ANIMO, Framework to simplify the real-time distributed communication

Y. Rodríguez, C. Alejo, I. Alejo and A. Viguria

Center for Advanced Aerospace Technologies (CATEC), 41309, La Rinconada (Sevilla), Spain
{yrodriguez, calejo, ialejo, aviguria}@catec.aero

Keywords: Software communication layer, Real-time, Distributed, Multi-systems, Quality of Service, Interoperability

Abstract. This paper presents a communication framework developed for interconnecting multi-systems based on the Data Distribution Service (DDS). The new built framework, called ANIMO, facilitates the integration of DDS in an application and the interoperability between the different data types of the Cooperating Objects (COs) with the great feature of real-time. Furthermore, a powerful tool to generate code has been developed bringing the incorporation and the updating of the data types as a very easy and simple task. In addition, a novel module has been performed to give the capacity to communicate the ANIMO framework with the ROS middleware that makes even more easier the integration of mobile robots as another Cooperating Object. This paper explains the complete architecture of the ANIMO framework, its diversity of possibilities and two principal works where it has been applied. One is a distributed simulator to validate embedded control algorithms. The other is the task of the supervision and message passing between quadrotors in an experiment of coordination and cooperation involving multiple aerial vehicles.

1 Introduction

The increasing usage of heterogeneous systems jointly, such as variety of sensors, microcontrollers, PCs, robots, etc., and which are distributed in an environment, implicates a more effort on inevitable tasks of integration and communication between them. In these cases, where the importance of the work lies in the development of applications and algorithms and not how the system data are available and interchangeable, a communication software fast tuning and high performance is desired. Also, many engineering applications related to smart cities require this data fusion to be performed in real time. For instance, efficient distributed video surveillance requires that real-time constraints be respected or, at least, quality of service guarantees (QoS) be provided [1]. In order to fulfill this necessity, the communication framework, called ANIMO, has been developed. Along this paper, the ANIMO framework will be explained.

The communication model underlying in network software is the most important factor in how applications communicate. The communications model impacts the performance, the ease to accomplish different communication transactions, the nature of detecting errors, and the robustness to different error conditions. Unfortunately, different communications models are better suited to handle different classes of application domains. The three main types of network communications models are: point-to-point, client-server and publish-subscribe.

For many applications related to cooperating objects in smart cities, the publish-subscribe communications model is the one that best suit its requirements since it is the best choice for systems with complex time-critical data flows [2]. Computer applications (nodes) "subscribe" to data they need and "publish" data they want to share. Messages pass directly between the publisher and the subscribers, rather than moving into and out of a centralized server. Publish-subscribe communication architectures are good for distributing large quantities of time-sensitive information efficiently, even in the presence of unreliable delivery mechanisms.

Taking into account the above, distributed applications and satisfying real-time, the ANIMO framework has been developed based on open standard of the Data Distribution Service for Real-Time Systems (DDS) from the Object Management Group (OMG). A well known open-source implementation of the DDS is the RTI (Real Time Innovations) DDS, which has been used in this work. Studies show that DDS implementations perform significantly better than non-DDS alternatives and are well-suited for certain classes of data-critical DRE information management systems [3].

In addition, other software modules have been built facilitating the integration of systems and devices, the interoperability between their different data types and the maintenance and enlargement of the framework itself. Moreover, the network features needed for an application can be adjusted to the network constraints through a wide Quality of Service controls (configuration parameters), achieving a high throughput.

The paper is organized as follows. In the next section the new developed framework, called ANIMO, based on the DDS is described. In Section 3, a brief description of two research projects, where the ANIMO framework has been successfully applied, is exposed and illustrated with results obtained from experiments. Finally, the last section is devoted to final conclusions.

2 The ANIMO Framework: Aiding to the Systems Intercommunication

As commented, the ANIMO framework has been developed to make easier the integration between different heterogeneous systems such as sensors, robots, systems, etc. that can be part of any smart cities application. It has a scalable architecture that is formed of the following modules:

- The communication layer (CL), it is the framework core based on DDS.

