Artificial Intelligence for Engineering Design, Analysis and Manufacturing (2017), *31*, 69–86.
© Cambridge University Press 2016 0890-0604/16
doi:10.1017/S0890060416000184

Development of flexible languages for scenario and team description in multirobot missions

DANIEL CASTRO SILVA,[1,2] PEDRO HENRIQUES ABREU,[3,4] LUÍS PAULO REIS,[2,5] AND EUGÉNIO OLIVEIRA[1,2]

[1]Department of Informatics Engineering, Faculty of Engineering, University of Porto, Porto, Portugal
[2]Artificial Intelligence and Computer Science Laboratory, Faculty of Engineering, University of Porto, Porto, Portugal
[3]Department of Informatics Engineering, University of Coimbra, Coimbra, Portugal
[4]Centre for Informatics and Systems, University of Coimbra, Coimbra, Portugal
[5]School of Engineering, University of Minho, Campus de Azurém, Guimarães, Portugal

(Received June 6, 2015; Accepted February 8, 2016)

Abstract

The work described in this paper is part of the development of a framework to support the joint execution of cooperative missions by a group of vehicles, in a simulated, augmented, or real environment. Such a framework brings forward the need for formal languages in which to specify the vehicles that compose a team, the scenario in which they will operate, and the mission to be performed. This paper introduces the Scenario Description Language (SDL) and the Team Description Language (TDL), two Extensible Markup Language based dialects that compose the static components necessary for representing scenario and mission knowledge. SDL provides a specification of physical scenario and global operational constraints, while TDL defines the team of vehicles, as well as team-specific operational restrictions. The dialects were defined using Extensible Markup Language schemas, with all required information being integrated in the definitions. An interface was developed and incorporated into the framework, allowing for the creation and edition of SDL and TDL files. Once the information is specified, it can be used in the framework, thus facilitating environment and team specification and deployment. A survey answered by practitioners and researchers shows that the satisfaction with SDL+TDL is elevated (the overall evaluation of SDL+TDL achieved a score of 4 out of 5, with 81%/78.6% of the answers $\geq$4); in addition, the usability of the interface was evaluated, achieving a score of 86.7 in the System Usability Scale survey. These results imply that SDL+TDL is flexible enough to represent scenarios and teams, through a user-friendly interface.

Keywords: Extensible Markup Language Dialect; Knowledge Representation; Multivehicle Mission; Scenario Definition; Team Definition

1. INTRODUCTION

In the development of a new software system, one of the fundamental initial decisions pertains to information storage and representation. System architects are faced with a decision that can have a perceivable impact in the manner in which the whole system will function.

Information can be stored in different formats and supports, from simple text files to complex database systems. Text files can contain data in a closed, proprietary format or in an open format. A markup language allows for structure (hierarchical mainly) to be easily represented, and being a self-documenting format, it provides not only data but also metadata. Current technological advances (mostly regarding processing and memory capabilities) make the two major disadvantages of using such formats (larger resulting files and processing costs) only minor disadvantages, which can, most of the times, even be overlooked. One of the most popular markup languages is Extensible Markup Language (XML; for more information see http://www.w3.org/XML/); since its inception, it has been adapted to a wide range of applications in a large number of areas, including medical (Schweiger et al., 2005; Kumar et al., 2009), multimedia (Deursen et al., 2007), corporate (ANSI/AIIM, 2009), military (Hobbs, 2003; Wittman, 2009), and other well-known applications (Groppe et al., 2009; Georgieva & Georgiev, 2010).

This problem presented itself in the context of the development of a framework to support the simulation of multivehicle missions, and brought forward the need to specify languages that would allow for the configuration of several aspects of the simulation. The details of this framework, which

Reprint requests to: Daniel Castro Silva, Department of Informatics Engineering, Faculty of Engineering, University of Porto, Rua Dr. Roberto Frias s/n, 4200-465 Porto, Portugal. E-mail: dcs@fe.up.pt

includes a realistic visual simulator, agents representing traffic controller entities and vehicles, as well as several utilities, such as Logging and Monitoring tools, are presented in Section 3.

In addition to the development of the simulation framework itself, the authors intend to create the basis for standard languages that can be used for scenario, team, disturbance, and mission definition for similar platforms, or for applications that use teams of heterogeneous vehicles (the authors also envision its use in helping schedule operations in transportation or logistics contexts, for instance), in order to increase interoperability and facilitate the comparison of features from distinct systems.

These languages should be easily readable by machines and humans (simultaneously providing a fast learning period for humans) and independent of the system being used. They should be easily extended to include new concepts, optional parts, or alternative portions; validation should be performed easily to assure the integrity of the data; and several concepts must be included, representing multiple entities such as vehicles, sensors, controllers, areas, or operational bases (including airport, ground base, or port facilities).

The description of teams and missions was divided into two categories: static and dynamic. This paper focuses on the static category, the description of the world (scenario), and the description of the team(s). The static category includes the Scenario Description Language (SDL) and the Team Description Language (TDL). The dynamic category encompasses the Disturbance Description Language (DDL) and the Mission Description Language (MDL). DDL describes disturbances to the environment (such as a fire, a focus of pollution, a vehicle involved in illegal activities, a person in need of rescue, or others), their location, and behavior (Silva et al., 2015), and MDL presents a high-level description of a mission (such as searching for a fire or maintaining surveillance on the actions of a vehicle; Silva et al., 2014). While the languages in the static category can be applied to a larger number of platforms with diverse goals, the languages in the dynamic category are more targeted at platforms such as the one described in Section 3; therefore, SDL and TDL are presented separately from DDL and MDL. Table 1 shows the classification of the developed dialects in terms of being static or dynamic, as well as in terms of orientation to scenario or team.

In order to specify the scenario in which the teams of vehicles are to operate, SDL was envisioned, including a description of facilities that can be used (providing robotic agents with the necessary knowledge to safely navigate through the known part of the environment), global environment restrictions, and some global control structures. Regarding the teams, TDL was created to include a description of each team, specific team restrictions, and a description of all vehicles that compose a team, their characteristics, and the sensors and/or cargo they transport.

A simple example of the type of missions that can be simulated is prevention and combat of wildfire (Casbeer et al., 2006). The SDL file would contain all static environment elements, such as the description of the base of operations (including airport) that can be used by the team; possible air traffic controllers (with jurisdiction over the base of operations or the mission area); possible areas the aircraft cannot fly through; and the existing types of aircraft. The team can be defined (using TDL) as a set of aircraft, some capable of carrying large amounts of water or fire retardant and others equipped with sensors that allow them to detect a fire (such as video cameras, infrared, carbon dioxide concentration, or temperature sensors). The DDL file would specify the characteristics of the fire, such as location, size, growth, and motion patterns. The MDL file would specify a mission composed of two stages: in the first stage, the team should detect any existing fire (using the aircraft that possess the necessary sensors) in a given area; in the second stage, which should be triggered when a fire is detected, the aircraft carrying water or fire retardant should fly over the fire and drop their load, in order to put out the fire.

Table 1. *Classification of the languages*

	Scenario	Team
Static	SDL	TDL
Dynamic	DDL	MDL

Only aerial vehicles and airports are considered for simplicity throughout the rest of the paper. The description of ports and ground bases and the details for ground and water vehicles are considered in the complete specification of both SDL and TDL (Santos, 2010).

The rest of this paper is organized as follows. Section 2 covers related work and languages, while Section 3 introduces the developed multivehicle mission simulation framework. Sections 4 and 5 cover the details of SDL and TDL, respectively. Section 6 presents the implementation details of both SDL and TDL, and Section 7 details the conducted survey and presents the obtained results. Finally, Section 8 presents some conclusions about the developed work and some lines for future developments.

2. RELATED WORK

A standardized textual description of the layout of a given physical area in a format that can be easily read by a person is not always easy to find. However, several applications exist that show that same information in a graphical and intuitive manner.

One of the most notable languages used to describe geographical data is Geography Markup Language (GML), a standard developed by the Open Geospatial Consortium (OGC; more information is available online at http://www.opengeospatial.org/) to express diverse geographical features (OGC, 2007*a*). After several years of developments and changes, it was approved as an international standard in 2007 and is being used by several applications (Huang et al., 2009; more information is available at http://www.ogcnetwork.net/gml).

CityGML is one of the most well-known applications of GML (OGC, 2008; more information about CityGML is available at http://www.citygml.org/). It is a markup language used to describe three-dimensional urban objects and scenes. It was adopted as an OGC standard in August 2008 and is used in several applications (Fan et al., 2009).

The Aeronautical Information Exchange Model (see http://www.aixm.aero/) is an international data standard and a model that supports aeronautical information collection, dissemination, and transformation throughout the data chain in a digital format (Brunk & Porosnicu, 2004). It started as an initiative from Eurocontrol (the European Organization for the Safety of Air Navigation) over 12 years ago and later had the active participation of other entities, such as the US Federal Aviation Administration, International Civil Aviation Organization (ICAO), and North Atlantic Treaty Organization. It is currently in its fifth version, adopting GML as a basis, which provides numerous features for several entities involved in the aeronautical industry (Brunner et al., 2007).

One of the foremost applications that uses a description similar to part of the one described in this paper is flight simulators. Airport descriptions in flight simulators can be very detailed, allowing for a very realistic visual simulation. However, and for that reason, airport description tends to be mostly centered on visual aspects, such as signs, lights, lines, and many other aspects. Even though these aspects are very important for a visual simulation, they do not contribute much for the goals of this project and the objectives of the languages being developed.

One of the flight simulators analyzed in more detail was Microsoft's Flight Simulator X (FSX). Airport information is contained in precompiled files and features numerous visual aspects along with airport structure. Extensive documentation on the format and information contained within the files is provided, and an application to compile these files is included in the software development kit (along with a Schema for the XML definition of the format; Microsoft Corporation, 2008). Some tools have been developed by the community and software vendors to help interact with this simulator. One such example is Airport Facilitator X, a product that provides a visual interface to design and edit airports and their elements, such as runways, taxiways, towers, and parking spaces (Flight One Software, Inc., 2009). Another example is the Airport Design Editor, which provides similar functionalities (Masterson et al., 2009).

Other largely known simulators have different representations of such information. For instance, FlightGear uses a format that allows for a compact representation of airport runways and taxiways, by using a series of letter-based codes to represent enumerations of surface types, signaling, and lighting (Peel, 2001). However, some information could be better represented (e.g., taxiways are not described as a whole, but by individual segments). Another flight simulator, X-Plane, uses a very similar representation, for airport structures (Peel, 2009). The file specification is based on a series of codes (mostly numeric) and is more complete than the one from FlightGear. However, both representations are far from being easily read by a human without the aid of an interpreting application.

