^3\)I.

This paper introduces an operational system concept addressing the strategic planning; including a mission planning tool compliant with STANAG, and a performance analysis tool to evaluate the inter-vehicular communications at flight segment level.

Keywords
UAS, C\(^3\)I, STANAG, interoperability, fleet management, mission planning system, inter-vehicular communications, MANET.

INTRODUCTION
From a solely economic point of view, the UAS segment is a continuously expanding niche within the aerospace market. According to recent market reports [1] [2], the world market for UAS may exceed $70 billion by 2015, mostly in the military sector.

With such investments it is easy to forecast that next generation of UASs will possess even better capabilities than today’s, which shall broaden its applicability to other sectors such as security (e.g. frontier surveillance, search-and-rescue, fire fighting), telecommunications (e.g. relays infrastructures), or environment (e.g. agriculture monitoring, forestry maintenance) among others. However, an important market enabler for introducing this technology in non-military applications is to grant the access to non-segregated airspaces through its seamless integration into Air Traffic Management (ATM). This should be achieved through advances in the regulatory and legal framework, the implementation of specific on-board functionalities such as detect and avoid, and the applicability of new standards to certify UAS technology.

The GEMA project, a technology study for heterogeneous unmanned aerial vehicle fleet management, addresses the issue of providing an interoperable mission planning and control system to integrate and operate at C\(^3\)I multiple and varied UAS, preserving at the same time custom UAS vendor solutions. The approach – and the work presented in this study –, is achieved by using compliant message structures and communications specified in NATO Standardization Agreement (STANAG) 4586 [3], and extending its use for fleet management. In addition, the study also researches schemes to improve the operability at flight segment level, through the exploitation of inter-vehicular communications and analyses the performance of a UAS network for a given mission planning. Ensuring ground and air data interoperability is an important challenge to be addressed for fleet management, and the approach is introduced in the Fleet Management section.

The Framework for Strategic Planning section introduces the tools implemented and estimated as Technology Readiness Level (TRL) 3, which follows then with a short review of the selected mission scenarios used in simulations to test and evaluate the concept, in the Evaluation of Tools and Concept Workflow section. Finally, conclusions and recommendations are provided in the last section.

THE INTEROPERABILITY CHALLENGE
Unmanned vehicles have quickly become a precious asset in the military domain with a consolidated potential for civilian applications, as stated in the previous section. Nevertheless, the main interoperability challenge remains nowadays in the operation of these usually legacy systems; due to mainly the non-applicability of standards at C\(^3\)I. This lack of interoperability reduces thus the operational performance and flexibility, emphasising this weakness in joint operations of international or multinational missions such as NATO-based, but appearing as well at national level. Countries like the United States have identified this as early as 2005 [4], arising from...
issues appearing within their armed services during the Iraq and Afghanistan scenarios.

**Figure 1 - Current example of UAS operations [3]**

The cause lies mainly in the development method of UASs. Traditionally, manufacturers have focused on vehicle-centric designs and have developed the ground control component as a tool for flight-testing the airframe. This has led to a lack of standardisation and the use of proprietary telemetry and sensor data streams. The problem becomes larger when the military bodies add on top of that their own data links, security measures, etc. Hence, current UAS are “stove pipe” systems without the ability to interoperate with each other, as illustrated in Figure 1, with the Air Vehicle (AV) communicating with his own command (C4I), through separate datalinks and control systems (UCS).

NATO, through the STANAG document, specifies the interfaces needed to achieve a certain Level of Interoperability, which is used to classify UAS in operations:

- **Level 1** – indirect receipt of UAS related data
- **Level 2** – direct receipt of intelligence, surveillance or other data
- **Level 3** – Control and monitoring of the UAS payload
- **Level 4** – Control and monitoring of the entire UAS, less launch and recovery
- **Level 5** – Full control and monitoring of the UAS

Regarding the interoperability of unmanned vehicles in civilian applications, the effort is driven by the military as it happens in other technology domains. GEMA reviewed the literature in the regulatory framework for civilian use in non-segregated airspace; and the safety and legal issues due to civil aviation procedures have not yet been resolved by the civil aviation authorities. This means that the majority of unmanned operations take place in military environments, and those that do take place in civilian airspace are severely restricted, requiring usually special certificates of authorisation from their corresponding national bodies.

Therefore, the military domain is where the interoperability problem has been approached and progressed further. The STANAG 4586, without providing a complete solution for interoperability, is certainly a major step taken in that direction and provides a baseline roadmap for future developments. There are other efforts such the Joint Architecture for Unmanned Systems (JAUS), the ASTM Technical Committee F41 on Unmanned Undersea Vehicle Systems or MIL-STD-1760 Aircraft/Store Electrical Interconnection System. However, the architecture and guidelines defined in STANAG could be adapted to the civilian domain. GEMA focused the study on this standard, whose approach could be applied to any sort of operations, and in fact, both scenarios simulated in this paper are based on possible domestic UAS operations such as search and rescue.