- The hardware abstraction layers (HAL). There is a HAL for each device or Cooperating Object to be incorporated into the framework. This layer is responsible for abstracting the peculiarities of the device and converting the data types used by its driver to the types defined in ANIMO. The HALs are libraries that wrap the device drivers to integrate them easily.
- The services of devices as high level applications (HLAs), that using the communication layer and the hardware abstraction layer, send the device data and receive the control command by DDS. There are already several devices integrated in ANIMO such as the haptics, the cyber gloves, the IS900 tracker, the webcams and the testbed accessing to the quadrotors. In this way, the devices are available for the possible HLAs. It is shown in Figure 1.

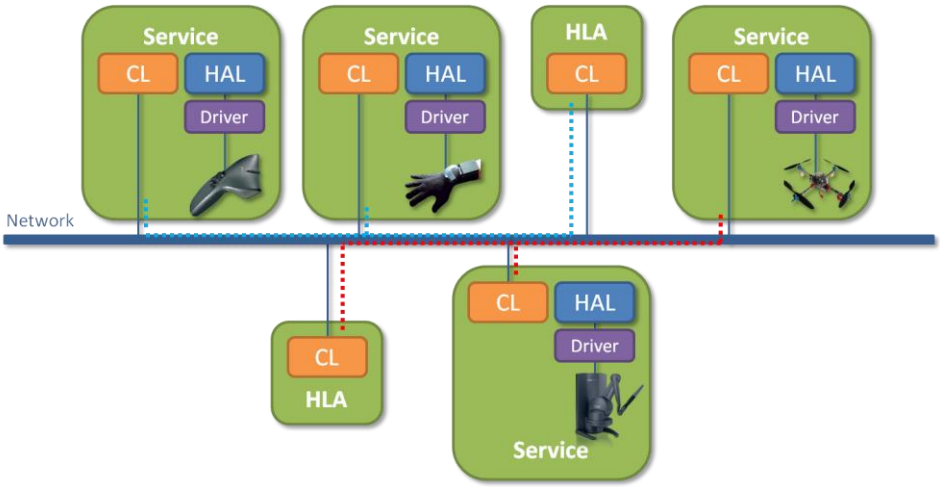

Fig. 1. Example of system intercommunication using the ANIMO framework

- The ROS Bridge module gives the capacity to communicate the ANIMO framework with the ROS middleware[1] that makes even more easier the integration of mobile robots as another Cooperating Object, i.e., the ROS world is accessible in ANIMO. This characteristic could be very interesting for smart cities applications where it is needed to interconnect robots with other smart sensors.
- A powerful tool to generate code brings the ANIMO extension as a very easy and simple task.

The benefits that the ANIMO framework provides to the applications are numerous. For example it accelerates development since it replaces low-level communications code with high-level interfaces. Also, it combines asynchronous publish/subscribe

[1] ROS (Robot Operating System) provides libraries and tools to help software developers create robot applications. It provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more. ROS is licensed under an open source, BSD license [4].

messaging and real-time data management, it automatically discovers and routes data between publishers (writers) and subscribers (readers) to the same topic, eliminates dependence on start-up order, automatically synchronizes state if there are disconnections or/and reconnections, and a change of the QoS used in the application means only a change in the configuration file avoiding the recompilation of the application.

2.1 Architecture of the Communication Layer

Given that the DDS API is very extensive and it is desired that the communication layer be reusable and simple, hiding the complexity of DDS, the ANIMO framework has built its own communication layer which can be integrated into any application requiring distributed software. Its architecture is detailed in Figure 2.

The communication layer is implemented in the C++ programming language and it is available for Linux and Windows platforms as dynamic library.

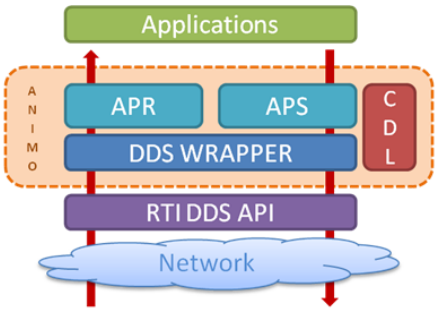

Fig. 2. The ANIMO communication layer architecture

The ANIMO framework has a first layer of DDS abstraction (DDS wrapper). It creates the topics, the participants, the publishers, the data writers, the subscribers and the readers. However, in order to further facilitate the use of the framework, ANIMO integrates another layer of abstraction that provides access points for sending (APS) and receiving (APR) data to the applications reducing the number of objects to create.