Most works found in the literature that deals with the definition of a team of vehicles focus on hierarchical or behavioral aspects, which are not represented here at the team level, but at the mission level, and thus fall outside the scope of this paper. Furthermore, most of these works do not express the vehicles that compose the team or their capabilities in an explicit form, but rather include that information in an ad hoc manner, in custom-developed formats, or even embedded within the application itself, thus reinforcing the need for the development of standard languages that can be used by these applications. There are still a few other approaches that use low-level concepts, but as such are not suitable to be used in a real-life context by nonexperts.

3. SIMULATION FRAMEWORK

As mentioned before, the specification of these languages stems from the development of a simulation framework for multivehicle mission simulation. Figure 1 represents the global architecture of this framework.

The central module of the system is the visual simulation platform. Given the added difficulty of simulating motion through a fluid environment such as the atmosphere, the chosen simulator was an aeronautical simulator, in this case Microsoft's FSX (Gimenes et al., 2008). The simulation can be performed by one or more instances of the simulator when the simulation load is too high for just one simulator or when the vehicles are spread over a large geographical area (Rodrigues et al., 2015).

The Control Panel has a central role in the system because it is responsible for environment and disturbance configuration, team and mission definition and loading, and providing system-wide status monitoring during mission execution. Environment and disturbance configurations are sent to the simulation platform, so that they can be correctly simulated and sensed by the several agents. It is the Control Panel that provides an interface between the framework and the user, providing a user-friendly manner for the user to specify all details (this tool is shown in more detail in Section 6). The Control Panel is responsible for creating a number of air traffic control (ATC) Agents (in according to their specification, using SDL, see Section 4.3), and Vehicle Agents (also according to their specification, using both SDL and TDL, see Sections 4.4 and 5.1).

The ATC Agent is responsible for ground, air, or sea traffic operations in and around the base of operations to which it is linked. Within the context of an airport, this agent represents the typical air traffic controller, being responsible for a central control of all ground traffic in the airport, as well as air traffic around the airport, routing all ground traffic from or to the

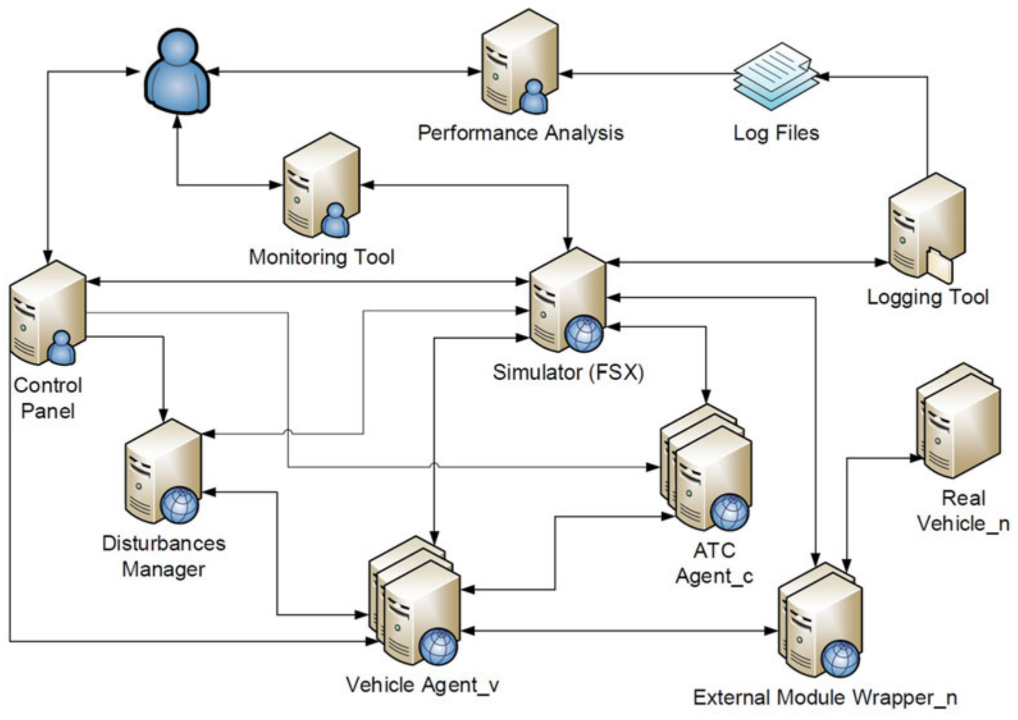


Fig. 1. Global system architecture.

defined landing or departure runway, and avoiding air traffic conflicts (Sousa et al., 2010). This agent can be configured for a more passive role, thus allowing for a decentralized traffic control approach, facilitating studies on the comparison between centralized and decentralized approaches to ATC (Camara et al., 2014).

The modules identified as *Vehicle Agent 1* through *Vehicle Agent v* represent the several (simulated) vehicles that are to exist within the system. Each one of these agents represents a vehicle and is responsible for handling actions such as navigation control, collision avoidance, and others. In addition, they are collectively responsible for mission planning and execution. Given that the application is intended to be used with both simulated and real vehicles, there is the possibility to use external modules, which communicate with the robotic agents represented by the virtual vehicles. These modules act as wrappers between application actions or commands and specific vehicle functionalities. They also allow the collection of real-world vehicle data that will both replace the simulated data, if discrepancies are detected, and serve as input to a calibration process that improves simulation realism.

The Monitoring Tool is responsible for providing both a real-time visualization of the status of the simulation and the agents, and providing updated values for several simulation and agent-related variables. The Logging Tool, together with vehicle and ATC Agents, is responsible for creating permanent log files for each simulation session, including general simulation parameters, the initial simulation status, communications among the several agents, and a detailed status description for each vehicle. These log files can then be used by the Performance Analysis Tool to provide the user with aggregated information regarding the simulation, and some analysis on the performance of a team in completing a given mission.

In a metaphorical comparison, the Control Panel can be seen as an airline operations center, the Vehicle Agents as representing aircraft pilots, the ATC Agent as the air traffic controller, the Logging Tool as the aircraft's flight data recorder (more commonly known as the black box), and the Monitoring Tool as a real-time flight tracker.

This framework allows for the execution of diverse cooperative multivehicle missions, including surveillance (forest surveillance that provides an early fire detection system; coastal and border patrol, in order to detect and track illegal activities, such as smuggling; urban observation that would detect dense traffic patterns, preventing larger traffic jams; and many other applications), reconnaissance and target tracking (especially useful in military operations and law enforcement activities, to provide real-time valuable information about enemy movements, or to follow a fugitive until apprehended by competent entities), aiding in search and rescue operations, and many other applications.

4. SDL

This section provides a full overview of the SDL, and each part of the scenario definition is analyzed in more detail. As previously stated, this dialect was developed in order to fully describe an operating scenario for a team (or several teams) of mobile robotic vehicles. The scenario contains elements describing the unchangeable part of the environment. It is defined as a tuple comprising set of bases of operations $B = \{b_1, b_2, \ldots, b_{n_B}\}$, which may contain an airport, port, and/or a ground base that can be used by one or more of the teams; a set of no-fly areas $N = \{n_1, n_2, \ldots, n_{n_N}\}$, which define the areas that no team can navigate through; a set of controllers $C = \{c_1, c_2, \ldots, c_{n_C}\}$, which control traffic on a well-defined area of space; and a set of agent types $T = \{t_1, t_2, \ldots, t_{n_T}\}$, which define the type of vehicles available. Equation (1) formalizes the representation of a scenario as a tuple constituted by the four sets: bases of operations, no-fly areas, controllers, and agent types.

$$\begin{aligned} \text{Scenario} &= \langle B, N, C, T \rangle, \\ B &= \{\text{Base OfOperations}\}, \\ N &= \{\text{Area}\}, \\ C &= \{\text{Controller}\}, \\ T &= \{\text{AgentType}\}. \end{aligned} \tag{1}$$

4.1. Bases of operations

This section of the SDL file contains a list of available bases of operations. Each base of operations has a unique identifier, which will later be referenced by TDL (see Section 5). For each base of operations, a number of information details are provided:

- *name:* The name by which the base of operations is known. If the base contains only an airport (or only a port, or only a ground base), the name of the base is usually coincident with its name.
- *mobility:* This element includes four Boolean attributes (air, land, water, and underwater), which indicate the type of vehicles the base provides support for. In addition, these attributes define the existence of the optional elements airport, port, and groundBase (see below).
- *description:* A brief textual description of the base of operations, which may include a listing of available services and facilities.
- *history:* This element can contain a detailed description of the base of operations, as well as the modifications it went through over time.
- *contactPerson:* This element contains detailed information about the person to contact regarding the base of operations. It includes name and title of the person to contact; the institution he works for and his position within that institution; address for physical correspondence (address, zip code, city, state, and country) and other contacts (e-mail, telephone, cell phone, and fax); and the possibility to add any additional information items, such as preferred contact hours or alternative contacts.
- *location:* Provides information regarding the location of the base of operations. It comprises a physical address (which may coincide with the address of the person to contact or not) and the coordinates for the location of the base (usually either the coordinates for the center of the base or those of the office where the contact person can be reached).
- *availability:* This element describes the temporal availability of the base for operations (e.g., one base of operations b_1 may only be available during daytime, but not for nocturnal operations, while another base b_2 may only be available during weekdays, but not during the weekend). If the base is not always available for operations, at least one availability slot must be indicated; each availability slot contains the start and end date and time of the period during which the base is available; in addition, if the specified availability slot occurs periodically, the rate of recurrence can be specified, as well as the initial and final dates during which the recurrence is valid.
- *airport:* This optional element contains a detailed description of the airport within the base of operations, its structure, and the services it provides (it is described in more detail below). The presence of this item is determined by the value of the *air* attribute of the *mobility* element.
- *port:* This optional element contains a detailed description of the port within the base of operations, its structure, and the services it provides to boats and/or submarines. The presence of this item is determined by the value of the *water* and *underwater* attributes of the *mobility* element.
- *groundBase:* This optional element contains a detailed description of the ground base within the base of operations, its structure, and the services it provides. The presence of this item is determined by the value of the *land* attribute of the *mobility* element.