Finally, integration of UAS with ATM services has not been an objective of this study, since this is addressed in SESAR [5], as well as past and current projects or initiatives such as INOUI [6]. Certainly, the lessons learned in the military will serve as a source of input towards the successful integration of these vehicles.

**FLEET MANAGEMENT INTEROPERABILITY BASELINE**

GEMA studies the following key areas – that would be part of an operational system –, and analyses the required technology push to achieve interoperable fleet management. These areas also have an impact on system automation and interoperability levels.

- **Mission Control**: component of the ground control segment commanding the fleet and assuring communication among system entities.
- **Mission Planning**: component of the ground control segment defining the Mission Plan (MP) and its conversion into precise instructions, which shall be executed by each of the fleet elements (i.e. operators, aircrafts and payloads).
- **Flight Segment**: airborne component executing the MP through actions commanded by the mission control; composed of vehicles, on-board logic, payload and data links. Additionally, the flight segment would require a certain degree of “intelligence” augmenting its endurance and robustness when confronted with unplanned situations.

**Interoperable Mission Control Architecture**

NATO STANAG 4586 specifies the basis for an Unmanned aircraft Control System (UCS) designed to preserve interoperability with other systems. To achieve such purpose, the standard defines the following principal components:

- **Core UCS (CUCS)** - represents the aircraft’s control system as manufactured by the provider. The UCS includes the human-computer interface (HCI), allowing the operator to manoeuvre the UAS.
- **Data Link Interface (DLI)** - a standard data interface between the CUCS and the aircraft.
- **Command and Control Interface (CCI)** – standard communication interface between the CUCS and the UAS control centre.
- **Vehicle Specific Module (VSM)** – module in which communication protocols and formats are included. At the same time, the VSM translate the DLI formats to the specific language of the aircraft.
- **Command and Control Interface Specific Module (CCISM)** – translates the information from the UAS command centre or other command structures to readable data for the CUCS in case of incompatibility.

### Interoperable Fleet Mission Control Architecture

GEMA analyses and proposes an adjustment to this reference architecture with the objective to extend it to the management of multiple UCSs, in order to carry out operations as an integrated fleet of heterogeneous vehicles; with the following key changes (depicted in Figure 2):

- Implementation of a **Master UCS (MUCS)**, a high-level centralised control system designed to interact with legacy and future UCS. MUCS allows the operator to generate flight plans; taking into account mission priorities, specifying Figures of Merit and establishing safety rules; and fusing this information with the characteristics of available assets and the environment of operation. Interoperability with CCI is guaranteed through the use of STANAG 4586-compliant message structures and communications.

- Interconnection of military (and civilian) command centres to the MUCS using standard interfaces (based on the NATO Common Standards Profile protocols), with secure networking. The objective is to depict UASs and other physical and logical systems in the theatre of operations as nodes of the same network.

- UAS control intelligence is part of the MUCS transforming high-level MPs into specific flight parameters for their corresponding vehicle.

Due to these modifications, GEMA introduces the figure of the **Fleet Operator**, controlling the flight grid of the fleet and the flight pattern of each UAS, allowing a more overall view of the mission objectives and supervising any contingency procedures. The Pilot-in-Command remains in command, but interacts with the Fleet Operator to ensure a maximum level of coordination and safe reaction to unforeseen events.

On the other hand, the Fleet Operator would become a supervisor of the mission, and the controller with the interfaces of external system entities such as C4I systems or ATM. The Operator would also manage the data distribution coming from the vehicles’ payload. This system’s role, with the automation mechanisms added by using an standard, would facilitate the Pilot-in-Command’s workload and overall mission coordination and safety. The scope of GEMA has not focused on the human-system interaction and its consequences of such task migration shift, but it introduces interesting questions regarding future research opportunities.

![Figure 2 - GEMA Mission Control Architecture](image)

### FRAMEWORK FOR STRATEGIC PLANNING

#### Mission Planning Tool

Planning a mission for a number of UASs involves considering several factors such as route selection, payload planning, contingency measures and so forth. To preserve the interoperability, the mission planning tool implemented in GEMA, converts the message structure specified in STANAG 4586 and uses it to develop an application covering these essential characteristics. The tool generates as output a MP compatible with any vehicle following NATO guidelines. Regarding input sources, the MUCS communicates with the existing ground stations in order to form a database of available assets and their options regarding payload configurations. These assets have been predefined in order to validate the tool in a simulation environment.

Five types of vehicles were defined so as to cover adequately the current spectrum of Unmanned Aerial Vehicle (UAV) technology present in Spanish forces: Heron (medium altitude, long endurance), Searcher (recon platform), Fulmar Aerovision (private-owned, small size UAV), SIVA (R&D demonstrator) and Camcopter (vertical take-off and landing). Once a vehicle has been selected for a mission, the fleet operator may also choose its payload based on several options defined in STANAG document. The fleet operator also composes the flight path selecting a series of waypoints, by means of a visual interface based on Google Earth technology. Each waypoint has the option to associate it an action like a predefined movement pattern, a payload-related instruction or a subsystem activation.