The common data types or structures that are exchanged between the applications are defined in Common Data Library (CDL), which gives support to the communication layer and the applications. At present, the CDL has more than forty different data types like position, rotation, velocity, angular speed, flight plan, MAVLink messages, etc. Thanks to the powerful tool built to generate code, the incorporation and the updating of the data types is a very easy and simple task.

2.2 A rapid and easy way to extend the ANIMO Framework

A feature that adds more value to the framework is to have a code generator for updating the framework. The code generator is joined to an useful GUI program helps to the developer in the maintenance task. From a description file of the data types that will travel through the network, all the ANIMO framework code that depends on the

data types is self-generated. Then, these files are copied to the respective projects and compiled. Finally, the installation packages are built to reinstall them updating the framework libraries on the computer.

XML has been chosen as the description file format because it is well-known by most developers. Managing the information that contains; it is possible to auto generate the ROS messages and actions for the ROS packages and ROS Bridge module, and the IDLs and the C++ code for the ANIMO framework core. The code generator is based on the Apache Velocity Project [5].

2.3 Connecting with the ROS Middleware

A novel module, called ROS Bridge, has been performed to give the capacity to communicate the ANIMO framework with the ROS middleware that makes even more easier the integration of mobile robots as another Cooperating Object. More specifically, ROS Bridge allows the automatic generation of a data parser between the ANIMO framework and ROS using a XML configuration file. In Figure 3 can be observed the link of the ANIMO framework core with the ROS Bridge module. The ROS Bridge API wrappers the ROS API providing methods for the publication and the subscription of the ROS messages. Moreover, the separate ROS networks can be communicated by a DDS network using the ROS Bridge module. In this way, the isolated clients and servers of the ROS actions achieve to be connected and are able to interchange data across the ANIMO framework.

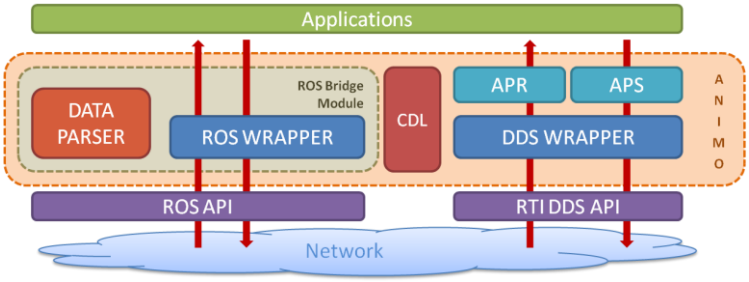

Fig. 3. The ROS Bridge module architecture

3 Example of Application Projects

The following sections explain two principal works where ANIMO has been applied. One is a distributed simulator to validate real embedded control algorithms. The simulator has been developed in the European Commission's 7th Framework Programme (FP7) project, EC-SAFEMOBIL [6]. The other is the task of the supervision and message passing between quadrotors in an experiment of coordination and cooperation involving multiple aerial vehicles. This task is still developing in the ARCAS project [8]; it is also a FP7 project.

In closing, appoint other works where ANIMO is involved:

- Real-time visualization of the experiments being carried out in the testbed of FADA-CATEC through a scale virtual world, with the possibility of 3D projection.
- Manipulation of an articulated arm of two degrees of freedom using an haptic device.
- Simulator for the formation of operators of Remotely Piloted Aircraft Systems (RPAS), implicating communication between the Ground Control Station (GCS), the simulation engine, the simulated aircraft with real embedded control algorithms, the instructor station and the databases. This project is still developing.
- Human Machine Interface (HMI) for the navigation in a 3D virtual world (zoom, translation and rotation) commanded from the gestures collected using a cyber glove and the IS900 tracker.