4.1.1. Airport

As previously mentioned, this paper focuses only on air traffic, and as such, only airport information is presented. Considering the different formats and information included in the analyzed simulators, the authors decided to include a stripped down version of the airport description found in FSX, focusing on the important aspects, such as positioning, dimensions, and intersections of possible paths, leaving out the information regarding visual details, such as lights or signs. Equation (2) shows the contents of the airport element, which are explained in more detail in Table 2.

$$\begin{aligned} \text{Airport} = \langle &\text{name, description, contactPerson, location,} \\ &\text{IATA, ICAO, magVar}, H, R, T, P, G, U \rangle, \\ H &= \{\text{Helipad}\}, \\ R &= \{\text{Runway}\}, \\ T &= \{\text{Taxiway}\}, \\ P &= \{\text{Parking}\}, \\ G &= \{\text{Hangar}\}, \\ U &= \{\text{Utility}\}. \end{aligned} \tag{2}$$

The information contained in these elements (viz., the information retrieved from the runways and taxiways elements)

Table 2. *Airport element definition*

Element	Description
Name	The name by which the airport is known
Description	Textual description of the airport and the services it provides
ContactPerson	Information regarding the person of contact for the airport; its definition is the same as for the base of operations.
Location	Information regarding the location of the airport; its definition is the same as presented above for the base of operations.
IATA	The International Air Transport Association code of the airport; this code comprises three letters and is widely used for major airports, namely, in baggage tags.
ICAO	The International Civil Aviation Organization code of the airport; this code comprises four alphanumeric characters and provides a unique code for each airport worldwide.
MagVar	The magnetic variation (difference, given in degrees, between true North and magnetic North) at the airport location
Helipad	Helipads are relatively small, round or square regions of the airport used by helicopters for vertical takeoff and landing Designation: the name by which the helipad is known Coordinates: coordinates for the center of the helipad Radius: radius of the helipad Surface: material the surface of the helipad is made of
Runway	Runways are the straight, flat, and long strips of terrain used by aircraft for takeoff and landing Coordinates: coordinates for the center of the runway Length: length of the runway Width: width of the runway Surface: material the surface of the runway is made of BaseEnd: Contains information regarding one of the two orientations of the runway; it includes the designation of the runway (a number, from 01 to 36, corresponding to one-tenth of the magnetic heading of the runway), coordinates for the start and end points of the runway and its orientation (heading) ReciprocalEnd: Contains information regarding the other orientation of the runway; the designation should differ from the designation of the baseEnd by 18 and the orientation by 180°; the start and end points may match the end and start points of the baseEnd, respectively, but in some cases that may not be true
Taxiway	Taxiways are used for ground operations (either by aircraft or by other vehicles operating in the airport), connecting runways with other areas of the airport, such as parking spaces, fuel facilities, hangars, or helipads Designation: the name by which the taxiway is known Surface: material the surface of the taxiway is made of Width: width of the taxiway Path: Contains information about the shape of the taxiway; it includes both initial and final points of the taxiway, as well as a variable number of middle points where the taxiway changes direction or where it intersects with another taxiway or runway; all points contain the coordinates of their location; in case of interception, the taxiway(s) and/or runway(s) it intercepts with is(are) also specified.
Parking	Parking spaces are specific locations within the airport, used to park aircraft, usually for a relatively short period of time. Designation: the designation of the parking space Description: description of the parking space: purposes (used mainly by commercial aircraft, cargo companies, privately owned jets, or others) and other information Coordinates: coordinates of the parking space Radius: radius of the parking space Airlines: lists which airlines have priority of use over the specific parking space Connection: specifies the coordinates where the parking space connects to the taxiway network
Hangar	Hangars are closed structures, usually used for long-termed housing of aircraft, maintenance, or repair operations. Designation: the designation of the hangar Description: brief description of the hangar, especially in terms of its purposes and possible owner Shape: specifies the shape of the hangar (expressed as a polygon), its height, area and the location and size of the doors
Utility	There are four types of utilities (tower, fuel facility, battery facility and water facility) with three common elements Designation: the designation of the utility Coordinates: coordinates for the center of the utility Radius: radius of the utility In addition to these, the tower also has a height, and fuel and water facilities have the available quantity of fuel or water

is structured in a manner that facilitates its use to construct a graph containing the possible paths to be used by airplanes while on the ground (e.g., taxiways includes for each point in its path, corresponding to a node of the graph, information regarding the taxiways/runways it connects to, thus allowing for the graph to be constructed; Sousa, 2010). This information is of major importance so that the agents can plan their paths with maximum efficiency.

4.2. No-fly areas

This section of the scenario description file has a straightforward purpose: clearly defining all areas that cannot be navigated through, at a global level, by any vehicle of any team. Even though the name suggests that the areas defined in this section only apply to airspace and air traffic, the areas can be defined for air, land, water, and/or underwater vehicles.

Every area element has a unique identifier, and is defined by a name; the medium it applies to (this element has the same definition as the one presented for the base of operations, with four Boolean attributes: land, air, water, and underwater), which determines the vehicles that cannot navigate through the area; the temporal availability of the area (e.g., aircrafts may not fly over a given area n_1 at night, but they may do so during daytime; the definition of availability is also the same as the one shown above for the base of operations); and the shape of the area, which can be expressed as either a polygon (composed of three or more vertexes, each identified by latitude and longitude) or a circle (composed of the center latitude and longitude and a radius), extruded vertically from minimum to maximum altitudes. Equation (3) shows the definition of the area element.

$$\begin{aligned} \text{Area}_{\text{ID}} &= \langle \text{Name, Medium, Availability, Polygon} \vee \text{Circle} \rangle, \\ \text{Polygon} &= \langle \text{minAlt, maxAlt, } \{\text{vertex}\}^{3+} \rangle, \\ \text{Circle} &= \langle \text{minAlt, maxAlt, center, radius} \rangle. \end{aligned} \tag{3}$$

4.3. Controllers

This section of the scenario description file contains a list of traffic controllers (for air, water, or land traffic) and their respective details: a unique identifier of the controller; the base of operations $b \in B$ it is associated with; the level of autonomy a vehicle has when moving through the controller area, indicated by the *requiredAction* attribute; an indication of the type (role) of controller it represents; the area it has jurisdiction over (area definition is the same as shown above); and the contact frequencies (e.g., approach and departure control can be handled in a different frequency than ground control). Figure 2 shows the information contained within this element in a graphical notation. The figure shows the selected *controller* XML element in blue on the left, with the associated attributes inside the attributes box on the top right of the controller element; other XML elements within the controller element are presented as a sequence (indicated by the three-dotted symbol right of the controller element) of one *area* element and one *frequencies* element (which has a sequence of one or more *frequency* elements). The constraints are related to the correct and valid reference to an identifier of a base of operations.

This information is used by ATC Agents as well as by the vehicle agents to determine their behavior when navigating within the areas defined in these elements. For instance, one controller c_1 may have full control over the area surrounding the airport (as air traffic controllers for major airports usually do), and all agents representing vehicles moving through that area would have to clear every decision with the controller; another controller c_2 may have more of a passive behavior (as do some controllers of smaller airfields), simply being informed of the vehicle decisions, but having no direct control over them. There are three levels of control defined in the *requiredAction* attribute: in the lowest level, the controller is only informed of the vehicle decisions and has no control over them; in the intermediary level, the vehicle has the autonomy to make decisions, but not to implement them without the consent of the controller; in the highest level, vehicles have no autonomy regarding motion and have to obey to all decisions made by the controller.

4.4. Agent types

This section of the scenario description file contains a list of available vehicle types. Each vehicle type has a unique identifier, which will later be referenced by the TDL file (see Section 5.1) to indicate the type of each vehicle that composes the teams. The information regarding the vehicle type is divided into five categories:

- *simulated agent:* This category contains only one element: title. The title is a string that uniquely identifies a vehicle type within the simulation platform, and several details regarding the vehicle type can be retrieved from the simulator, given this string (even though this option may seem to be closely related to the simulator being used, it is usually enough to identify an object type in most simulators; however, if this is not the case, the language can easily be extended to include the necessary information).
- *real agent:* Information describing the real vehicle includes the category (aircraft, car, boat, or submarine), type, manufacturer, model, and variation of the vehicle, as well as any additional information one might want to include.
- *physical:* This element includes characteristics regarding aircraft dimensions: length, height (measured at the highest point, usually the tail), and wingspan (width of the aircraft measured from one wingtip to the other); wing area; empty weight; maximum payload (cargo) and fuel it can carry; and engine information (number and type of engines).
- *performance:* This category includes maximum allowed weight at takeoff; the minimum required runway length for takeoff and landing; reference speeds: cruise, maximum (the maximum speed at which the aircraft can fly without the risk of structural damages), and stall (the minimum speed at which the aircraft is capable of generating lift; with a lower speed the aircraft starts falling); rate of climb (the maximum vertical velocity of the aircraft); range (the distance the aircraft can travel without refueling) and service ceiling (the maximum altitude at which the aircraft can operate); fuel flow (the average

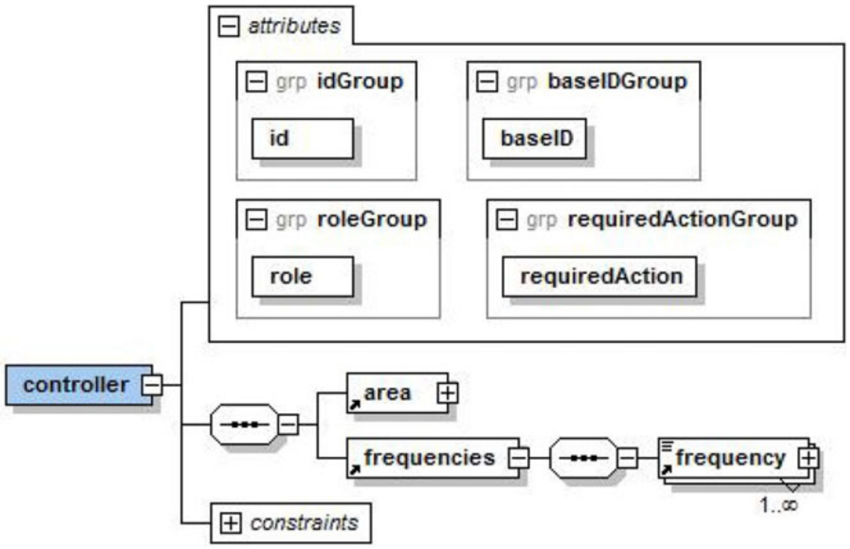


Fig. 2. Controller element definition.