Once the operator has successfully designed the route of each vehicle with their corresponding actions, the application outputs a XML file containing all flight plans in a STANAG format, and available to be distributed within the network and uploaded to any UAS.
Performance Analysis Tool for Inter-vehicular Communications

The flight segment executes a mission defined by the ground segment’s planning tool, after the successful mission definition undertaken by the fleet operator. The objective of GEMA within the flight segment context, is to study state-of-the-art schemes to improve the operability – and therefore enhance system automatization –, through the exploitation of inter-vehicular communications. Facilitating the C²I and payload data exchange among vehicles and directly at flight segment extends the capabilities of the mission such as,

- larger mission coverage, e.g. the use of inter-vehicular data links to forward ground commands;
- a longer mission endurance, e.g. releasing ground links from unnecessary data exchanges saves resources such as power, bandwidth, etc.;
- an increase of safety, e.g. redundant data communications with ground;
- an increase of autonomy and flight performance, e.g. possibility to exchange updated and real-time guidance, navigation and control (GNC) data among vehicles to autonomously adapt trajectories with less ground intervention and improve overall fleet coordination; or,
- a cost reduction of the ground segment operations with a formal definition of the autonomy levels, e.g. alleviates the inherent scalability problem resulting from the increase in the number of UAS to be controlled simultaneously with the number of Pilot-in-Command operators.

At strategic planning stage and considering the need for inter-vehicular data communications; flight plans – output of mission planning tool – need to be validated in terms of performance and viability of generated mission trajectories. For such purpose, a tool has been implemented to analyse the network performance. This application simulates the network data traffic and reports its performance metrics (throughput, latency, routing load, etc.). Additionally, the trajectories are displayed on a terrain map of the mission area with a set of common navigation functions available to the fleet operator, such as play, fast forward and rewind. Network topology evolution, vehicle flight dynamics and data traffic patterns are considered input parameters to the simulation model.

An aerial Mobile Ad-hoc Network (MANET) has been proposed for the management of the data exchange among all vehicles in the fleet. This type of self-organized network does not require any existing communications infrastructure to operate, thus it becomes especially attractive for challenged environments or isolated areas where a fast deployment and operation of autonomous teams is essential. Every node (i.e. UAS) in a MANET acts as a router forwarding information to the destination node as a communication relay. Hence, finding an appropriate route to establish end-to-end connectivity between two UAS in the fleet becomes crucial. A performance evaluation of the two most relevant MANET approaches for path discovery (namely, reactive and proactive) has been carried out by means of simulation of two well-known routing protocols: Ad-hoc On-demand Distance Vector (AODV) and Destination Sequenced Distance Vector (DSDV). Proactive protocols continuously update and broadcast routing information across the network, thus making it available to all nodes at all times. On the other hand, reactive protocol does not generate routing traffic as long as it does not exist an actual transmission request by the nodes. The main advantage of proactive routing is the permanent availability of routes to all nodes, although that implies a considerable routing overhead and unnecessary control messaging. Conversely, reactive protocols are more agile in the sense that routes are calculated on demand but they increase the end-to-end transmission delay introduced by the route discovery processes. A performance evaluation of these complementary routing approaches will allow an a priori estimation of the adequacy of each alternative for the mission profiles defined herein. Several network configurations have been generated to emulate different possible contingencies that might occur in real fleet operations.

SIMULATIONS AND EVALUATION OF TOOLS

We first reviewed several classifications of UAS applications available in the literature, selecting for the purpose of GEMA, Maritime Surveillance and Search and Rescue. The mission characteristics of these two scenarios provide a representative framework to test and evaluate the tools and the concept for commissioning the mission at strategic planning level.

The objectives of Maritime Surveillance scenario were to emulate large area coverage and a regular flight pattern. This mission was defined with five vehicles in the area of the Strait of Gibraltar. The vehicles flying in V-shaped formation repeatedly scanned the surveillance area and were supported by two additional rotor aircrafts hovering at a lower altitude. This scenario is intended to analyse the behaviour of the network in high topology variation conditions. The relative movement of the aircrafts and the possible gaps created in the formations allowed us to test the ability of the candidate protocols to cope with these challenges. On the contrary, the objectives of Search and Rescue scenario were to emulate small area coverage as well as use one of the three vehicles in an even more reduced area as communication relay. The selected area was the surroundings of a ski station located in the Pyrenees. In this case, the network topology remains static during most of the mission time but a certain deviation of one of the aircrafts with respect to its nominal trajectory has been induced. Then, the network is forced to coordinate the safe return of the deviated aircraft by means of inter-vehicular data exchanges towards the leader of the formation.