3.1 Distributed Simulator to Validate Embedded Control Algorithms

The EC-SAFEMOBIL project is devoted to the development of sufficiently accurate common motion estimation and control methods and technologies in order to reach levels of reliability and safety to facilitate unmanned vehicle deployment in a broad range of applications [6]. With this goal, several scenarios are defined in the project to be performed both in simulation and in real experiments. The distributed simulator offers the opportunity to test the developed methods in the project in a repeatable and thorough manner, validating embedded control algorithms before executing them in a real experiment, in regard scalability, fault-tolerance, trajectory planning and cooperative tracking. The most important scenarios are:

- Rotary-wing UAV landing on a ship (see Figure 4 (a)).
- Autonomous distributed warehousing traffic and control (see Figure 4 (b)).
- Tracking for surveillance. Many UAVs (Unmanned Aerial Vehicles) follow to their targets (see Figure 4 (c)).

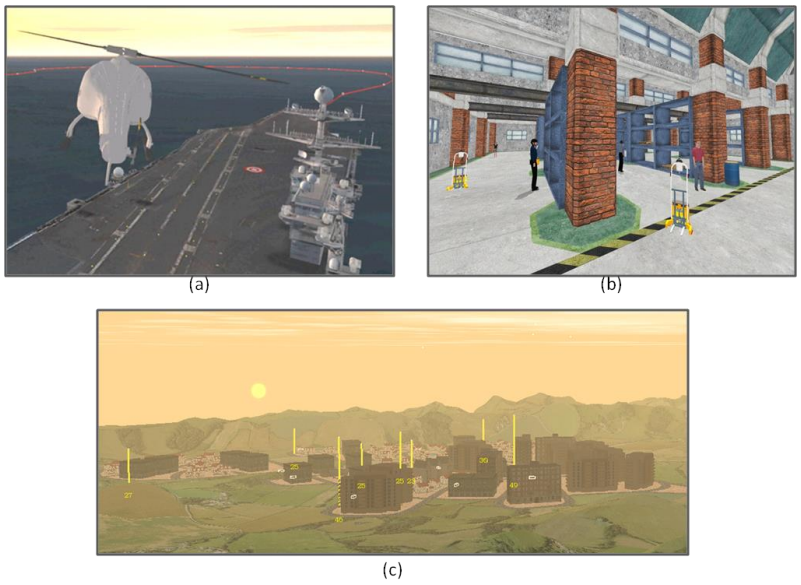

Fig. 4. Images of the graphic interface of the simulator

The simulator has been developed based on the VT MÄK tools [7], VR-Forces (back-end module or simulation engine) and VR-Vantage (front-end module or visual image generator), and has the great feature that, the simulations can be visualized in 3D. The rotary-wing UAV landing on a ship scenario will be explained as example. The modules are implicated in this scenario constituting the simulator framework is shown in Figure 5. There are two programs exchange data by using the ANIMO framework core[2]. One is the simulation engine; it is responsible of the synchronization between all modules managing the simulation time. The other is a wrapper of the real UAV control embedded algorithms. The developed control in Simulink is generated as C++ classes which are integrated without changes in the simulator framework. In the side of the simulator, the ship, the radio beacon sensor[3] and the UAV are created like VR-Forces plugins. The UAV behaviour is simulated in a very truthful way using the auto generated code of the developed UAV model in Simulink. The control loop is implemented in the simulator as shown in Figure 5. In each simulation step, the RBS data

[2] In the tracking for surveillance scenario, can be observed in the Figure 4 (c), there are many UAVs and targets, therefore, the simulator framework is composed of a wrapper program for each UAV more the simulation engine program.

[3] It is the proposed sensor for the scenario. RBS System is composed of two subsystems, one located in the UAV itself and another one installed on the ground surrounding the landing area. The RBS airborne system provides a relative position to the UAV on board computer (relative to the moving landing pad, located on the ship, in body fixed frame).

are inputs in the control algorithms which act in the simulated UAV. After that, the new UAV state together the ship state update the simulated sensor closing the loop.