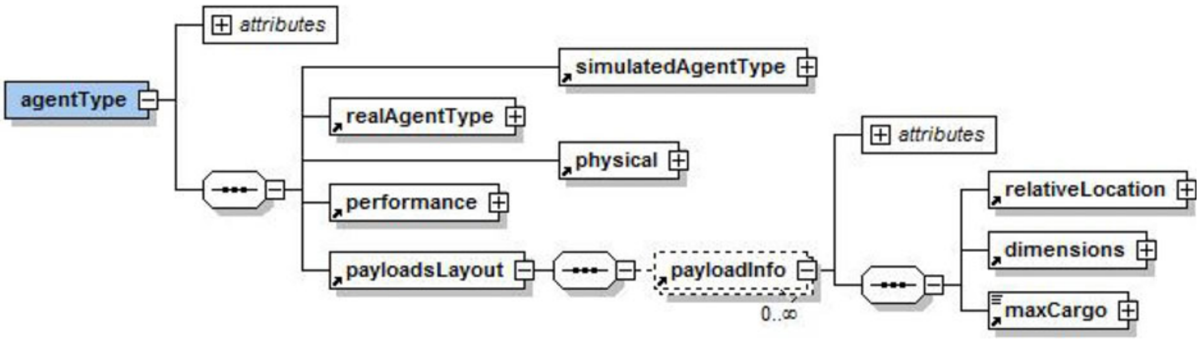


Fig. 3. Agent type element definition.

fuel consumption at cruise speed, considering a straight level flight); and drag coefficient.

- *payload layout:* Payload layout includes information regarding each payload station, namely, its location (in relation to the aircraft's center), dimensions, and maximum cargo it can transport. This information will then be used by TDL as restrictions to the cargo or sensors each payload may contain (see Section 5.1.1).

Information regarding vehicles other than aircraft differs from the information shown above in the physical and performance categories, while the remaining three categories have identical structure for all vehicle types (aircraft-specific characteristics are replaced by characteristics specific to other vehicle types). Figure 3 shows the definition of the *agentType* element in a graphical notation.

5. TDL

This section provides an overview of TDL. The TDL file allows for the description of any number of teams. Each team has a unique identifier and contains several elements, such as the bases of operations $U = \{u_1, u_2, \ldots, u_{n_U}\} \subseteq B$ it can use, additional no-fly areas $A = \{a_1, a_2, \ldots, a_{n_A}\}$, and the description of all agents $V = \{v_1, v_2, \ldots, v_{n_V}\}$ composing the team [see Eq. (4)].

$$\begin{aligned}\text{Team}_{\text{ID}} = \langle &\text{Name, Description, History, Purposes,}\\ &\text{Mobility, ContactPerson}, A, U, V\rangle,\\ A &= \{\text{Area}\},\\ U &= \{\text{UsableBaseOfOperations}\},\\ V &= \{\text{Agent}\}.\end{aligned} \tag{4}$$

It is important to note that TDL is intended to solely specify team composition and operational constraints; all other information, namely, related to team organization, hierarchy, behavior, and others, are specified at the mission level, using MDL. Table 3 describes the contents of the team element in more detail.

Table 3. *Team element definition*

Element	Description
Name	The name of the team
Description	Textual description of the team and its capabilities
History	Description of the team's history; it may include information regarding past missions, team composition through time, or any other information.
Purposes	Describes the purposes of the team, such as the type of missions it was designed to, or is capable of performing
Mobility	Indicates the aggregated mobility of the team (the definition of this element is the same as presented above)
ContactPerson	Information regarding the person of contact for the team; the definition of this element is the same as for the base of operations.
AdditionalNoFlyAreas	This element contains a list of area elements, with the same definition as shown above (see Section 4.2), that indicates additional areas the specified team cannot navigate through (even though other teams might be able to). As such, the specific team cannot navigate through both the areas specified in SDL and in TDL, $N \cup A$
UsableBaseOfOperations	This element contains a list of references to the identifier of the base of operations, as defined in the SDL file (see Section 4.1), and indicates which bases can be used by the team.
Agent	Contains a description of each vehicle that composes the team; this element is detailed below.

5.1. Agents

This section of the team description file contains the key element of the team: the list of vehicles that compose the team, each containing the following information, in addition to a unique identifier:

- *agentTypeID:* This attribute $agentTypeID \in T$ references the vehicle type identifier, as described in Section 4.4.
- *name:* The name by which the vehicle is known within the team.
- *description:* A textual description of the vehicle, and its foremost capabilities.
- *initialLocation:* The initial location of an aircraft is given by the identifier $b \in U$ of a base of operations, the identifier of a parking space, hangar, helipad, or utility location within the airport of the base of operations b, and the direction the aircraft is facing (for other vehicle types, the location may refer to the port or ground base instead of airport).
- *state:* Vehicle status includes information regarding the amount of fuel it has; the position of the landing gear and several control surfaces (flaps, aileron, rudder, elevator); an indication of which lights (if any) are turned on; and an indication as to whether or not the doors are open.
- *realAgent:* Real vehicle information includes registration code (also known as tail number, the aircraft's equivalent to the license plate); name of the aircraft; and communication frequencies (when an actual vehicle is being used, these values are also used to configure the external module wrapper).
- *simulatedAgent:* Simulated vehicle information includes the vehicle's tail number (necessary to instantiate the vehicle in the simulation platform), which should be

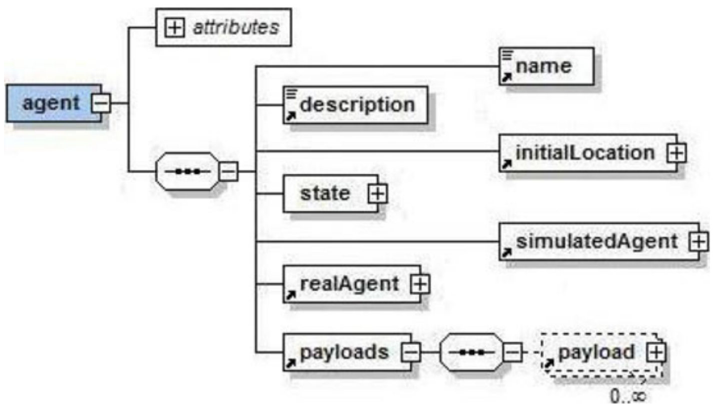


Fig. 4. Agent element definition.

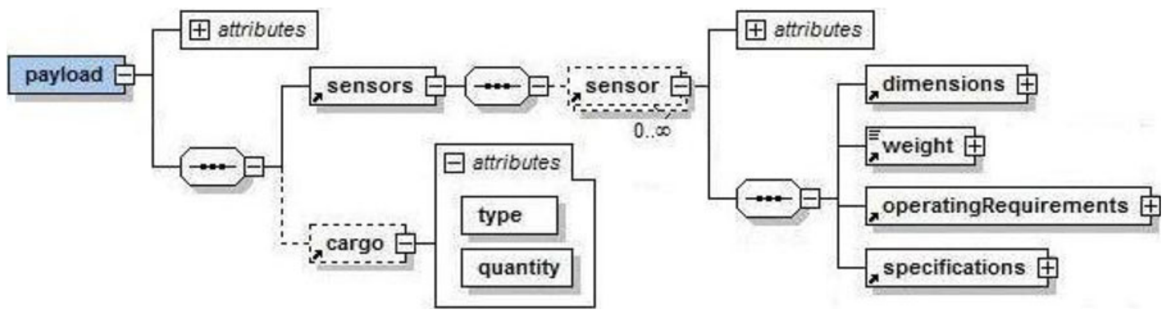


Fig. 5. Payload element definition.

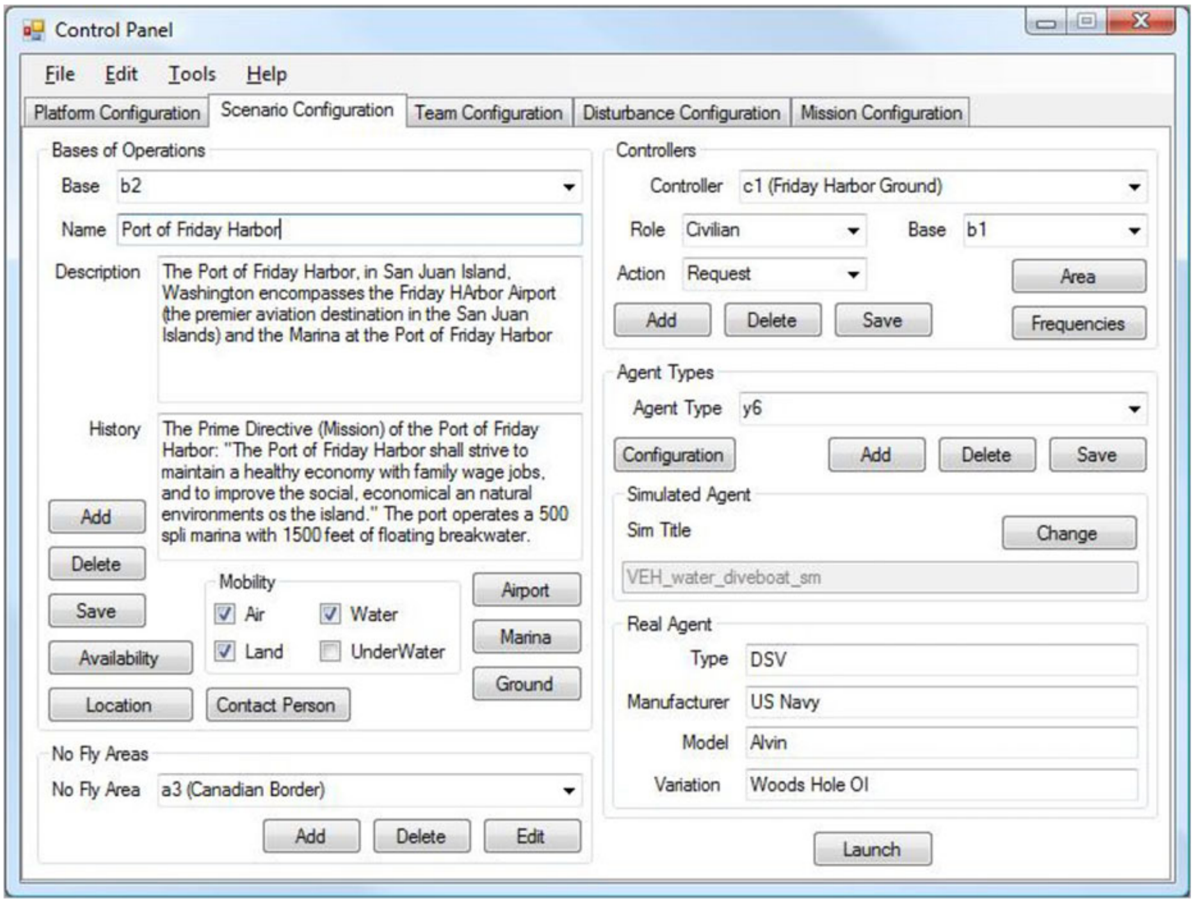
the same as the one specified for the real vehicle, and may not be repeated.