Both scenarios were modelled with the Mission Planning Tool, designing each flight plan according to STANAG specifications (Figure 3). Each generated mission plan was then fed into the Performance Analysis Tool for
Inter-vehicular Communications, to assess the performance of different routing protocols for each scenario. The absence of a de-facto standard protocol for these types of networks highlights the relevance of this study, given that it will aid the a priori selection of the most suitable protocol for the mission. In particular, proactive and reactive routing approaches were compared in [7] for the different data traffic patterns and flight plans in order to identify the best solution. Figure 4 shows an example of the tool with the visualisation, time navigation, and metrics components.

Simulations showed that there is not an optimum candidate for all scenarios. AODV has a better performance in high topology variation scenarios whereas it shows a greater latency that might make real time communications difficult. Likewise, DSDV offers a good performance in the Search and Rescue scenario because of its static topology. However, the large volume of routing overhead generated is excessive for example, for sporadic transmissions of small size files. Then it is necessary to analyze in a case-by-case basis the appropriateness of each routing approach for every scenario. To that end, the performance evaluation carried out constitutes a valuable tool to find out the most suitable solution depending on the specific constraints of the mission.

Additionally, it has been observed that MANET protocols fail to establish communications when the destination node cannot be reached. This situation can be prompted by a link failure, a temporary disconnection of the communication equipment or an obstacle in the line of sight. Hence, the introduction of Disruption Tolerant Network (DTN) mechanisms should be considered in future works, in case we wish to prioritize the integrity of the communications and avoid data losses.

CONCLUSIONS AND RECOMMENDATIONS

Interoperability in UAS still remains a challenge, offering nowadays merely basic payload control and operation. Additional levels of interoperability are not yet attainable even when the same aircraft is operated by different command centres from different countries. One of the challenges when tackling this issue is the applicability of standards or guidelines to implement C1 (sub)systems and components. In GEMA project, we have approached the implementation of UAS fleet management at strategic planning level, using compliant C1 NATO standards easing its future integration with existing UAS vendor solutions or ground control centres.

This implementation – evaluated with a Maritime Surveillance and a Search and Rescue scenarios –, includes a Mission Planning Tool and a Performance Analysis Tool for Inter-vehicular Communications. The later analyses the performances of inter-vehicular data exchange with a given mission plan. This would first imply, in an actual scenario, the availability of that data links at flight segment level among UASs; and second an extension of current ground-air definitions of standard to air-to-air, e.g. VSM or DLI. With the possibility of such complex missions, the automation levels could be increased for fleet management, transforming the current role of the Pilot-in-command into a fleet operator role, responsible for a mission which includes multiple heterogeneous aircraft, regardless of their level of autonomy.

ACKNOWLEDGMENTS

This work has been performed under a grant of the Spanish funding agency entitled Centro para el Desarrollo Tecnológico Industrial (CDTI), under the Aerospace Subprogram. The GEMA consortium includes Deimos Space, CTAE and INDRA.

REFERENCES

3. NATO STANAG 4586, Standard Interface of the Unmanned Control System (UCS) for NATO UAV interoperability.
5. Dalinchampt, M., Overview of the EUROCONTROL UAS ATM Integration Activity, 3rd Air-TN Forum CIR4 (Capua IT, February 2009).
Figure 3 – Graphical representation of Mission #2 in the Mission Planning Tool

Figure 4 - Performance Analysis Tool for Inter-vehicular Communications
A Possible Solution to Introduce UAVs Into Non Segregated Areas

Sergio Taraglio, Vincenzo Nanni
Robotics Lab, ENEA
Via Anguillares 301
Rome, 00123 Italy
taraglio, nanni@enea.it

Damiano Taurino
Deep Blue s.r.l.
Piazza Buenos Aires 20
Rome, 00198 Italy
damiano.taurino@dblue.it

ABSTRACT
Presently Unmanned Aerial Vehicles (UAV) are flown in segregated areas, but it is foreseen that in the near future their operation will be conducted in any airspace. This will in turn boost the number of UAV civil applications and the related market. Some results of simulated flight trials of UAVs in segregated and non segregated areas are here presented. The management of such flights is based on a distributed algorithm grounded on Game Theory which supplies maneuvers aimed at providing separation distance among aircraft. These experimental data show that this algorithm, previously proposed by the authors, is able to yield separation assurance at a reasonable cost, in terms of flight timings and deviation from optimal trajectory.