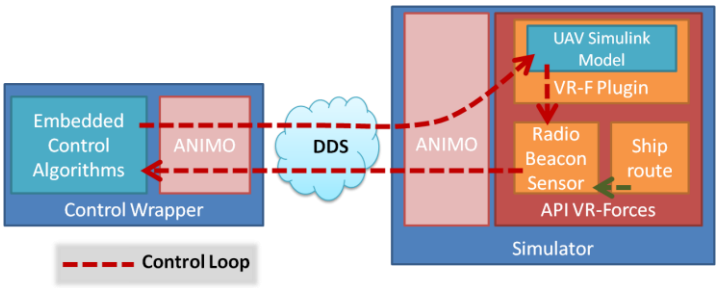

Fig. 5. Principal components of the distributed simulator in the UAV landing on a ship scenario

Figure 6 shows the evolution of the UAV and target position in wave conditions (a) and calm sea (b). When the sea is in calm, the UAV is able to land. However, when there are waves, it is necessary another helper mechanism to carry out the complete landing. The EC-SAFEMOBIL project proposes to use a tether linking the UAV and the platform (tether guiding) in order to increase the stability. This experiment is being conducted this year.

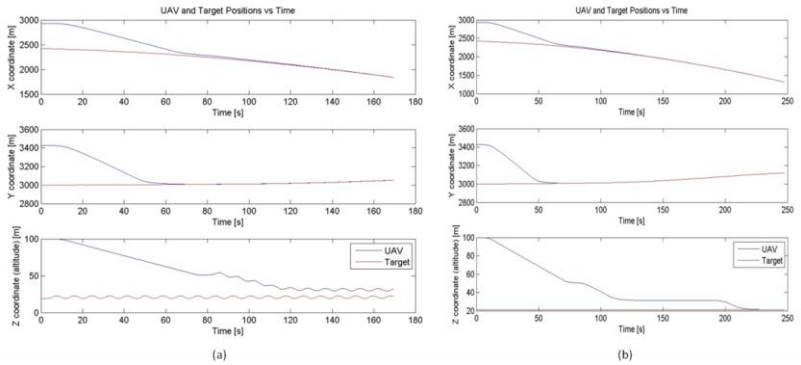

Fig. 6. The evolution of the UAV and target position in wave conditions (a) and calm sea (b) in the UAV landing on a ship scenario

3.2 Experiment of Coordination and Cooperation Involving Multiple Aerial Vehicles

The ARCAS project proposes the development and experimental validation of the first cooperative free-flying robot system for assembly and structure construction [8]. For the missions involve inspection tasks and translation of structures, experiments of coordination and cooperation of multiple aerial vehicles, avoidance collision, are proposed. At present, these experiments have been executed in simulation using Ga-

zebo (see Figure 7 (b)). The real experiments are programmed for early March this year using the testbed[4] of FADA-CATEC (see Figure 7 (a)).

The architecture of the multivehicle experiments is presented in Figure 8. The modules executed on the ground, the Trajectory Planning (TP) and the Global Supervisor (GS) are connected to a DDS network. Whereas that the modules executed on board, the Trajectory Generation (TG), the Collision Avoidance (CA) and the UAV Abstraction Layer (UAL) are connected by a ROS network. The two network are united thanks to the ROS Bridge (ROSB) forward the ANIMO topics to the ROS topic and vice versa. The ROSB runs on the ground. In the beginning, the GP sends some waypoints of the desired trajectory of all involved UAVs to the TP. Then, the TP, which knows the world, plans the trajectories for each UAV avoiding the obstacles of the area and send them. The trajectories arrive to the UAVs through the ROS Bridge module. At that point, the TG generates the control references for its UAV. These control references act in the low-level software using UAL. Furthermore, the control references are modified from CA if a collision with other UAVs is detected. For this it is necessary that each UAV knows the state of all the other UAVs, which is achieved using the ROS Bridge module.

Figure 9 shows the results of an experiment with three UAVs. Figure 9 (a) is an experiment without executing the CA module. Therefore, there is a collision implicating two UAVs which go to ground. In Figure 9 (b) can be observed that the collision is avoided because the CA module is being executed.

Fig. 7. Scenario of the first experiments of multivehicle in ARCAS. (a) Testbed of FADA-CATEC with mock-up. (b) Virtual environment in Gazebo for the previous simulations.

[4] This testbed is based on an indoor positioning system that uses 20 VICON cameras. This system can calculate the position and attitude of any moving object within the volume of the testbed (15x15x5 m) in real time (with an update rate of up to 500 Hz).