- *payload:* Specifies the contents of each payload station (detailed below).

Figure 4 shows the agent element in a graphical notation.

5.1.1. *Payloads*

For each payload station identified in the vehicle type *agentTypeID* ∈ *T* definition (see Section 4.4), the details about the sensors it carries and/or cargo it transports are provided. Sensor definition includes type (temperature, CO_2 detector, infrared, camera, and so on), dimensions, weight, operational requirements (such as temperature or humidity ranges, voltage and current specifications, or power consumption), and other specifications (such as accuracy, response time, range of values detected, output definition, and any other specifications). Alternatively, the payload station can contain cargo, in which case the type and quantity are indicated (e.g., 350 L of fire retardant). Figure 5 shows the definition of a payload.



Bases of Operations
Base: b2
Name: Port of Friday Harbor
Description: The Port of Friday Harbor, in San Juan Island, Washington encompasses the Friday HArbor Airport (the premier aviation destination in the San Juan Islands) and the Marina at the Port of Friday Harbor
History: The Prime Directive (Mission) of the Port of Friday Harbor: "The Port of Friday Harbor shall strive to maintain a healthy economy with family wage jobs, and to improve the social, economical an natural environments os the island." The port operates a 500 spli marina with 1500 feet of floating breakwater.
Mobility: [✓ Air] [✓ Water] [✓ Land] UnderWater

No Fly Areas
No Fly Area: a3 (Canadian Border)

Controllers
Controller: c1 (Friday Harbor Ground)
Role: Civilian
Base: b1
Action: Request

Agent Types
Agent Type: y6
Simulated Agent
Sim Title: VEH_water_diveboat_sm
Real Agent
Type: DSV
Manufacturer: US Navy
Model: Alvin
Variation: Woods Hole OI

Fig. 6. Scenario configuration tool.

6. IMPLEMENTATION

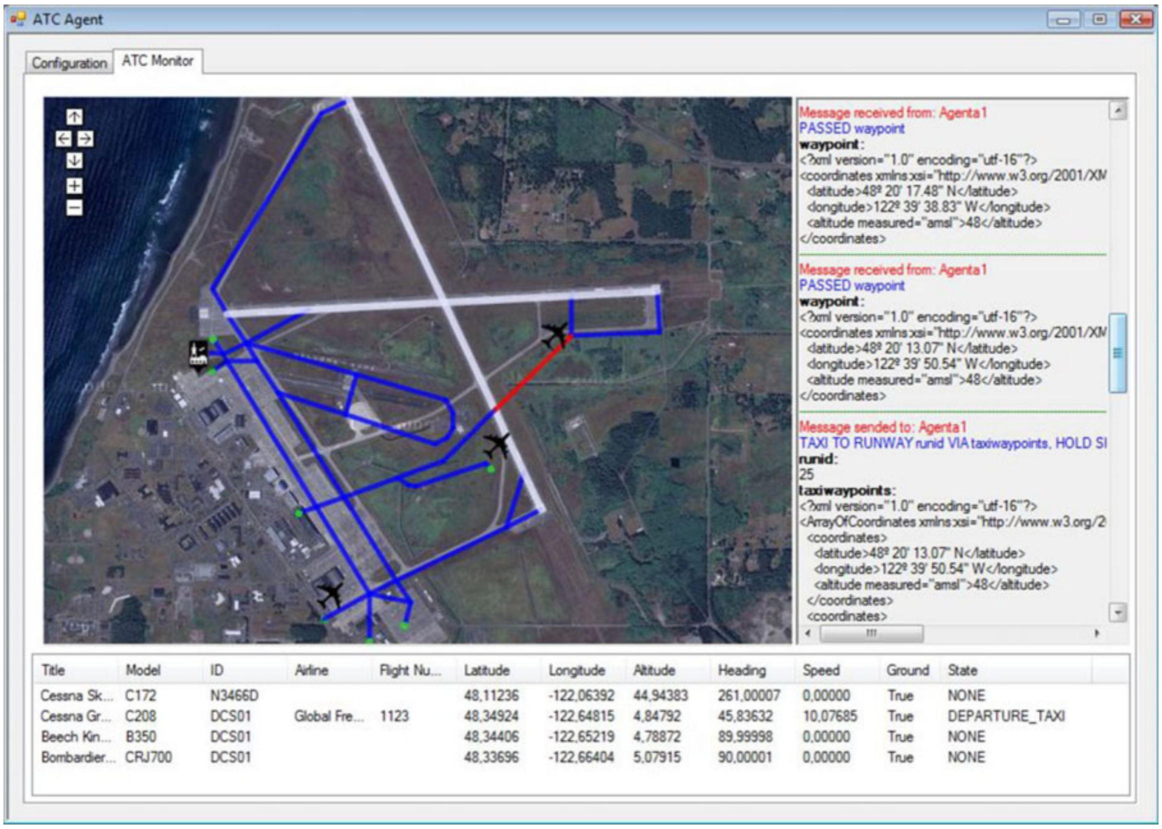
The two dialects have been specified using XML Schema (thus ensuring extensibility and readability), and C# classes were created to reflect the contents of said schemas (more specifically, the XML Schema Definition Tool was used, which can generate a set of classes, based on a given XML Schema (Microsoft Corporation, 2010)). The dialects were then integrated into the framework under development, by including in the Control Panel (as described briefly in Section 1) the ability to create and edit both SDL and TDL files. Identifiers are generated automatically by the application, to abstract the user from low-level implementation details, and some graphical aids are provided to facilitate the task of specifying all elements; for instance, areas (both no-fly areas and the areas controllers have jurisdiction over) are defined using an interactive Google Maps plugin, with the defined area overlaying satellite imagery of the location; the several points that define a polygonal area, for instance, can be interactively added and moved in the map simply by clicking and dragging.

Figure 6 shows the main screen for the scenario editing tool. The file containing the scenario description can be easily loaded into a scenario object, which is then used by the interface to manipulate the data it contains. The Launch button, on the bottom right, deploys the ATC Agents, using the configurations provided in the controllers section and providing them with the information regarding the base of operations they will have jurisdiction over.

This implementation allows ATC Agents to be aware of their own operating configuration and of airport configuration. With this information, the ATC Agent is capable of performing typical traffic control operations. Figure 7 shows the interface of an ATC Agent; the airport configuration of the SDL file (partly shown in Fig. 8) was used to construct the airport runway and taxiway network (runways are shown in light gray, while taxiways are shown in either blue, when not in use, or red, when being used by an aircraft), which is used for ground traffic management (Sousa, 2010).

Figure 9 contains an example of a vehicle type (an aircraft) specification using SDL (only one payload station is shown for simplicity).

Figure 10 shows the main screen for the teams editing tool. By using the same process, this interface allows for the manipulation of TDL files and their contents. The Launch button deploys the vehicle agents, using the configurations from both this panel and the scenario editing panel (viz., vehicle type definition), as well as information regarding operational constraints, namely, the bases of operations that can be used, the areas that cannot be flown through, and the areas that require some sort of communication with the respective controller. The information regarding the bases of operations that can be used is important to determine where the plane can



```
Message received from: Agenta1
PASSED waypoint
waypoint:
<?xml version="1.0" encoding="utf-16"?>
<coordinates xmlns:xsi="http://www.w3.org/2001/XM
  <latitude>48º 20' 17.48" N</latitude>
  <longitude>122º 39' 38.83" W</longitude>
  <altitude measured="amsl">48</altitude>
</coordinates>
Message received from: Agenta1
PASSED waypoint
waypoint:
<?xml version="1.0" encoding="utf-16"?>
<coordinates xmlns:xsi="http://www.w3.org/2001/XM
  <latitude>48º 20' 13.07" N</latitude>
  <longitude>122º 39' 50.54" W</longitude>
  <altitude measured="amsl">48</altitude>
</coordinates>
Message sended to: Agenta1
TAXI TO RUNWAY runid VIA taxiwaypoints, HOLD SI
runid:
25
taxiwaypoints:
<?xml version="1.0" encoding="utf-16"?>
<ArrayOfCoordinates xmlns:xsi="http://www.w3.org/2
  <coordinates>
    <latitude>48º 20' 13.07" N</latitude>
    <longitude>122º 39' 50.54" W</longitude>
    <altitude measured="amsl">48</altitude>
  </coordinates>
  <coordinates>
```

Title	Model	ID	Airline	Flight Nu...	Latitude	Longitude	Altitude	Heading	Speed	Ground	State
Cessna Sk...	C172	N3466D			48,11236	-122,06392	44,94383	261,00007	0,00000	True	NONE
Cessna Gr...	C208	DCS01	Global Fre...	1123	48,34924	-122,64815	4,84792	45,83632	10,07685	True	DEPARTURE_TAXI
Beech Kin...	B350	DCS01			48,34406	-122,65219	4,78872	89,99998	0,00000	True	NONE
Bombardier...	CRJ700	DCS01			48,33696	-122,66404	5,07915	90,00001	0,00000	True	NONE

Fig. 7. ATC Agent Monitoring tab.

```
<airport>
      <name>Whidbey Island Naval Air Station</name>
      <description>...</description>
      <contactPerson>...</contactPerson>
      <location>...</location>
      <ICAO>KNUW</ICAO>
      <IATA>NUW</IATA>
      <magVar>19</magVar>
      <runways>
            <runway id="r07-25">
                  <coordinates>...</coordinates>
                  <length lengthUnit="Meter">2434</length>
                  <width lengthUnit="Meter">60.96</width>
                  <surface>Concrete</surface>
                  <baseEnd>
                        <designation>07</designation>
                        <startpoint>...</startpoint>
                        <heading headingType="True">86.97</heading>
                        <endpoint>...</endpoint>
                  </baseEnd>
                  <reciprocalEnd>...</reciprocalEnd>
            </runway>
            <runway id="r14-32">...</runway>
      </runways>
      <helipads>  ...  </helipads>
      <taxiways>   ...
            <taxiway id="x7">
                  <designation>Taxiway G</designation>
                  <surface>Concrete</surface>
                  <width lengthUnit="Meter">30.48</width>
                  <path>...</path>
            </taxiway>    ...
      </taxiways>
      <parkingSpaces>  ...  </parkingSpaces>
      <hangars>  ...  </hangars>
      <utilities>  ...  </utilities>
</airport>
```

Fig. 8. Airport example (SDL).

land, in case it needs to refuel, or when the mission is over. The no-fly areas have a direct impact in route planning, because they must be avoided; the information regarding controllers also has a direct impact in the behavior of the aircraft when inside the areas controlled by them. In addition, agents representing robotic vehicles now have self-awareness as to what sensors or cargo they carry (in a structured manner).