Keywords
Game Theory, Multi-Agent Systems, Conflict Detection and Resolution, Separation Assurance, Unmanned Aerial Vehicles

INTRODUCTION
The use of Unmanned Aerial Vehicles (UAV) is a major challenge for the aviation in the near future. Extensive use of UAVs is a reality in military operations in the Middle East, even if in tele-controlled mode [1]. On the contrary the use of UAVs in civil environments is still limited to restricted airspaces where they cannot mix with civilian aircraft. Nevertheless the market of these applications is forecasted to be quickly surging to a very wide one. Typical UAV applications are airborne inspections and monitoring of any kind, but in a simpler and more flexible way and at an extremely less expensive cost as compared to manned missions. Besides there are dangerous or monotonous missions where unmanned operations are by far preferable. UAVs can observe natural and man-made disasters from above, inspect pipelines or aerial power lines, monitor pollution or highways, help in enforcing law, etc.. It is therefore evident that in a near future scenario UAVs and commercial aircraft will have to share the same airspace. This implies the rise of several problems, safety as the most relevant.

The UAV flight can be controlled in different ways, from remote to fully autonomous control through waypoints and autopilot modules, see e.g. [2][3].

Autonomy can be defined as the ability to take decisions without human intervention. From this perspective, most UAVs are not autonomous at all, autonomy is a recently emerging field. Compared to the manufacturing of UAV flight hardware, the market for autonomy technology is fairly immature and still undeveloped. Because of this autonomy has been, and may continue to be, one of the bottlenecks for future UAV developments, as it has been pointed out in [4], a very interesting survey among UAV stakeholders. The overall value and rate of expansion of the future UAV market for scientific and civil usage will be largely driven by advances in the field of autonomy.

The key issue for the introduction of UAVs in non segregated airspaces is represented by the ability to “keep at a distance” from all the other flying vehicles in the neighborhood (separation distance assurance and collision avoidance), regardless if they are UAVs or commercial aircraft.

In previous work [5][6] a game theory approach to the solution for the problem of separation assurance has been proposed: the ARCA algorithm (Adaptive Routing and Conflict mAnagement for UAVs). This algorithm can be considered as an industrial asset capable to greatly enhance the market appeal of an UAV. A vehicle equipped with such an algorithm will be able to be autonomously flown in any airspace, avoiding air traffic problems. The ARCA module implementation in an additional processor cooperating with the actual vehicle autopilot, is presently in progress within the ARCA EUROSTARS project.

The ARCA algorithm must not be thought as a sense and avoid oriented algorithm, but it is conceived as a mid-term approach to separation assurance. This means that it operates in order to foresee possible too close distances, keeping the UAV separated from all other traffic. Nevertheless the ARCA algorithm must be considered as one of various safety components of a UAV architecture, which will also possess a collision avoidance system such as the TCAS (Traffic alert and Collision Avoidance System). In the following the effort is limited to the analysis of the capability of the ARCA algorithm to assure separation from other traffic, leaving integration issues with other systems designed to guarantee safety.
(the so called safety nets) to a later stage of system implementation.

In this paper some experimental results of the ARCA algorithm are presented. The simulation of a double set of flight trials, as a necessary step before a future set of actual ones, is performed. On one side the algorithm is tested with physically simulated UAVs in segregated space in order to validate the capabilities of the algorithm. On the other the same experiments are performed in a non segregated airspace. This is accomplished taking into account the real time flight data incoming from the commercial aircraft in the area. These data are collected decoding the ADS-B messages broadcasted by the various civilian aircraft [7]. Through the use of these data the simulated UAVs will be able to avoid interaction with the actual traffic.

In the first section the algorithm is briefly reviewed. In the second section the experimental setup is described, complete of the overall software architecture, the aircraft physical simulator and the ADS-B parser. In the third section the experimental results are presented. In the fourth section the conclusions are drawn.

THE ARCA ALGORITHM

The ARCA algorithm is inspired by Game Theory and in particular it is based on Satisficing Game Theory approach (SGT). [8]

Basically, SGT computes the best choice among various alternatives by means of two utility functions, selectability and rejectability, which respectively represent benefits and costs for each agent while making a choice. There may exist dependence between utility functions of different agents that implements an “altruistic” behaviour.

This approach has already proven to be effective in dealing with several real-world problems (e.g. packet routing [9] and [10], transport logistics [11], automated car driving [8], airborne self-separation [12]) by providing robust and dependable solutions that can be achieved with limited computational resources.

In the UAV context each UAV can be considered as an agent and SGT can be used as follows. Selectability and rejectability of each agent represent the benefits and costs of the UAV’s maneuvering choices. Benefits are essentially proportional to the optimality of the possible route to reach the final destination. Costs are proportional to the risk of collision with other vehicles. Each aircraft computes the rejectability and selectability of each maneuver, then selects the directional option that maximises the difference between the selectability and rejectability utilities. Loosely speaking, each aircraft selects the best path with respect to the minimum risk.

At each time step, each aircraft exchanges information with all other aircraft within its communication range. This information includes: current position, destination, actual heading, flight time (basically the information of an ADS-B communication frame). Each aircraft chooses one of five directional options: flying straight, moderate turn to the left, sharp turn to the left, moderate turn to the right, sharp turn to the right. The length of each time step is an algorithm parameter and can be changed according to necessity. In our case it has been used the time step of the ADS-B protocol (half a second).