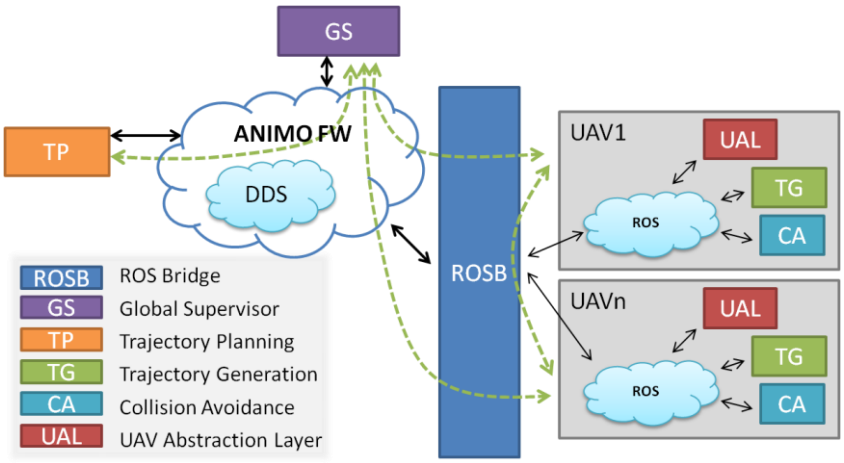

Fig. 8. Architecture of the experiment of coordination and cooperation involving multiple aerial vehicles

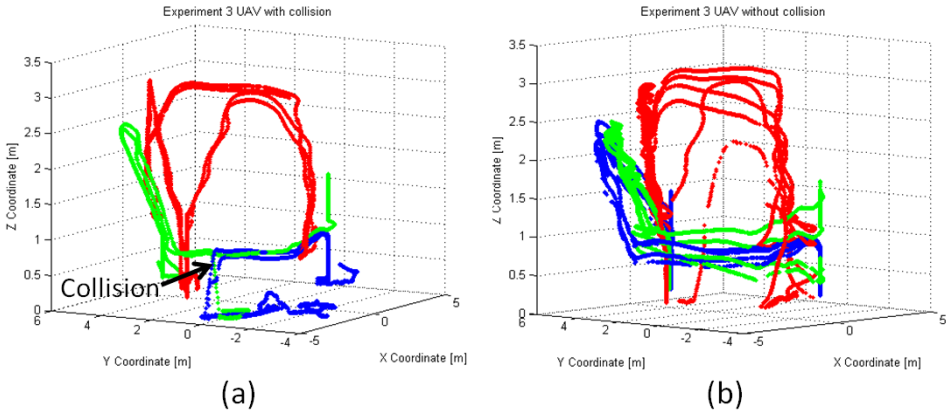

Fig. 9. Results of an experiment with three UAVs. (a) No executing CA module. (b) Executing CA module.

4 Conclusions

In this work, a complete communication software solution, with fast tuning and high performance, has been presented for Cooperating Object systems. It is especially appropriate in distributed system with real-time requirements.

Also, the novel modules have been explained allowing the interconnection with other middlewares like ROS and the extension of the framework with any information type without difficulty.

Finally, it has been shown how the developed ANIMO framework has been applied to two different projects with very different real-time communication needs showing the countless possibilities that the framework could have where applied to smart cities applications.

References

1. García-Valls, M., Basanta-Val, P.; Estévez-Ayres, I: Adaptive real-time video transmission over DDS In: Dept. of Telematics Eng., Univ. Carlos III de Madrid, Leganes, Spain.
2. Core Libraries and Utilities User's Manual of Real-Time Innovations, Version 4.5, http://www.rti.com
3. Xiong, M., Parsons, J., Edmondson, J., Nguyen, H., Schmidt, D. C.: Evaluating the Performance of Publish/Subscribe Platforms for Information Management in Distributed Real-time and Embedded Systems. In: Vanderbilt University, Nashville TN, USA
4. Official website of the ROS middleware, http://wiki.ros.org
5. The Apache Velocity Project, http://velocity.apache.org
6. Official website of the EC-SAFEMOBIL project, http://www.ec-safemobil-project.eu
7. Official website of VT MÄK, http://www.mak.com/
8. Official website of the ARCAS project, http://www.arcas-project.eu