Figure 11 contains an example of a vehicle specification using TDL. Notice that the *agentTypeID* attribute corresponds to the vehicle type shown above, and as such, the payload stations also correspond to the ones shown above (and the respective dimensions and weight restrictions must be respected).

SDL and TDL combine the advantages of XML with the ability to describe a set of entities used to depict a scenario and a team, respectively. Unlike the languages referenced in Section 2, which are targeted at specific needs (geographical features, aeronautical information, or others), SDL and TDL allow for the description of distinctive entities, of not only geographical nature (such as bases of operations and no-fly areas), but also controllers and vehicles, and their distinctive and idiosyncratic features. This diversity of represented entities allows for the definition of all the necessary elements to describe the static components of the envisioned system.

7. VALIDATION AND RESULTS

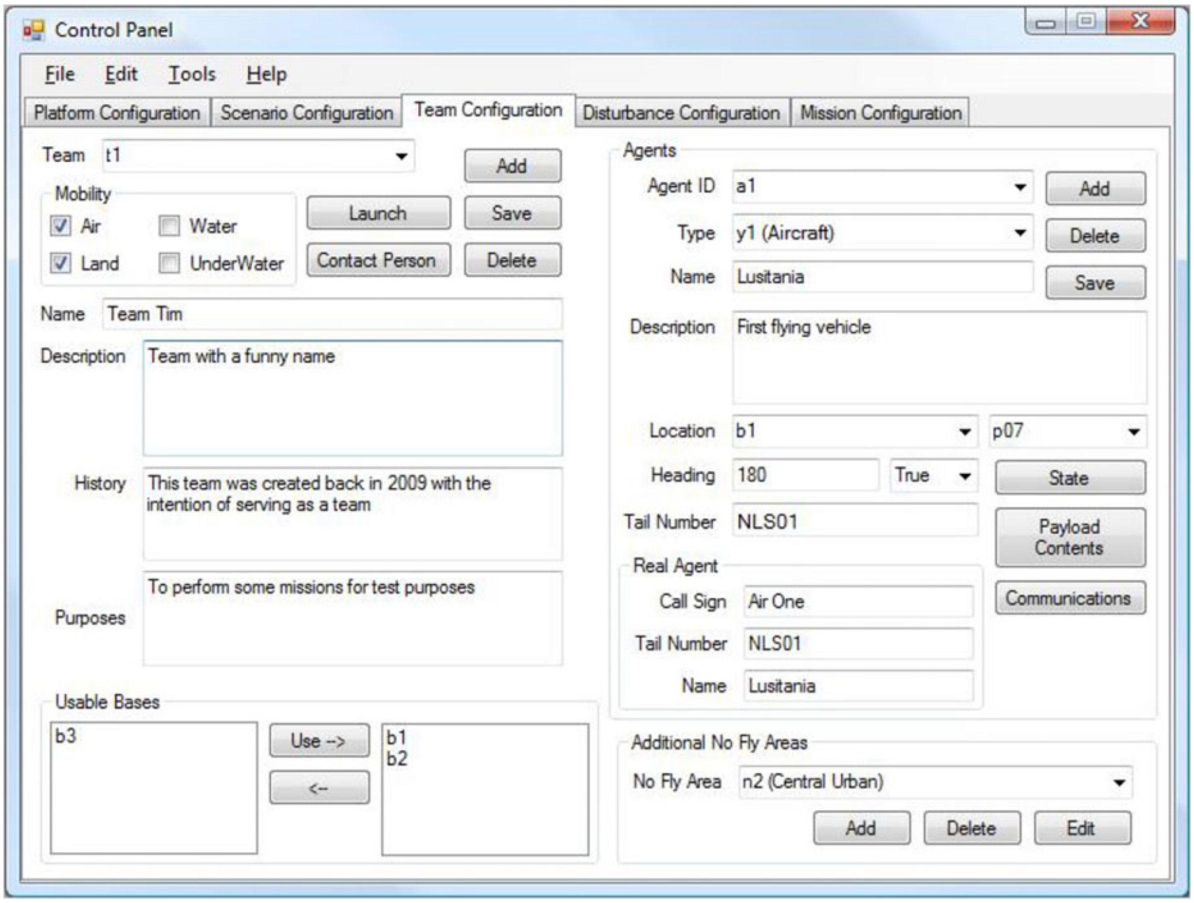
To validate the developed dialects, a group of experts and practitioners was formed. Most members of this group were informally consulted during the development stage of the dialects, using an iterative process by which each of them was able to both review the current state of the dialects and provide additional input for their development, in a Delphi-like process, used in many works (Abreu et al., 2012;

```
<agentType id="y2">
      <simulatedAgentType>
            <simTitle>Cessna Skyhawk 172SP Paint1</simTitle>
      </simulatedAgentType>
      <realAgentType>
            <category>Aircraft</category>
            <vehicleType>Light Passenger Aircraft</vehicleType>
            <manufacturer>Cessna</manufacturer>
            <model>172SP</model>
            <variation>Blue and Gold</variation>
      </realAgentType>
      <physical>
            <length lengthUnit="Meter">8.28</length>
            <height lengthUnit="Meter">2.72</height>
            <emptyWeight massUnit="Kilogram">736</emptyWeight>
            <maxPayload massUnit="Kilogram">326</maxPayload>
            <nEngines engineType="TurboProp">1</nEngines>
            <maxFuel volumeUnit="Gallon" fuelType="Avgas">135</maxFuel>
            <wingspan lengthUnit="Meter">11</wingspan>
            <wingArea areaUnit="Square Meter">16.2</wingArea>
      </physical>
      <performance>
            <cruiseSpeed velocityUnit="Nautical Mile per Hour (Knot)">126
</cruiseSpeed>
            <maxSpeed velocityUnit="Nautical Mile per Hour (Knot)">163
</maxSpeed>
            <range lengthUnit="Nautical Mile">610</range>
            <fuelFlow volumeUnit="Gallon" timeUnit="Minute">3.54</fuelFlow>
            <dragCoefficient>0.571243</dragCoefficient>
            <stallSpeed velocityUnit="Nautical Mile per Hour (Knot)">58
</stallSpeed>
            <climbRate velocityUnit="Meter per Second">3.7</climbRate>
            <ceiling lengthUnit="Foot">14000</ceiling>
            <requiredRunwayLength lengthUnit="Foot">1633
</requiredRunwayLength>
            <maxTakeoffWeight massUnit="Kilogram">1157</maxTakeoffWeight>
      </performance>
      <payloadsLayout>
            <payloadInfo name="PayloadCenter">
                  <relativeLocation>      ...       </relativeLocation>
                  <dimensions>            ...       </dimensions>
                  <maxCargo massUnit="Kilogram">57</maxCargo>
            </payloadInfo>
      </payloadsLayout>
</agentType
```

Fig. 9. Vehicle type example (SDL).

Almeida et al., 2013). In addition, in a second stage, after the dialects were defined, a survey was conducted among the members, in order to assess their satisfaction with the quality of the dialects, based on the concepts and concerns each one brought forward.

The survey was divided into three parts: the first part was divided into two sections, each one related to each of the dialects as a whole, and each one composed of nine questions that evaluate the overall satisfaction with the specific dialect, as well as aspects that could be improved. The second part is divided into five smaller sections focusing on five portions of the dialects (bases of operations, controllers, no-fly areas, and agent types from SDL, as well as individual agents from TDL), each of which includes three questions. The third part, which includes one question and a practical exercise, is focused on the Control Panel, and relates to the usability of this tool as a means of creating and editing a simple scenario and a team using SDL and TDL. In the practical exercise, the participants were asked to use the Control Panel to define a simple airport (with a single runway, two taxiways,



Team: t1

Mobility: [✓ Air] [✓ Land] Water UnderWater

Name: Team Tim

Description: Team with a funny name

History: This team was created back in 2009 with the intention of serving as a team

Purposes: To perform some missions for test purposes

Usable Bases: b3 | b1 b2

Agents

Agent ID: a1

Type: y1 (Aircraft)

Name: Lusitania

Description: First flying vehicle

Location: b1 p07

Heading: 180 True

Tail Number: NLS01

Real Agent

Call Sign: Air One

Tail Number: NLS01

Name: Lusitania

Additional No Fly Areas

No Fly Area: n2 (Central Urban)

Fig. 10. Teams configuration tool.

and four parking spaces); a controller that operates within the vicinities of that airport; two vehicle types; and a team of four vehicles (two of each kind) to operate from the previously defined base of operations. After that, they were asked to fill out a System Usability Scale (SUS; Brooke, 1996) survey to measure its usability.

A total of 42 responses were collected from the group (leaving out 4 members who were unavailable to participate). Out of these 42 participants, 14 are practitioners (most of which have some sort of connection to protective and armed forces) and 28 are researchers in the area and adjacent areas.

Because of the difference in background of the two identified groups, the authors felt pertinent to assess as to whether or not different results were obtained from these two groups. For that, and in each question, the Shapiro–Wilk test was used to test the normality of the data (using S as the value of the test and P as the significance for the test, considering an α level of 0.05). The results obtained for each question varied from $S = 0.7896$ for $P = 0.06647$ to $S = 0.9099$ for $P = 0.4671$. After that, the two groups were tested using a Mann–Whitney–Wilcox test (using W as the value of the test), with an α level set at 0.05. These tests produced results that show that the two groups are not distinct; the p values for the several questions ranged from 0.1719 to 0.7449. Results range from $W = 5.5$ for $P = 0.1719$ to $W = 10.5$ for $P = 0.7449$. Because the two groups cannot be considered distinct in terms of results, these were analyzed together.