For each aircraft a priority set is defined as the set of all viewable aircraft with higher ranking than the considered one that could conflict with the aircraft itself (the parents) for some heading choices. The rank may be assigned using different criteria, in this case it is based on the delay each aircraft has accumulated during its past flight.

Rejectability of agents is unconditioned: each aircraft responds to threats with exclusive self-interest by comparing the linear extension of each of its directional options with linear projections of current headings of all aircraft in its priority set. Each projected conflicts adds a weight to the rejectability function related to the directional option, depending on distance in time and severity of the conflict. After all the parents are considered, the weight of each option is normalized over the option space. These rules increase the rejectability of flight options that lead to conflicts or small separations, with more weight for incidents closest in time.

The selectability function reflects goal achievement. Differently from rejectability, selectability is influenced by the preferences of other agents. Thus, we have two distinct components for the selectability: the base selectability ($\sigma_{B}$) that accounts for the considered aircraft heading preferences and the parent selectability ($\sigma_{P}$) that accounts for higher priority agents preferences.

Thus, the selectability function $p_{S}$ for a given direction $u_{i}$ is formed by the convex combination:

$$p_{S}(u_{i}) = \lambda \sigma_{B} + (1 - \lambda) \sigma_{P}$$

with $\lambda = 1$ if the priority set is empty, otherwise $\lambda = 0.001$. In non restricted airspace commercial aircraft have always higher priority than UAVs. This rule is introduced for safety purposes.

After computing rejectability and selectability, each agent (aircraft) has to choose the heading change to perform. In order to do that, each aircraft selects the heading option ($u'_{i}$) that maximize the difference between the selectability ($p_{S}$) and rejectability ($p_{R}$) utilities:

$$u'_{i} = \arg\max_{u} (p_{S}(u_{i}) - p_{R}(u_{i}))$$

Thus, each satisficing agent is looking for the highest gain, with the lowest risk, taking also into account preferences of other agents, thus obtaining a solution that could be effective for the whole system.

In Figure 1 the algorithm flowchart is provided for the algorithm in order to better explain its different phases.

In the present work the ARCA algorithm has been simplified. The reason behind this is represented by the need to implement it onto a physical processor to be actually integrated on board an UAV. The chosen processor is already tested and cleared for avionic purposes, but does not offer a too brilliant computing performance (approximately 125 Dhrystone MIPS, similar toa Pentium 90 PC). The algorithm has thus been
lightened of the part concerning the computation of the parent selectability. This means that each agent does not take into account the flight preferences of the parent aircraft, i.e. it does not consider the parent aircraft flight destination. It is important to stress that presently the ADS-B message broadcasted by aircraft does not yet include this information, in this sense the simplified assumption of this version of the algorithm is more realistic. In the ADS-B standard the flight destination is already available among the different message fields for future use, and the ARCA algorithm may naturally exploit it as soon as the aircraft will broadcast it.

In the following the simplified algorithm performance are presented.

**ARCA algorithm performance – cinematic aircraft**

Here the performance of the ARCA algorithm, as measured in numerical simulations, is presented. In these initial tests **cinematic** aircraft have been used, i.e. ideal vehicles moving at a constant speed and turning on the spot without considering any dynamical effect.

The results shown in Table 1 and Table 2 are respectively relative to a **choke point** scenario and to a **random flight** scenario [6].

In the first scenario all aircraft start on a circle of radius 60 nautical miles and head directly through the circle centre towards the opposite side. The aircraft speed is equal for all and set at 150 knots, statistics is on 30 trials.

<table>
<thead>
<tr>
<th>UAVs number</th>
<th>Missed Separations</th>
<th>System Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Std.Dev.</td>
</tr>
<tr>
<td>3</td>
<td>0,000</td>
<td>0,000</td>
</tr>
<tr>
<td>5</td>
<td>0,200</td>
<td>0,551</td>
</tr>
<tr>
<td>10</td>
<td>0,233</td>
<td>0,430</td>
</tr>
<tr>
<td>15</td>
<td>0,933</td>
<td>0,868</td>
</tr>
</tbody>
</table>

Table 1. Choke point scenario.

In the second scenario the given number of UAVs are randomly flown in a square of side 120 nautical miles with a speed of 150 knots.

<table>
<thead>
<tr>
<th>UAVs number</th>
<th>Missed Separations</th>
<th>System Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Std.Dev.</td>
</tr>
<tr>
<td>3</td>
<td>0,000</td>
<td>0,000</td>
</tr>
<tr>
<td>5</td>
<td>0,033</td>
<td>0,183</td>
</tr>
<tr>
<td>10</td>
<td>0,267</td>
<td>0,450</td>
</tr>
<tr>
<td>15</td>
<td>0,700</td>
<td>0,651</td>
</tr>
</tbody>
</table>

Table 2. Random flight scenario.