7.1. Part I: Dialect validation

This part of the survey was composed of five questions using a 5-point Likert scale, three questions that called for written comments and one question to assess knowledge of languages or syntaxes with similar goals (followed by a comparison with SDL or TDL, if the answer is positive). For the first question ("Are you satisfied with the representation of your concerns in SDL/TDL?"), 81% of the respondents answered *agree* or *strongly agree* regarding SDL, obtaining an average classification of 4.07 out of 5, while 78.6% answered *agree* or *strongly agree* regarding TDL, obtaining an average classification of 4.02 out of 5. For the second question, regarding the flexibility of SDL/TDL in the specification of a scenario and team, an average of 3.9 was achieved for SDL, with 69% of answers being rated as *agree* or *strongly agree*, and an average of 4.0 was achieved for TDL, with 73.8% of answers

```
<agent id="a2" agentTypeID="y2">
      <name>Skyhawk One</name>
      <description>1st auton. vehicle based on Cessna Skyhawk</description>
      <initialLocation baseID="b1" locationID="p2">
            <heading headingType="True">180</heading>
      </initialLocation>
      <state>
            <fuel volumeUnit="Gallon">134</fuel>
            <lights cabin="false" logo="false" wing="false"
recognition="false" panel="false" strobe="false" taxi="false"
landing="false" beacon="false" nav="false" />
            <doorsOpen>false</doorsOpen>
            <gearDown>true</gearDown>
            <flapsHandlePosition>3</flapsHandlePosition>
            <rudder>-0.01334</rudder>
            <aileron>-0.1213</aileron>
            <elevator>0.01223</elevator>
      </state>
      <simulatedAgent>
            <tailNumber>NC342</tailNumber>
      </simulatedAgent>
      <realAgent>
            <communication>  ...  </communication>
            <ATC>
                  <callSign>Hawk One</callSign>
                  <tailNumber>NC342</tailNumber>
                  <name>Skywawk One</name>
            </ATC>
      </realAgent>
      <payloads>
            <payload name="PayloadCenter">
                  <sensors>
                        <sensor sensorType="CO2">
                              <dimensions>      ...        </dimensions>
                              <weight massUnit="Kilogram">1.12</weight>
                              <operatingRequirements>
                                    <temperature>-40°C - 75°C</temperature>
                                    <humidity>0 - 99%</humidity>
                                    <current>10 mA</current>
                                    <powerConsumption><100mW
</powerConsumption>
                              </operatingRequirements>
                              <specifications>
                                    <specification name="Range">0-
2000ppm</specification>
                                    <specification name="Resolution">10ppm
</specification>
                                    <specification name="Output Range">0-22
mA</specification>
                                    <specification name="Response Time">5
ms</specification>
                              </specifications>
                        </sensor>
                  </sensors>
            </payload>
      </payloads>
</agent>
```

Fig. 11. Vehicle example (TDL).

being rated as *agree* or *strongly agree*. The following three questions are free-text questions, and ask the user to point the main concerns he had regarding SDL and TDL that were present in the dialects' specification, concerns that are not represented in SDL or TDL, and concerns he feels are represented in SDL or TDL that are not necessary or desired. These collected answers were used to plan future extensions of both SDL and TDL, as presented in Section 8. It should be noted that a wide range of answers was obtained, in most cases not presenting consensual results (e.g., one respondent may consider one concern to be superfluous while others consider it to be an asset). The users were then asked as to whether they were aware of other dialects or syntaxes with similar goals. Fifteen out of the 42 respondents answered *yes* regarding SDL, and 12 answered *yes* regarding TDL. For these individuals, two additional questions were considered, regarding the comparison of SDL and/or TDL to that/those other dialects. The first one, considering again a response in a 5-point Likert scale, obtained an average of 4.1 out of 5 for SDL, with 12 out of the 15 respondents considering SDL to be *better* or *far better* than the other one(s). Regarding TDL, an average of 3.9 was achieved, with 9 out of the 12 respondents considering TDL to be *better* or *far better* than the other one(s). The second question, in free-text, asked the user to list the main advantages/disadvantages of SDL/TDL (which were then added to other free-text responses to plan for future extensions of the dialects). The final three questions asked the user if he feels SDL and TDL helps in defining a scenario and a team (in contrast to not having such dialects); if he thinks SDL /TDL has applicability; and finally to make an overall evaluation of SDL/TDL. The average results for SDL are 3.88, 3.88, and 3.95 with 73.8%, 71.4%, and 81%, respectively, of answers being *agree* or *strongly agree* for each of the three questions. The average results for TDL are 3.93, 3.98, and 4.05 with 76.2%, 76.2%, and 78.6%, respectively, of answers being *agree* or *strongly agree* for each of the three questions.

7.2. Part II: Detailed dialect validation

This part of the survey was composed of five sections, each of which contained three questions. The first question, using a Likert scale, asks the user to indicate if he agrees that that part of SDL (or TDL, for the fifth section) represents his concerns. The remaining two questions are free-text questions and are meant to identify concepts and concerns either not represented in SDL/TDL or that the user thinks are superfluous or overdetailed. The results for the first question average 4.2 for the section regarding bases of operations, 4.3 for the no-fly areas section, 3.9 for the controller section, 3.8 for the agent type section, and 4.1 for the agent section, which results in a combined average of 4.06. Despite these results, few respondents with an answer distinct from *strongly agree* in Question 1 were able to identify and list any concerns under- or overrepresented in the dialects. All the answers to the last two questions were compiled for each section, and the suggestions are being taken into consideration (see Section 8).

7.3. Part III: Tool usability

This part of the survey was composed of one question and a practical exercise followed by a SUS questionnaire. The first question asks the user if he believes the Control Panel accurately allows for the specification of SDL and TDL files. The results have an average of 4.1 out of 5, with 85.8% of respondents answering *agree* or *strongly agree*.

The practical exercise consisted in asking the user to define a simple scenario and a team using the Control Panel; the characteristics of each of the components of the scenario and team were specified and given to the users. These exercises were timed, and the definitions were also validated according to the desired characteristics. On average, participants took 34:51 min to define the scenario and team, with the minimum time being 24:37 min and the maximum of 44:53 min. While these values may seem excessive for the definition of a simple scenario and team, one has to keep in mind that the definition of both scenario and team only needs to be performed once. The SUS survey produced an overall score of 86.7 (out of a maximum of 100), which can be considered a good score (Lewis & Sauro, 2009).

8. CONCLUSIONS AND FUTURE WORK

The main conclusions are drawn and the main future work areas are pointed out in this section. Regarding the intended characteristics of the languages, most of them were accomplished by using an XML dialect to implement the languages: readability, extensibility, system independence, and data validation (through the use of XML Schemas). Being XML-based dialects, both SDL and TDL can easily be extended, by adding or altering the structure or contents of any part of the dialects. Automated tools (the XML Schema Definition Tool) facilitate the process of adapting the developed application to rapidly meet the changes made to the dialects. Because XML is easily readable and editable by humans, both SDL and TDL files can be manually altered in any text editor.

All intended structures were represented in either SDL or TDL: operating scenario (bases of operations, including airport, port, and ground stations, and vehicle types), control structures (controllers) and constraints, both at a global (no-fly areas) and a team (additional no-fly areas and usable bases of operations) level, as well as team composition (vehicles). These dialects allow for a high-level definition of the operating scenario and team composition and capabilities, which allows any platform that is based on these dialects to be used by non-experts as well.

A survey conducted among 42 participants showed that SDL and TDL generally receive a positive evaluation, as well as the Control Panel sections that deal with SDL and TDL files. The overall evaluation of SDL achieved a score of 4 points (out of 5), with 81% of the answers at ≥ 4; the overall evaluation of TDL was a score of 4 (out of 5), with 78.6% of the answers at ≥ 4. Regarding the usability of the developed tool, an average score of 86.7 was achieved in the SUS test.

These results prove not only that a tool based on SDL and TDL can be easily used by nonexperts to setup an operating scenario and define a team of vehicles but also that SDL and TDL can be used to specify all aspects regarded as important for scenario and team definition.

Some extensions and part of existing dialects could have been used in specific portions of the described dialects:

- GML could have been used to describe areas (no-fly and controller areas) in more detail. Even though controller and no-fly areas are expressed as either a polygon or a circle, with lower and upper altitude limits (which correspond to how these areas are usually defined in the real world; Federal Aviation Administration, 2010), a generalization could be made, as to express other geometrical forms, or even area composition (e.g., applying Boolean operations to areas);
- The X3D dialect could have been used for describing in detail the payload areas and sensor dimensions (or even the rough external aspect of the vehicles; Web3DConsortium, 2008). Even though the information contained in the language definition is enough for the purposes of this simulation platform, a more detailed description of the geometry of payload areas and cargo could help automate the process of matching sensors to payloads (Allen et al., 2009);
- SensorML (OGC, 2007*b*) could have been used to describe the sensors that each agent is equipped with, thus allowing for higher interoperability (Aloisio et al., 2006);
- The Aeronautical Information Exchange Model could have been used in the description of some aeronautical structures, namely, within the airport (Brunk & Porosnicu, 2005).

However, we decided not to implement these features on the short term (instead using simplified versions of several concepts from these dialects), mainly because of two reasons: it would represent a level of detail that was not intended in the languages and small tools can be easily developed to make the conversion between dialects (or the SDL and TDL definitions can be changed to allow both forms of element definition).

A complete description of the road system, as well as maritime and aerial routes (or at least the corridors for the airports) could have been also included, either as information explicitly present on the files or as a specification of a source for the information, such as a geographic information system-based database or an external service (Huang et al., 2009).

As previously mentioned, flight simulators have detailed descriptions of airports and their structures. As such, a tool could also be developed to extract that information from a flight simulator, such as FSX, and convert it into SDL, thus automating the process of creating airport descriptions, and easily building a set of bases of operations that could be used for aerial operations.

Further feedback from the community and improvement suggestions are welcome, as to advance these dialects into an extent where they will become more and more of a standard, providing support for several projects, and in several areas.

ACKNOWLEDGMENTS

The authors thank the Artificial Intelligence and Computer Science Laboratory for the excellent laboratory conditions and all of those who provided feedback about the implementation of SDL and TDL, especially Abel Santos and Pedro Daniel Sousa.

REFERENCES

Abreu, P.H., Mendes-Moreira, J., Costa, I., Castelão, D., Reis, L.P., & Garganta, J. (2012). Human versus virtual robotics soccer: A technical analysis. *European Journal of Sport Science 12(1)*, 26–36.