The above two scenarios are well known hard challenges to any separation assurance or collision avoidance method. Nevertheless they are far from any real operational scenario.

The **system efficiency** is here computed in terms of flight time,

\[
SE = \frac{1}{N} \sum_{i=1}^{N} \frac{t_i}{t_i + t_d}
\]

as the average over the number of UAVs of the ratio between ideal flight time (no detour) and ideal flight time plus delay time introduced by the aircraft maneuvers. Thus maximum efficiency is 1.

**THE EXPERIMENTAL SETUP**

The ARCA algorithm can be considered as an **add-on** for the vehicle autopilot. The autopilot computes the route towards the next waypoint and notifies the ARCA module which, in turn, checks cyclically the danger associated with the present UAV route and possible deviations, taking into account the air traffic in the neighborhood. In order to avoid possible separation miss it can “steer” the UAV.

In the following experiments the ARCA algorithm has been fused with the UAV autopilot. This means that the UAVs are made fly to a single destination waypoint to be reached via a direct route. Therefore the ARCA algorithm directly acts as an autopilot computing new routes if the need arises.

The overall software architecture of the simulation system is shown in Figure 2. The links among the different subsystems are differently colored in order to point out their genre: html, socket or variable data.

![Figure 2. The software architecture of the simulation system](image-url)
The simulation scenario is graphically represented via an html page on the Web Server that reads the flight data from a KML (Keyhole Markup Language) file composed by the Data Server. The ARCA algorithm accesses the various UAVs simulators, one for each UAV, with bi-directional socket links and receives data from the ADS-B Parser in order to take into account the commercial aircraft routes.

Each UAV involved in the experiments has been physically simulated. The simulator is based on the JSBSim aircraft simulator which is an Open Source flight dynamics model that can be run under several operating systems [13].

The airspace where the experiments have been performed has been graphically displayed through the use of an interface based on the Google Earth Plug-in. The geographical area used is that of central Italy, above Rome.

The ADS-B messages are usually broadcasted by all flying commercial aircraft specifying several information concerning the aircraft itself such as position, heading, speed, altitude, ID, etc. A parser has been implemented on the SBS-1 system of the Kinetic Avionic [14] that is able to process the ADS-B signals as collected via an antenna and decoded by the SBS-1 device. It outputs some aircraft relevant data (position, speed, heading, ID, time) to the ARCA algorithm for its purposes.

**SIMULATED EXPERIMENTAL FLIGHT TRIALS**

Present flight regulations don’t allow UAV flights outside segregated areas. This implies that flight trials involving UAVs in non segregated areas are very difficult to be actually performed, especially for civil applications. From this arises the need for simulated flight trials that are here presented.

In the first batch of experiments a set of two or four UAVs are considered in a segregated space, this means that the ADS-B flight data from commercial aircraft are not considered. Two different experiments are presented. In the first two UAVs are made fly one against the other specifying the starting point of each one as the destination point of the other. In the second four UAVs are placed at the four corners of a square and made fly towards the facing corner, i.e. passing all at the same time in the centre of the square.

In the second batch of experiments the UAVs perform the same flight paths but in a non segregated airspace. This means that the ARCA algorithm takes into account also the ADS-B messages from commercial traffic in the area and not only the ones from the UAVs. The aim of these experiments is to show that the algorithm is able to avoid separation infringements for the UAVs both against each other and with commercial flights.

**UAVs only (segregated space) – physical aircraft**

The two performed experiments, i.e. with two or four physically simulated UAVs, are essentially similar to the ones reported in Table 1 and 2. The main differences are the initial configuration and the physical simulation of the aircraft: here a full dynamical simulation of all the vehicles is performed.

In the case of two vehicles one of the two has a right of way over the other, i.e. a higher priority. In the case of four a ranking in priority is set up.

No ADS-B data from the actual air traffic is taken into account. Thus the simulation can be considered as in a segregated airspace.

<table>
<thead>
<tr>
<th>UAVs number</th>
<th>Missed Separations Mean</th>
<th>Std.Dev.</th>
<th>System Efficiency Mean</th>
<th>Std.Dev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0,000</td>
<td>0,000</td>
<td>0,942</td>
<td>0,039</td>
</tr>
<tr>
<td>4</td>
<td>0,000</td>
<td>0,000</td>
<td>0,945</td>
<td>0,049</td>
</tr>
</tbody>
</table>

Table 3 Segregated space physical UAVs only results

In Table 3 results are presented, the data are relative to 30 trials for each experiment. As it can be seen the algorithm is able to successfully separate the UAVs with a very limited deterioration in system efficiency, see also Figure 3. The algorithm is able to produce near optimal routes while avoiding other aircraft. Later in the paper a brief discussion on the aspect of algorithm system efficiency will be given.

**UAVs and actual commercial traffic (non segregated)**

Also in this case two experiments have been performed, with the same characteristics of the segregated space ones: two UAVs have been flown one against the other and four of them from the vertices of a square. The main, and key, difference is represented by the presence of the surrounding commercial traffic. The considered commercial aircraft obviously possess a much higher priority than any of the UAVs in order to force the UAVs to maneuver and to leave the commercial aircraft routes unchanged. Then, as in the preceding experiments, a rank in priority is given to the UAVs.

The commercial traffic near the Rome Fiumicino airport is always quite dense, with several aircraft flying at different altitudes, with approaching, landing and departing aircraft together with cruising altitude flights.

The ADS-B antenna used is able to pick an average number of about 15 to 20 aircraft at a time. The problem
is that these aircraft may be intermittently seen by the system depending on the antenna sensitivity and on their distance from the antenna. If a contact is lost for a given period of time the relative aircraft is cancelled from the system. But if a new contact is made a new aircraft will pop up and must be taken into consideration by the ARCA algorithm. From this stems a problem since the aircraft may appear very near to one of the simulated UAVs, with a velocity usually several times greater than the UAV one. This often implies an unavoidable separation miss and the failure of the algorithm.

This event can be taken into account disregarding the near misses if the life time of the newly born aircraft is smaller than a given threshold. This is clearly a shortcut to overcome the problem which is due to the physical location of the ADS-B receiver during the simulation. In real flight condition the ADS-B antenna is on board the UAV, allowing the full awareness of the other aircraft in the neighborhood. Studies taking into account loss or degradation of the data link will be performed in the future, in order to provide a more reliable approach to ADS-B communication.

In our experiments all the received aircraft are virtually transposed to the same flight level in which the UAVs are flying and climb or descend maneuvers are not allowed in order to stress the algorithm with a much denser traffic.

A visual idea of the area and the traffic involved is presented in Figure 4. Here it can be seen the user graphical display based on the Google Earth plug-in in the neighborhood of the Fiumicino Rome airport. The light blue aircraft are the commercial ones, while the red are the simulated UAVs, the circle surrounding each vehicle has a 2.5 nautical miles radius: if two circles don’t intersect, the separation is assured.

Figure 4. High density traffic in the Rome area.

In these experiments, the average traffic of commercial flights has been of 11 aircraft in the area. This number is quite different from the situation shown in Figure 4, this because a threshold in the actual aircraft altitude has been applied before their virtual placement at the same altitude of the simulated UAVs.

The results of the full traffic simulation show that there are an average number of separation infringements equal to 1.33 at the cost of a large degradation in the UAV system efficiency, the average value is 0.775. It must be bore in mind that in these simulations all the traffic is artificially considered at the same altitude in order to stress the algorithm and in order to guarantee the existence of several conflicts among the UAVs and the civilian traffic.

In order to simulate a more realistic scenario, we have only considered those aircraft flying at very high altitude, i.e. those flying at cruise speed and altitude (runway crossing scenario). The number of tests performed is three employing four simulated UAVs. During these experiments it has been recorded an average number of civilian aircraft of 3.5. In this case the average number of conflicts reduce to zero with an average UAV system efficiency of 0.912.

CONCLUSIONS

In this work several simulated flight trials experiments of mixed UAVs and commercial aircraft traffic have been presented. These experiments show that the devised algorithm can provide acceptable performance in ensuring separation and trajectory efficiency while introducing UAVs in simulated non segregated airspace.

Such an algorithm installed aboard an UAV will increase its industrial and commercial worth facilitating its integration within the air traffic.

Naturally the ARCA algorithm must not be thought as a substitute for a short term collision avoidance system (as TCAS), but as a cooperating agent for the safe guidance of an unmanned aircraft. It has been designed as a tool for mid-term separation assurance, to be integrated on board a UAV. It is the outcome of an ongoing industrial research project, aimed at the realization of a physical prototype.

Under the aspect of system efficiency, it must be noted that the ARCA algorithm is a hierarchical one. It is well known that the symmetrical cooperative algorithms for separation assurances give near optimal solutions in term of global efficiency. This algorithm uses a ranking in priority which reflects the real world. For example in this case the civilian aircraft must possess a higher priority than the UAVs. In more general terms, the ARCA algorithm (in its full version) features a dynamical computation of the priority ranking based on different considerations such as delays, fuel consumption, closeness to arrival, weather conditions, etc. The ARCA algorithm is a self contained approach with no need for information exchange among the various agents in the scene, but only the passive collection of ADS-B flight data.
Obviously a deeper analysis of safety related issues is necessary, nonetheless the results here presented show that this solution is following a promising path to industrial applications.

Naturally the future success of this kind of technology greatly depends on the laws and regulations that will be adopted by the involved institutions in the next few years.

ACKNOWLEDGMENTS

This work has been developed within the Adaptive Routing and Conflict Management for UAVs (ARCA) project, funded by the EUROSTARS Programme of the European Commission.

REFERENCES