Allen, S., Burke, E.K., Hyde, M.R., & Kendall, G. (2009). Evolving reusable 3D packing heuristics with genetic programming. *Proc. 11th Annual Genetic and Evolutionary Computation Conf.*, GECCO 2009 (Rothlauf, F., Ed.), pp. 931–938. New York: ACM.

Almeida, F., Abreu, P.H., Lau, N., & Reis, L.P. (2013). An automatic approach to extract goal plans from soccer simulated matches. *Soft Computing 17(5)*, 835–848.

Aloisio, G., Conte, D., Elefante, C., Epicoco, I., Marra, G.P., Mastrantonio, G., & Quarta, G. (2006). SensorML for grid sensor networks. *Proc. 2006 Int. Conf. Grid Computing & Applications, GCA 2006*, pp. 147–152, Las Vegas, NV, June 26–29.

ANSI/AIIM. (2009). *Standard Recommended Practice—Strategy Markup Language: Part 1. StratML Core*. Standard ANSI/AIIM 21. New York: ANSI/AIIM.

Brooke, J. (1996). SUS—A quick and dirty usability scale. In *Usability Evaluation in Industry* (Jordan, P.W., Thomas, B., McClelland, I.L., & Weerdmeester, B., Eds.), pp. 189–194. London: Taylor & Francis.

Brunk, B.K., & Porosnicu, E. (2004). A tour of the AIXM concepts. *24th Annual ESRI Int. User Conf.*, San Diego, CA, August 9–13.

Brunk, B.K., & Porosnicu, E. (2005). Aeronautical information exchange model (AIXM) GIS interoperability through GML. *Proc. 25th Annual ESRI Int. User Conf.*, San Diego, CA, July 25–29.

Brunner, H., Mikula, A., & Eier, D. (2007). A concept for service based information quality and safety enhancement in aeronautical information management. *Proc. 52nd Annual Conf. Air Traffic Control Association 2007*, pp. 43–47, Washington, DC, October 28–31.

Camara, A., Silva, D.C., Abreu, P.H., & Oliveira, E. (2014). Comparing centralized and decentralized multi-agent approaches to air traffic control. *Proc. 28th European Simulation and Modelling Conf., ESM'2014*, pp. 189–193, Porto, Portugal, October 22–24.

Casbeer, D.W., Kingston, D.B., Beard, R.W., & McLain, T.W. (2006). Cooperative forest fire surveillance using a team of small unmanned air vehicles. *International Journal of Systems Science 37(6)*, 351–360.

Deursen, D.V., Bruyne, S.D., Lancker, W.V., Neve, W.D., Schrijver, D.D., Hellwagner, H., & de Walle, R.V. (2007). Mu-MiVA: a multimedia delivery platform using format-agnostic, XML-driven content adaptation. *Proc. 9th IEEE Int. Symp. Multimedia (ISM '07)*, pp. 131–138. Los Alamitos, CA: IEEE Computer Society.

Fan, H., Meng, L., & Jahnke, M. (2009). Generalization of 3D buildings modelled by CityGML. *Advances in GIScience: Proc. 12th AGILE Conf.* (Sester, M., Bernard, L., & Paelke, V., Eds.), Lecture Notes in Geoinformation and Cartography, pp. 387–405. Berlin: Springer.

Federal Aviation Administration. (2010). *Temporary flight restrictions*. Accessed at http://tfr.faa.gov/tfr2/list.html

Flight One Software, Inc. (2009). *Airport Facilitator X Manual*, 1.08 ed. Accessed at http://www.flight1.com

Georgieva, A., & Georgiev, B. (2010). Nontraditional approach to XML web services interactions. *Proc. 5th Int. Conf. Internet and Web Applications and Services*, pp. 67–72. Los Alamitos, CA: IEEE Computer Society.

Gimenes, R., Silva, D.C., Reis, L.P., & Oliveira, E. (2008). Flight simulation environments applied to agent-based autonomous UAVs. *Proc. 10th Int. Conf. Enterprise Information Systems (ICEIS 2008)*, pp. 243–246, Barcelona, June 12–16.

Groppe, S., Groppe, J., Böttcher, S., Wycisk, T., & Gruenwald, L. (2009). Optimizing the execution of XSLT style sheets for querying transformed XML data. *Knowledge and Information Systems 18(3)*, 331–391.

Hobbs, R.L. (2003). Using XML to support military decision-making. *Proc. XML Conf. Exposition 2003, XML 2003*, Philadelphia, PA, December 7–12.

Huang, C.-H., Chuang, T.-R., Deng, D.-P., & Lee, H.-M. (2009). Building GML-native web-based geographic information systems. *Computers & Geosciences 35(9)*, 1802–1816.

Kumar, C.S., Govardhan, A., & Rao, C.G. (2009). Usage of XML technology in electronic health record for effective heterogeneous systems integration in healthcare. *International Journal of Medical Engineering and Informatics 1(4)*, 399–406.

Lewis, J.R., & Sauro, J. (2009). The factor structure of the system usability scale. *Proc. 1st Int. Conf. Human Centered Design*, pp. 94–103. San Diego, CA: Springer–Verlag.

Masterson, J., Keeshan, B., & Hauck, H. (2009). *Airport design editor use manual, 1.47 ed*. ScruffyDuck Software Company. Accessed at http://www.scruffyduck.org/airport-design-editor/4584106799

Microsoft Corporation. (2008). *Compiling BGL*. Microsoft Developer Network, Microsoft ESP SDK. Accessed at http://msdn.microsoft.com/en-us/library/cc526978.aspx

Microsoft Corporation. (2010). *XML Schema Definition Tool (Xsd.exe)*. Microsoft Developer Network Library. Accessed at http://msdn.microsoft.com/en-us/library/x6c1kb0s.aspx

Open Geospatial Consortium. (2007*a*). *OpenGIS Geography Markup Language (GML) Encoding Standard*. OpenGIS Standard OGC 07–036, Open Geospatial Consortium Inc. Accessed at http://www.opengeospatial.org/

Open Geospatial Consortium. (2007*b*). *OpenGIS Sensor Model Language (SensorML) Implementation Specification*. OpenGIS Implementation Specification OGC 07–000, Open Geospatial Consortium Inc. Accessed at http://www.opengeospatial.org/

Open Geospatial Consortium. (2008). *OpenGIS City Geography Markup Language (CityGML) Encoding Standard*. OpenGIS Encoding Standard OGC 08–007r1, Open Geospatial Consortium Inc. Accessed at http://www.opengeospatial.org/

Peel, R. (2001). *Airport, navigation aid and IFR intersection data in FlightGear*. FlightGear Development Documents. Accessed at http://www.flightgear.org/Docs/AirNav/AptNavFAQ.FlightGear.html

Peel, R. (2009). *X-Plane Airport Data (Apt.Dat) file epecification, edition 850*. Accessed at http://data.x-plane.com/file_specs/XP\%20APT850\%20Spec.pdf

Rodrigues, C., Silva, D.C., Rossetti, R.J.F., & Oliveira, E. (2015). Distributed flight simulation environment using Flight Simulator X. *Proc. 10th Iberian Conf. Information Systems and Technologies*, pp. 1293–1297, Águeda, Portugal, June 17–20.

Santos, A. (2010). *Autonomous intelligent vehicle adaptation and performance analysis in Flight Simulator X*. Master's thesis. University of Porto.

Schweiger, R., Brumhard, M., Hoelzer, S., & Dudeck, J. (2005). Implementing health care systems using XML standards. *International Journal of Medical Informatics 74(2)*, 267–277.

Silva, D.C., Abreu, P.H., Reis, L.P., & Oliveira, E. (2014). Development of a flexible language for mission description for multi-robot missions. *Information Sciences 288*, 27–44.

Silva, D.C., Abreu, P.H., Reis, L.P., & Oliveira, E. (2015). Development of a flexible language for disturbance description for multi-robot missions. *Journal of Simulation*. Advance online publication. doi:10.1057/jos.2015.4.

Sousa, P.D. (2010). *Autonomous air traffic control for intelligent vehicles using Microsoft Flight Simulator X*. Master's thesis. University of Porto.

Sousa, P.D., Silva, D.C., & Reis, L.P. (2010). Air traffic control with Microsoft Flight Simulator X. *Proc. 5th Iberic Conf. Information Systems and Technologies (CISTI 2010)*, pp. 378–383, Santiago de Compostela, Spain, June 16–19.

Web3D Consortium. (2008). *Extensible 3D (X3D)*. ISO Standard ISO/IEC 19775–1:2008, Web3D Consortium, Inc. Accessed at http://www.web3d.org/

Wittman, R.L., Jr. (2009). Defining a standard: The military scenario definition language—version 1.0 standard. *Proc. 2009 Spring Simulation Multiconference (SpringSim 2009)* (Wainer, G.A., Shaffer, C.A., McGraw, R.M., & Chinni, M.J., Eds.). San Diego, CA: SCS/ACM.

Daniel Castro Silva is an Assistant Professor in the Faculty of Engineering at the University of Porto, where he received his PhD in artificial intelligence. His interests include artificial intelligence, including cooperation and coordination in multiagent systems, traffic management, human–machine interaction, and information visualization. Daniel has authored over 40 conference and journal papers over the last decade.

Pedro Henriques Abreu is an Assistant Professor in the Department of Informatics Engineering at the University of Coimbra. He attained his informatics engineering degree and a PhD in soccer teams modeling from the University of Porto. His interests include medical informatics and personal healthcare systems applied to oncological diseases. Pedro is the author of more than 40 publications in international conferences and journals.

Luis Paulo Reis is an Associate Professor at the University of Minho, a member of the Directive Board of the LIACC Laboratory, and Coordinator of the Human–Machine Intelligent Cooperation Research Group in Portugal. He was principal investigator of more than 10 research projects in the areas of artificial intelligence and robotics, including FC Portugal, and three time World Champion and eight time European Champion at RoboCup. Luis also won more than 30 other scientific awards. He supervised 13 PhD theses and 80 MS theses to completion and is the author of more than 250 publicatins in international conferences and journals. He is the president of the Portuguese Robotics Society.

Eugénio Oliveira is a Professor at the University of Porto. He coordinates a research group on distributed artificial intelligence in the LIACC Lab and a doctoral program in informatics. He obtained his PhD in knowledge engineering/logic programming from the Universidade Nova in Lisbon. He was Guest Academic at IBM/IEC, Belgium, and was awarded the Gulbenkian Prize. Eugénio has participated in European and nationally funded projects involving intelligent agents. His current topics of interest are agent-based frameworks for B2B, multiagent learning, and agent-based teamwork. He has published a number of papers in journals and proceedings.